Towards Unified Memory Adaptation for LLM Agents: Textual, Latent, and Parametric Pathways

Rui Li[1,*] **Haoran Tan**[1,*] **Xiaohe Bo**[1,*] **Yiqun Chen**[1] **Meizhi Zhong**[2]
Xiaochi Wei[2] **Yan Gao**[2] **Yi Wu**[2] **Yao Hu**[2] **Xu Chen**[1,†]
[1]Gaoling School of Artificial Intelligence, Renmin University of China
[2]Xiaohongshu Inc.
{lirui121200, tanhaoran1321, xiaohe, xu.chen}@ruc.edu.cn

Abstract

Equipping large language models (LLMs) with *external memory banks* is a critical stepping stone towards autonomous agents. Current approaches have concentrated substantial efforts on the design of memory construction and retrieval mechanisms. However, they typically relegate *memory utilization* to the simplistic concatenation of explicit texts for LLM inputs, thereby remaining constrained by the fundamental bottlenecks of *context overload* and *semantic dilution*. In response to this dilemma, we present **UniMem**, a unified memory adaptation framework that empowers LLM agents to holistically harness memory information, extending beyond *textual* signals into *latent* and *parametric* dimensions. Operating along these pathways, UniMem not only elevates contextual density through *latent compression*, but also facilitates semantic grounding via *parametric modulation*. Extensive experiments show that UniMem (I) achieves superior performance across diverse benchmarks, including long-term dialogue, memory personalization, and open-domain question answering; and (II) can seamlessly complement prevailing memory construction and retrieval mechanisms for synergistic enhancements. The code is available at xxrep/UniMem.

1 Introduction

Recent advancements in large language models (LLMs) [1–5] have driven significant breakthroughs in language understanding and complex reasoning. Despite their immense potential as general-purpose agent engines [6–8], LLMs are inherently hindered by their static parameter weights and finite context windows when operating in long-term interactive environments [9–12]. These limitations underscore the imperative of a cohesive *memory bank* [13–15] for autonomous agents, enabling them to broaden cognitive horizons based on episodic history [16–20] and acquired knowledge [21–26].

Current prevailing approaches have dedicated substantial efforts to the design of memory mechanisms, focusing either on memory construction (e.g., from tabular [16] to structured representations [26, 27]) or on memory retrieval (e.g., from global [19] to multi-grained search [21, 28]). However, these works essentially overlook the subsequent phase of *memory adaptation*, giving rise to such a critical question:

How can LLM agents effectively harness memories to adapt their behaviors on the fly?

The mainstream practice for memory adaptation revolves around *textual* prompt augmentation, where the retrieved memory fragments are directly appended to LLMs' input prompts [14, 29]. Despite its simplicity, this shallow textual interaction inherently traps the agents in two fundamental bottlenecks, as shown in Figure 1. On one hand, the retrieved memory entries inevitably harbor redundant signals and irrelevant noises [30], such that their direct concatenation would diminish the signal-to-noise ratio within the finite context window (i.e., *Context Overload* [10, 11, 31]), thereby obscuring the salience of genuinely pertinent information and precipitating marked degradation in response quality [32, 33].

*Equal contributions. †Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

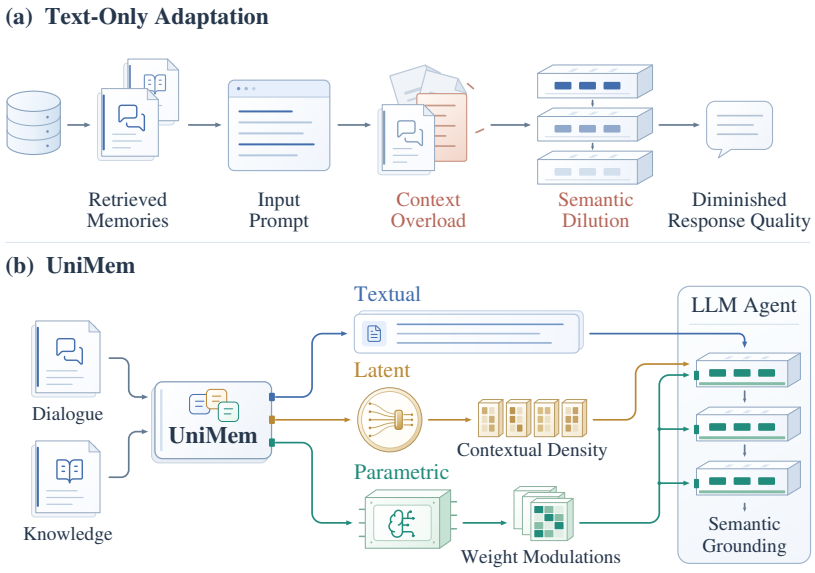

Figure 1: Illustrations of text-only and UniMem memory adaptation paradigms.

On the other hand, prompt-level memory injection confines its influence to the input embedding layer, beyond which the semantic signals originating from appended memories undergo progressive attenuation through successive transformer blocks (i.e., *Semantic Dilution* [34–36]), thereby rendering the LLM agent increasingly oblivious to the relevant memory it was intended to leverage [37, 38]. These limitations reveal the need for memory adaptation approaches beyond superficial textual interactions.

To this end, we present **UniMem**, a unified memory adaptation framework that empowers LLM agents to harness memory information across multiple dimensions. Specifically, UniMem moves beyond the plain *textual* signals, forging *latent* and *parametric* pathways to weave memories for agentic reasoning. Along the latent pathway, UniMem learns to compress verbose memories into compact representations, which function as dense contextual signals capturing task-relevant semantics within continuous spaces. On top of such latent-enriched context, our proposal further shifts the memory adaptation process from superficial injection to deep internalization via a parametric pathway. Along this pathway, UniMem coherently generates instance-specific weight modulations to ground memory semantics throughout the agent's entire network depth, preventing memories from fading across successive transformer blocks. Ultimately, by harmonizing the *latent compressor* and the *parametric modulator* within a *lightweight optimizable architecture*, our design effectively circumvents the fundamental bottlenecks of simplistic textual augmentation, delivering a flexible and dynamic memory adaptation paradigm for LLM agents. Moreover, UniMem is agnostic to the upstream memory construction and retrieval mechanisms, such that it can be naturally integrated with existing designs to yield synergistic performance enhancements.

Empirically, we systematically evaluate UniMem across diverse agentic scenarios, spanning long-term dialogue, memory personalization, and open-domain question answering. The extensive experimental results demonstrate that UniMem (I) consistently achieves superior performance over all benchmarks without compromising the inference efficiency; (II) exhibits seamless compatibility with off-the-shelf memory designs; and (III) showcases profound potential to generalize across diverse memory regimes, from curated knowledge to experiential rollouts. Our main contributions are summarized as follows:

- **Blueprint:** Moving beyond the prevalent research focus on memory construction and retrieval, we present a multi-pathway blueprint dedicated to the critical phase of *memory adaptation*. This blueprint seeks to mitigate the dual limitations of context overload and semantic dilution inherent in superficial textual memories through the integration of *latent* and *parametric* memory pathways.
- **Framework:** We then develop a prototype for unified memory adaptation (UniMem) to coherently realize *latent compression* and *parametric modulation* within a lightweight end-to-end framework, empowering LLM agents with superior capabilities to holistically harness the memory information.
- **Evaluation:** Extensive experiments across diverse benchmarks demonstrate that our framework effectively synthesizes textual, latent, and parametric pathways for memory adaptation, delivering consistent performance superiority and seamless compatibility with existing memory mechanisms.

2 Background

Memory-Augmented LLM Agents. The integration of external memory modules into LLM agents has emerged as a critical research direction for enhancing long-term interactive capabilities [13–15]. A prominent line of research investigates *how to organize the memories* (i.e., memory construction), exploring diverse memory representations spanning from tabular formats [16, 39] to knowledge graphs [27, 40, 41] and hierarchical schemas [26, 28, 42]. In tandem with structural design, a parallel body of work centers on *how to recall task-relevant memories* (i.e., memory retrieval), typically leveraging various retrieval techniques such as sparse [43, 44], dense [45, 46], and multi-grained retrieval [26]. Nevertheless, once the memory fragments are retrieved, current approaches often directly append the lengthy memory texts to the input prompts of LLMs for inference. Such simplistic textual interactions inherently encounter the dual challenges of context overload [10, 11] and semantic dilution [34–36].

Context Compression for LLMs. To alleviate the context capacity bottleneck, existing literature has explored various compression techniques to enhance information throughput. One prevalent line of methods focuses on prompt compression within the natural language space, preserving core semantics through textual summarization [47–49]. Recent research exhibits a compelling trend towards compression in the continuous latent space [50] for greater expressive capacity, such as encoding contexts into compact latent vectors to reduce storage and computational overhead [51–53], or utilizing soft prompt tuning [35] to derive the task-specific virtual tokens [54–56]. These approaches collectively inspire our *latent* pathway, which compresses memories into dense representations to enhance contextual density.

Parametric Modulation for LLMs. Imbuing LLMs with task-specific parametric knowledge represents another core challenge in the literature. Parameter-efficient fine-tuning techniques, epitomized by Low-Rank Adaptation (LoRA) [57–59], have emerged as the standard paradigm for agent tuning [60–62], allowing the agents to acquire new knowledge while keeping vast pre-trained weights frozen. Despite their efficacy, the newly learned parameters remain fixed during inference, thereby inherently depriving the agents of dynamic responsiveness to real-time contexts [63]. To alleviate this limitation, a promising avenue lies in input-conditioned parameter modulation, enabling the model to adjust its internal weights based on the varying inputs. For instance, hypernetwork-based architectures [64–67] utilize specialized auxiliary modules to directly generate or predict weights for target neural networks. Similarly, recent knowledge editing techniques [68–70] also demonstrate the feasibility of injecting new knowledge into LLMs via localized parameter modifications. Inspired by these explorations, our *parametric* pathway further advances the memory adaptation towards dynamic parametric modulation, enabling LLM agents to instantly tailor memory-aware adapter weights for deep semantic grounding.

Positioning of UniMem. Prevailing efforts focus on the construction and retrieval of external memories, yet they typically reduce the crucial phase of memory adaptation to simplistic text concatenation. In this paper, we bridge this gap by shifting the research lens towards *how LLM agents can effectively harness memories to adapt their behaviors*. To overcome the dual bottlenecks of context overload and semantic dilution inherent in singular textual pathways, we introduce **UniMem**, a unified framework that cohesively integrates the principles of latent compression and parametric modulation for memory adaptation. This framework not only encodes verbose memories into compact latent representations to enhance contextual density, but also dynamically modulates the agent's internal parameters to align its behavior with memories. Notably, UniMem is agnostic to upstream memory mechanisms, serving as a complementary design that synergizes with prevailing memory construction and retrieval strategies.

3 Preliminary

Memory-Augmented LLM Agents. We consider an LLM agent operating in a long-horizon interactive environment, in which its response to each query q must be grounded in an evolving memory bank $\mathcal{M} = \{m_1, \ldots, m_{|\mathcal{M}|}\}$ accumulated from past interactions or curated knowledge. The form of each entry m_i varies with the upstream construction strategy, ranging from raw dialogue traces and extracted facts to structured knowledge fragments. The query q specifies the agent's immediate task, such as a user utterance in long-term dialogue or a question over an external corpus. We denote the underlying LLM as f_θ with parameters θ, and write $p_\theta(y \mid \cdot)$ for its response distribution.

The Conventional Pipeline. The conventional memory-augmented pipeline factors into three sequential stages: (*i*) *memory construction* organizes raw interactions or curated knowledge into structured entries that constitute $\mathcal{M}$; (*ii*) *memory retrieval* selects a task-relevant subset $\mathcal{D} = \mathrm{Retrieve}(q, \mathcal{M}) =$

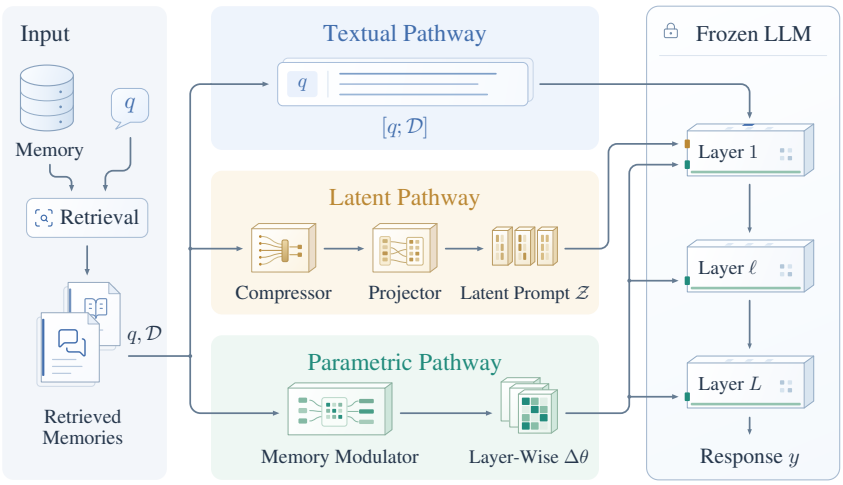

Figure 2: Overview of our **UniMem** framework for unified memory adaptation.

$\{d_1, \ldots, d_K\}$ via sparse, dense, or graph-based scoring [43, 45, 41]; and (*iii*) *memory adaptation* integrates $\mathcal{D}$ into the response process so that the agent answers as $p_\theta(y \mid q, \mathcal{D})$. Prior work has concentrated overwhelmingly on the upstream construction and retrieval stages, whereas adaptation has remained largely undeveloped, defaulting to plain prompt concatenation, $p_\theta(y \mid [q; \mathcal{D}])$, where $[\cdot;\cdot]$ stitches its operands into a single input. We refer to this default operator as the *textual pathway*.

Two Bottlenecks of the Textual Pathway. The textual pathway suffers from two intertwined limitations. First, retrieved entries $\mathcal{D}$ are intrinsically heterogeneous in salience, and even after careful retrieval a sizable fraction of tokens conveys redundant or off-topic content. Direct concatenation thus dilutes the signal-to-noise ratio inside the finite context window and triggers the well-documented *context overload* [10, 11, 31]. Second, prompt-level injection enters the network only at the input embedding layer, beyond which the salience of memory tokens decays through successive transformer blocks, an effect we refer to as *semantic dilution* [34–36]. Together, these two limitations expose a structural ceiling: no matter how carefully $\mathcal{M}$ is constructed or $\mathcal{D}$ is retrieved, the textual pathway alone cannot guarantee that memory signals will effectively steer the agent's downstream decisions.

Memory Adaptation Blueprint. To circumvent this ceiling, we elevate the memory adaptation stage from rigid textual concatenation into a *multi-pathway* formulation $\mathcal{A}$. This blueprint empowers LLM agents to effectively harness the memories across latent and parametric dimensions:

$$p(y \mid q, \mathcal{D}) \;=\; p_\theta\big(y \,\big|\, \mathcal{A}(q, \mathcal{D})\big), \quad \mathcal{A}(q, \mathcal{D}) \;=\; \big([q; \mathcal{D}],\; \mathcal{Z}(q, \mathcal{D}),\; \Delta\theta(q, \mathcal{D})\big), \tag{1}$$

in which $[q; \mathcal{D}]$ is the raw textual signal composed of the query q and the retrieved memory $\mathcal{D}$, while $\mathcal{Z}$ and $\Delta\theta$ are learnable adaptation operators that directly target the two aforementioned bottlenecks. The *latent* operator $\mathcal{Z}$ compresses $(q, \mathcal{D})$ into a fixed-length sequence of continuous representations, amplifying the signal-to-noise ratio of the system input and thus relieving the context overload issue. The *parametric* operator $\Delta\theta$ generates instance-specific weight modulations injected into the projection matrices of f_θ at every layer, thereby propagating memory salience throughout the network depth as a remedy for semantic dilution. Both operators are jointly optimized under the downstream task objective. Our UniMem framework, developed in Section 4, is the first realization of this formulation.

4 Methodology

In this section, we introduce **UniMem**, a unified framework that realizes the multi-pathway adaptation operators within an end-to-end optimizable architecture. Section 4.1 gives an overview of our design. In Sections 4.2 and 4.3, we then elaborate on the latent and parametric pathways of memory adaptation. Section 4.4 describes the inference pipeline and end-to-end optimization strategy.

4.1 Overall Architecture

Figure 2 depicts the overall architecture of our UniMem framework. Given a query q with a memory subset $\mathcal{D}$ retrieved from the external memory bank $\mathcal{M}$, UniMem realizes the multi-pathway adaptation

blueprint $\mathcal{A}$ in Eq. (1) as a lightweight optimizable architecture around the frozen inference LLM f_θ. The architecture comprises two trainable modules, a *Compressor* $\mathcal{C}_\phi$ for the latent pathway and a *Modulator* $\mathcal{H}_\xi$ for the parametric pathway, whose respective outputs are the latent prompt $\mathcal{Z}$ and an instance-specific weight modulation $\Delta\theta$. Together with the textual concatenation $[q; \mathcal{D}]$ inherited as the conventional LLM input, these three signals jointly condition f_θ's response generation.

The *Compressor* $\mathcal{C}_\phi$ encodes $(q, \mathcal{D})$ into a sequence of latent token embeddings $\mathcal{Z}$ and appends them to the input of f_θ, enhancing the contextual signal-to-noise ratio beyond the simplistic text concatenation. The Modulator $\mathcal{H}_\xi$ maps the semantic representation c of $(q, \mathcal{D})$ into a set of weight modulations $\Delta\theta$, which is applied to each LLM layer of f_θ, sustaining the memory signal across the full network depth to counteract semantic dilution. In this way, these two pathways operate synergistically along the contextual and parametric dimensions, jointly facilitating the memory adaptation efficacy for LLM agents.

Notably, only the parameters $\Phi = \{\phi, \psi, \xi, L\}$ are updated in the training phase, with the inference backbone f_θ remaining frozen throughout. Φ accounts for less than eight percent of the backbone size, such that the additional training and inference overhead of UniMem is modest. The entire pipeline is end-to-end differentiable through Φ, thus naturally admitting various agentic optimization techniques.

4.2 Latent Compression

General Form of Compressor. To mitigate the context overload issue, UniMem compresses the raw context $(q, \mathcal{D})$ into a fixed-length sequence of latent representations. This latent pathway is driven by a *Compressor* $\mathcal{C}_\phi$ that distills the inputs into k latent vectors, together with a projection module Proj_ψ that maps these compact representations into the input embedding space of f_θ:

$$\mathcal{Z} \;=\; \mathrm{Proj}_\psi(H) \in \mathbb{R}^{k\times h}, \qquad H \;=\; \mathcal{C}_\phi(q, \mathcal{D}) \in \mathbb{R}^{k\times h}. \tag{2}$$

Such aligned representations $\mathcal{Z}$ are then ingested by f_θ as the soft prompt at its input embedding layer.

Design Space. The design of $\mathcal{C}_\phi$ centers on the choice of its backbone model and compression mode. A compressor can be independent of f_θ, granting full design freedom in its own representation space at the cost of a heavier Proj_ψ that bridges into f_θ's input embedding space. Alternatively, it may share f_θ's parameters through a lightweight adapter, with outputs already aligned to f_θ's input space and Proj_ψ reducing to an affine mapping. For the compression mode, autoregressive encoding [71] models the latent vectors sequentially, with each conditioned on its predecessors to preserve causal structure in the compressed sequence. Parallel encoding [51] instead appends all learnable latents at once and reads them out via a single forward pass, which is faster but forgoes the recurrence among the latents.

Instantiation. In our instantiation, we employ the backbone-shared architecture to avoid the GPU memory overhead of loading additional LLM models. Specifically, $\mathcal{C}_\phi$ reuses the parameters of f_θ via a dedicated LoRA adapter ϕ, and this alignment effectively simplifies Proj_ψ into the affine mapping. For the compression mode, we adopt the parallel encoding rather than the autoregressive counterpart. As demonstrated in Appendix J, the latter provides only marginal advantages for the relatively short latent tokens, despite requiring one forward step for each latent. Formally, we append k learnable embeddings $L \in \mathbb{R}^{k\times h}$ to the concatenated input $[q; \mathcal{D}]$. Following a single forward pass, we extract the hidden states of these trailing positions as H, which are then projected to yield the soft prompt $\mathcal{Z}$.

4.3 Parametric Modulation

General Form of Modulator. Beyond the latent compression, UniMem also addresses the semantic dilution issue by injecting memories directly into the parameter weights of every network layer in f_θ. We follow the standard hypernetwork design [64, 72] to realize the parametric pathway. Formally, a shared Modulator $\mathcal{H}_\xi$ predicts per-(layer, type) LoRA factors from a concatenated input $x_{\ell,t}$,

$$A_{\ell,t},\, B_{\ell,t} \;=\; \mathcal{H}_\xi(x_{\ell,t}), \qquad x_{\ell,t} \;=\; \mathrm{concat}[\,c,\, E[t],\, E[\ell]\,], \tag{3}$$

where $c \in \mathbb{R}^h$ is the condition vector summarizing $(q, \mathcal{D})$, and E is a learnable embedding dictionary for any module type t or layer index ℓ. The weight perturbation then takes the LoRA-structured form:

$$\Delta W_{\ell,t} \;=\; \lambda \cdot A_{\ell,t}\, B_{\ell,t}^\top, \qquad A_{\ell,t} \in \mathbb{R}^{d_{\text{in}}\times r},\;\; B_{\ell,t} \in \mathbb{R}^{d_{\text{out}}\times r}, \tag{4}$$

with rank r, modulation strength $\lambda > 0$, and (ℓ, t) ranging over a chosen target set $\mathcal{S}$ of (layer, module-type) pairs. Each pair thus gains its own memory-conditioned LoRA adapter.

Design Space. This formulation is naturally compatible with diverse encoder backbones for producing c, ranging from a CLS token of a bidirectional transformer to the last-token activation of an LLM. One could specify the target set $\mathcal{S}$ to control which network layers and module types of f_θ are subject to the parametric modulation pathway.

Instantiation. In our instantiation, we extend the Compressor's parallel-mode forward pass (Section 4.2) to the parametric pathway by reserving k_c of its k latents, denoted by $L_c \in \mathbb{R}^{k_c \times h}$, whose hidden states are then pooled into the condition vector c. For $\mathcal{H}_\xi$, we use a lightweight MLP commonly adopted in hypernetwork designs, with per-type output heads producing the LoRA factors $(A_{\ell,t}, B_{\ell,t})$. The output heads are initialized to produce $\Delta W_{\ell,t} \approx 0$ at step zero, anchoring early training to the pre-trained behavior of f_θ. More details on such parametric modulation are included in Appendix D.

4.4 End-to-End Optimization

Inference Pipeline. Algorithm 1 traces how UniMem processes a single query end-to-end. A single forward pass through the Compressor produces both the latent prompt and the condition vector, the Modulator turns the latter into instance-specific weight perturbations, and the inference LLM then generates the response from the combined textual, latent, and parametric signals.

Optimization Strategy. Since UniMem is end-to-end differentiable through the adaptation operator $\mathcal{A}$, it naturally accommodates standard training paradigms such as supervised fine-tuning (SFT) and typical agentic reinforcement learning techniques, with f_θ frozen throughout and gradients flowing only through the trainable parameters $\Phi = \{\phi, \psi, \xi, L\}$. We follow a two-stage training curriculum:

$$\Phi^{(1)} \leftarrow \arg\min_{\Phi} \mathcal{L}_{\text{SFT}}(\Phi), \qquad \Phi^{\star} \leftarrow \arg\max_{\Phi} \mathcal{J}_{\text{GRPO}}\big(\Phi;\, \Phi^{(1)}\big). \tag{5}$$

Stage 1 minimizes the cross-entropy loss $\mathcal{L}_{\text{SFT}}$ on a self-collected pool $\mathcal{T}^+$ of positive rollouts whose exact-match or F1 against the gold response exceeds a threshold, warm-starting Φ for the next stage. Initialized at $\Phi^{(1)}$, Stage 2 then employs GRPO [73] to further refine Φ for the downstream task metric via reward-driven group-relative advantages of sampled rollouts, with KL anchoring to the SFT reference. The training procedure is described in Algorithm 2 (Appendix H). Notably, the textual pathway could be dispensable under sufficient end-to-end training, as $\mathcal{Z}$ is itself a dense representation of $(q, \mathcal{D})$ that already covers the salient information of the textual concatenation. In this sense, we retain $[q; \mathcal{D}]$ only for compatibility with the conventional LLM input interface.

5 Experiments

To comprehensively validate the effectiveness of UniMem, we conduct extensive experiments across a range of agentic scenarios spanning long-term dialogue, memory personalization, and open-domain question answering. In the main experiment (§5.2), we compare our UniMem framework against representative memory-augmented baselines to demonstrate its performance superiority. Next, we integrate UniMem with various memory construction and retrieval mechanisms to investigate their synergistic potential (§5.3). Finally, §5.4 provides more in-depth analysis of different configurations and variants.

5.1 Experimental Setup

Benchmarks. We evaluate UniMem on four widely used benchmarks tailored to memory-augmented agentic scenarios: LoCoMo [9] for long-term dialogue, PersonaMem [74] for memory personalization, and HotpotQA [75] and PopQA [76] for open-domain question answering. Adhering to the standard evaluation protocols, we employ *Accuracy* for PersonaMem, BLEU-1 (B1) and F1 for LoCoMo, and Exact Match (EM) and F1 for the others. More dataset details are included in Appendix C.

Baselines. We compare UniMem against a suite of representative baselines that collectively reveal the evolutionary trajectory of memory-augmented LLM agents: (*i*) Anchored in the retrieve-then-read paradigm, Vanilla RAG [22] directly concatenates the retrieved raw passages with input queries, while HippoRAG [41] refines the retrieval through a hippocampus-inspired graph index; (*ii*) Extending this paradigm, a substantial body of work develops dedicated memory architectures, exemplified by Mem0 [24], A-Mem [23], CAM [26], LightMem [77], and CompassMem [78], spanning tabular, graph, and hierarchical schemas, with retrieval strategies ranging from global similarity to multi-grained traversal; (*iii*) More recent advances including Mem1 [79], Mem-α [80], and Memory-R1 [81] further employ reinforcement learning techniques to optimize the triggering, construction, and retrieval of memories.

Table 1: Overall performance comparison of UniMem against representative baselines across four memory-augmented agentic benchmarks. The best results are in **bold**, and the second-best results are underlined. EM and F1 represent Exact Match and F1 scores, respectively, while B1 denotes BLEU-1.

Method	PersonaMem	LoCoMo		HotpotQA		PopQA	
	Accuracy	B1	F1	EM	F1	EM	F1
Vanilla RAG [22]	45.3	28.7	33.9	24.5	32.7	17.5	28.6
HippoRAG [41]	51.8$_{\uparrow 6.5}$	32.5$_{\uparrow 3.8}$	40.2$_{\uparrow 6.3}$	33.7$_{\uparrow 9.2}$	45.4$_{\uparrow 12.7}$	30.2$_{\uparrow 12.7}$	42.9$_{\uparrow 14.3}$
Mem0 [24]	43.2$_{\downarrow 2.1}$	38.2$_{\uparrow 9.5}$	45.4$_{\uparrow 11.5}$	23.1$_{\downarrow 1.4}$	34.0$_{\uparrow 1.3}$	19.7$_{\uparrow 2.2}$	27.6$_{\downarrow 1.0}$
A-Mem [23]	56.5$_{\uparrow 11.2}$	34.5$_{\uparrow 5.8}$	41.6$_{\uparrow 7.7}$	21.6$_{\downarrow 2.9}$	30.5$_{\downarrow 2.2}$	18.1$_{\uparrow 0.6}$	25.2$_{\downarrow 3.4}$
CAM [26]	59.7$_{\uparrow 14.4}$	39.4$_{\uparrow 10.7}$	46.3$_{\uparrow 12.4}$	40.8$_{\uparrow 16.3}$	53.2$_{\uparrow 20.5}$	37.8$_{\uparrow 20.3}$	46.5$_{\uparrow 17.9}$
LightMem [77]	44.8$_{\downarrow 0.5}$	38.2$_{\uparrow 9.5}$	45.5$_{\uparrow 11.6}$	27.4$_{\uparrow 2.9}$	38.6$_{\uparrow 5.9}$	21.2$_{\uparrow 3.7}$	30.3$_{\uparrow 1.7}$
CompassMem [78]	62.2$_{\uparrow 16.9}$	41.3$_{\uparrow 12.6}$	49.2$_{\uparrow 15.3}$	41.2$_{\uparrow 16.7}$	52.7$_{\uparrow 20.0}$	38.4$_{\uparrow 20.9}$	47.0$_{\uparrow 18.4}$
Mem1 [79]	57.3$_{\uparrow 12.0}$	25.5$_{\downarrow 3.2}$	29.7$_{\downarrow 4.2}$	43.7$_{\uparrow 19.2}$	55.3$_{\uparrow 22.6}$	39.7$_{\uparrow 22.2}$	47.3$_{\uparrow 18.7}$
Mem-α [80]	72.9$_{\uparrow 27.6}$	42.0$_{\uparrow 13.3}$	50.4$_{\uparrow 16.5}$	44.2$_{\uparrow 19.7}$	56.5$_{\uparrow 23.8}$	40.6$_{\uparrow 23.1}$	47.8$_{\uparrow 19.2}$
Memory-R1 [81]	69.6$_{\uparrow 24.3}$	36.8$_{\uparrow 8.1}$	43.5$_{\uparrow 9.6}$	45.5$_{\uparrow 21.0}$	58.8$_{\uparrow 26.1}$	45.2$_{\uparrow 27.7}$	51.3$_{\uparrow 22.7}$
UniMem (Ours)	**80.2**$_{\uparrow 34.9}$	**49.3**$_{\uparrow 20.6}$	**55.7**$_{\uparrow 21.8}$	**53.5**$_{\uparrow 29.0}$	**64.8**$_{\uparrow 32.1}$	**50.4**$_{\uparrow 32.9}$	**56.0**$_{\uparrow 27.4}$

Implementation Details. Unless otherwise specified, we adopt Qwen3-4B-Instruct [5] as the base LLM model throughout our main experiments, and additionally employ LLaMA3.1-8B-Instruct [4] to investigate the effect of distinct LLM backbones. To fully showcase the role of memory adaptation, we directly adopt the raw passage collection as the memory bank without any sophisticated memory construction. For the memory retrieval, we leverage BM25 [43] to retrieve the top-K most relevant documents per query by default. The length of latent representations is set to 16 by default, while we also examine the performance variations across alternative hyperparameter configurations. More implementation details, such as SFT and GRPO training configurations, are provided in Appendix D.

5.2 Main Results

Table 1 compares UniMem with a suite of representative memory-augmented baselines across four agentic benchmarks. UniMem achieves the highest performance across all seven evaluation metrics.

Consistent Performance Superiority. UniMem outperforms the strongest respective baselines on each benchmark by substantial margins: $+7.3$ Accuracy on PersonaMem, $+5.3$ F1 on LoCoMo, $+6.0$ F1 on HotpotQA, and $+4.7$ F1 on PopQA. Notably, these four benchmarks evaluate distinct memory capabilities, ranging from dialogue continuity and preference tracking to multi-hop reasoning and rare-entity recall. The persistent performance gains across such diverse workloads point to a fundamental bottleneck shared among memory-augmented tasks—precisely the structural limitation that our unified adaptation mechanism is designed to overcome.

Brittleness of the Textual Pathway. Baselines employing sophisticated memory architectures (e.g., CAM, CompassMem) or RL-driven optimization (e.g., Mem-α, Memory-R1) exhibit improvements over Vanilla RAG, yet these gains are often bounded and inconsistent. For instance, Mem1 underperforms Vanilla RAG on LoCoMo (B1: 25.5, -3.2) despite remaining competitive on HotpotQA, while Mem0 similarly trails Vanilla RAG on PersonaMem and PopQA. We attribute this brittleness to their reliance on the purely textual pathway, where memories enter only through the input prompt and inevitably trigger the challenges of context overload and semantic dilution.

The Value of Memory Adaptation. Even under the simplest memory configuration (raw passages coupled with sparse BM25 retrieval), UniMem surpasses all evaluated baselines, including those equipped with elaborate memory structures and RL strategies. In other words, a lightweight upstream, when paired with a profound downstream adaptation mechanism, consistently surpasses the pipelines that front-load complexity into memory construction and retrieval. This suggests that, for LLM agents, how to *harness* memories is at least as critical a design axis as how to *organize* memories.

5.3 Compatibility Analysis

Beyond the stand-alone evaluation, we further investigate the compatibility of UniMem with prevailing memory construction and retrieval mechanisms, aiming to examine whether the adaptation gains can generalize to more advanced upstream designs. Specifically, we integrate our UniMem with four

Table 2: Compatibility analysis of UniMem with representative memory mechanisms.

Method	Construction	Retrieval	PersonaMem	LoCoMo		HotpotQA	
			Accuracy	B1	F1	EM	F1
UniMem (Default)	Raw Passages	BM25	80.2	49.3	55.7	53.5	64.8
+ HippoRAG [41]	Graph	PPR	82.5	49.8	57.1	55.6	68.0
+ Mem0 [24]	Tabular	Dense	80.7	52.2	58.3	54.8	65.9
+ A-Mem [23]	Graph	Dense	83.2	50.5	56.8	54.4	66.2
+ CAM [26]	Hierarchical	Prune-and-Grow	85.6	52.7	59.0	56.3	68.5

representative baselines (Mem0, A-Mem, HippoRAG, and CAM), covering tabular, graph, and hierarchical memories, as well as retrieval strategies based on dense similarity and multi-grained traversal.

As reported in Table 2, all four pairings outperform both the default UniMem and their corresponding standalone counterparts on every benchmark. The integration not only adds $+1$ to $+5$ points over the default UniMem across most metrics, but also lifts each counterpart by substantially larger margins, e.g., UniMem+CAM exceeds CAM alone by $+12.7$ on LoCoMo F1 and $+25.9$ on PersonaMem. Furthermore, the relative ranking among the four methods is broadly preserved after the integration, suggesting that their distinctive merits are largely inherited rather than overshadowed by our paradigm.

These results confirm that the stages of construction, retrieval, and adaptation play synergistic roles in shaping agentic memory. On one hand, stronger upstream construction and retrieval mechanisms can reduce redundancy and noise within the memory fragments, empowering our latent compressor and parametric modulator with more salient materials to precisely capture task-relevant semantic signals. On the other hand, the unified adaptation principle of UniMem equips LLM agents with the capacity to effectively harness the upstream memory information through both latent and parametric pathways, thereby circumventing the dual limitations of context overload and semantic dilution inherent to the plain textual concatenation. In this sense, UniMem can be positioned as an orthogonal layer to be integrated with existing memory designs in a synergistic manner, such that the advances in upstream engineering and downstream adaptation can mutually reinforce rather than substitute for one another.

5.4 More Analysis: Configurations and Variants

Performance by Question Type. To verify the efficacy of UniMem from a fine-grained perspective, we report its performance on the four official question categories of LoCoMo, spanning local factual lookup (*Single-Hop*), multi-step composition (*Multi-Hop*), temporal integration (*Temporal*), and open-ended generation (*Open-Domain*). As shown in Table 3, UniMem outperforms the baselines across the four categories, with the margins widening on those that require deep memory engagement. On *Temporal* questions, where the model must connect events across multiple conversational turns, UniMem reaches 63.3 F1, exceeding the Vanilla RAG by 41.4 points; on *Open-Domain* questions, where answers are generated from the entire conversation history, UniMem lifts F1 from 10.8 to 38.7. Such performance patterns demonstrate the advantages of our unified adaptation design in harnessing memory signals: latent compression preserves the salience of cross-turn associative signals within the input context, while parametric modulation propagates their influence throughout the network layers. We also observe the same gain patterns across the seven question types of PersonaMem (Appendix E).

Efficiency. As shown in Figure 3, UniMem delivers superior performance across all the benchmarks, while incurring only marginal inference overhead relative to Vanilla RAG. This favorable trade-off is intrinsic to our architectural design: the *Compressor* entails merely a single forward pass through the frozen backbone, and the *Modulator* is implemented as a lightweight MLP that generates low-rank weight modulations; neither component adds appreciable computational cost beyond the base LLM. During the training phase, only the *Compressor* and the *Modulator* participate in optimization, together accounting for less than 8% of the LLM backbone whose parameter weights remain frozen throughout. As a result, the training pipeline of our UniMem also fits within a budget comparable to the baselines.

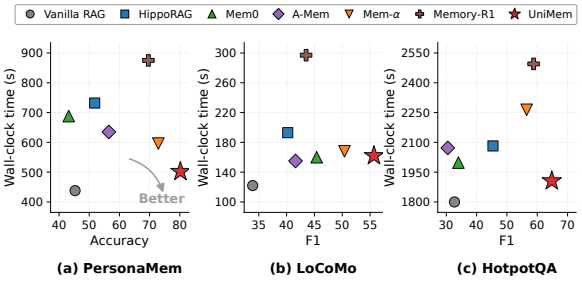

Figure 3: Efficiency-Performance analysis of UniMem.

Table 3: Performance across different question types on LoCoMo.

Method	Single-Hop		Multi-Hop		Temporal		Open-Domain		Overall	
	B1	F1	B1	F1	B1	F1	B1	F1	B1	F1
Vanilla RAG [22]	39.8	45.2	18.7	24.5	16.8	21.9	8.3	10.8	28.7	33.9
Mem0 [24]	41.9	49.4	27.1	36.2	43.0	48.9	25.5	29.3	38.2	45.4
A-Mem [23]	39.3	47.2	23.6	31.2	39.8	44.6	11.9	18.4	34.5	41.6
Mem-α [80]	47.4	55.1	27.5	37.4	46.5	55.5	26.9	35.2	42.0	50.4
Memory-R1 [81]	38.3	44.7	29.3	33.5	42.8	53.6	27.4	31.3	36.8	43.5
UniMem	**54.9**	**60.7**	**30.5**	**39.6**	**57.6**	**63.3**	**32.9**	**38.7**	**49.3**	**55.7**

Hyperparameter Sensitivity. We now analyze the sensitivity of UniMem to its three core hyperparameters: the latent length k, the hyper rank r, and the modulation strength λ. As shown in Figure 4, the response curves remain smooth across the three benchmarks: performance rises with longer latents and larger hyper ranks before saturating, while the modulation strength admits a relatively broad optimum near $\lambda = 1.0$. The most pronounced exception is LoCoMo, whose more compact per-conversation memory peaks at $k = 16$ and degrades as k grows further, in line with the intuition that latent capacity should match the underlying memory density [82].

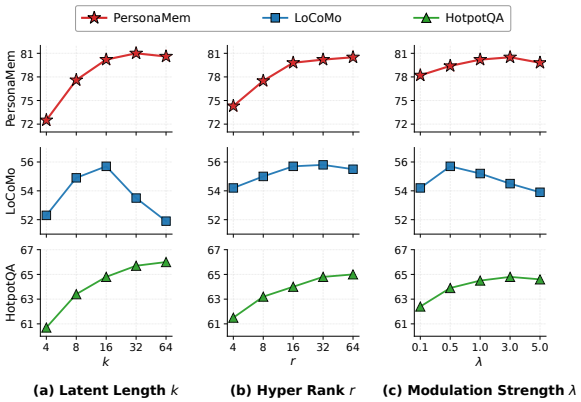

Figure 4: Hyperparameter sensitivity of UniMem.

Variants. Table 4 reports a variant analysis of our UniMem along two axes (i.e., backbone substitution and component ablation). For the LLM backbone, we replace Qwen3-4B-Instruct with LLaMA3.1-8B-Instruct, and this substitution preserves the overall performance level, with UniMem continuing to outperform the baselines in Table 1. This demonstrates that the advantages of UniMem primarily stem from our unified adaptation principle rather than the choice of backbone model. For the component ablation, removing either the latent or the parametric pathway incurs notable performance degradation, thus corroborating the efficacy of our multi-pathway memory adaptation framework. Moreover, ablating the GRPO training stage alone also leads to comparable performance decline, demonstrating that the RL-driven optimization can elicit the memory adaptation capacity beyond the reach of the static SFT objective.

Table 4: Variant analysis of UniMem (Accuracy for PersonaMem; B1/F1 for LoCoMo; EM/F1 for HotpotQA).

UniMem Variant	PersonaMem	LoCoMo	HotpotQA
w/ Qwen3-4B-Instruct	80.2	49.3/55.7	53.5/64.8
w/ LLaMA3.1-8B-Instruct	75.5	51.2/58.1	51.7/65.3
w/o Latent Pathway	70.9	44.2/50.6	47.5/60.2
w/o Parametric Pathway	77.6	46.5/53.8	50.4/61.3
w/o GRPO Training	74.2	43.7/52.4	48.2/60.5

Generalization to Experiential Memory. We further evaluate UniMem under the experiential-learning protocol [18] on HotpotQA and ALFWorld [83], where the memory bank is initialized empty and populated solely through the agent's training-time rollouts prior to evaluation on held-out queries. As reported in Appendix F, UniMem outperforms all baselines on both benchmarks by clear margins. For the agent-generated trajectories that are inherently noisier than curated factual passages, the latent compressor in our UniMem effectively suppresses low-utility memory signals, while the parametric modulator then propagates the salient remainder into the agentic network layers for robust utilization.

6 Conclusion

This paper recasts memory adaptation as the underexplored third stage of memory-augmented LLM agents, where the dominant textual pathway suffers from context overload and semantic dilution. We introduced **UniMem**, a unified framework that augments the textual pathway with a latent *Compressor* and a parametric *Modulator*, jointly relieving these two bottlenecks within a frozen LLM. Across four agentic benchmarks, UniMem consistently outperforms representative baselines while remaining composable with off-the-shelf memory mechanisms and incurring only marginal inference overhead. We hope this adaptation-centric perspective will offer valuable insights for agentic memory research.

Acknowledgments and Disclosure of Funding

This work is supported in part by the Outstanding Innovative Talents Cultivation Funded Programs 2025 of Renmin University of China. The authors declare no competing interests.

References

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901, 2020.

[2] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24:240:1–240:113, 2023.

[3] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

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

[5] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[6] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *arXiv preprint arXiv:2308.11432*, 2023.

[7] Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. The rise and potential of large language model based agents: A survey. *arXiv preprint arXiv:2309.07864*, 2023.

[8] Yuanchun Li, Hao Wen, Weijun Wang, Xiangyu Li, Yizhen Yuan, Guohong Liu, Jiacheng Liu, Wenxing Xu, Xiang Wang, Yi Sun, et al. Personal llm agents: Insights and survey about the capability, efficiency and security. *arXiv preprint arXiv:2401.05459*, 2024.

[9] Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of llm agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13851–13870, 2024.

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

[11] Runchu Tian, Yanghao Li, Yuepeng Fu, Siyang Deng, Qinyu Luo, Cheng Qian, Shuo Wang, Xin Cong, Zhong Zhang, Yesai Wu, Yankai Lin, Huadong Wang, and Xiaojiang Liu. Distance between relevant information pieces causes bias in long-context llms. *CoRR*, abs/2410.14641, 2024.

[12] Junhao Zheng, Chengming Shi, Xidi Cai, Qiuke Li, Duzhen Zhang, Chenxing Li, Dong Yu, and Qianli Ma. Lifelong learning of large language model based agents: A roadmap. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026.

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

[14] Yuyang Hu, Shichun Liu, Yanwei Yue, Guibin Zhang, Boyang Liu, Fangyi Zhu, Jiahang Lin, Honglin Guo, Shihan Dou, Zhiheng Xi, et al. Memory in the age of ai agents. *arXiv preprint arXiv:2512.13564*, 2025.

[15] Wei-Chieh Huang, Weizhi Zhang, Yueqing Liang, Yuanchen Bei, Yankai Chen, Tao Feng, Xinyu Pan, Zhen Tan, Yu Wang, Tianxin Wei, et al. Rethinking memory mechanisms of foundation agents in the second half. *arXiv preprint arXiv:2602.06052*, 2026.

[16] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 1–22, 2023.

[17] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik R Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[18] Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. *arXiv preprint arXiv:2308.10144*, 2023.

[19] Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In *Thirty-Eighth AAAI Conference on Artificial Intelligence*, pages 19724–19731, 2024.

[20] Zouying Cao, Jiaji Deng, Li Yu, Weikang Zhou, Zhaoyang Liu, Bolin Ding, and Hai Zhao. Remember me, refine me: A dynamic procedural memory framework for experience-driven agent evolution. *arXiv preprint arXiv:2512.10696*, 2025.

[21] Kuang-Huei Lee, Xinyun Chen, Hiroki Furuta, John Canny, and Ian Fischer. A human-inspired reading agent with gist memory of very long contexts. In *International Conference on Machine Learning*, pages 26396–26415. PMLR, 2024.

[22] Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, 2020.

[23] Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*, 2025.

[24] Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.

[25] Rui Li, Quanyu Dai, Zeyu Zhang, Xu Chen, Zhenhua Dong, and Ji-Rong Wen. Knowtrace: Bootstrapping iterative retrieval-augmented generation with structured knowledge tracing. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 1470–1480, 2025.

[26] Rui Li, Zeyu Zhang, Xiaohe Bo, Zihang Tian, Xu Chen, Quanyu Dai, Zhenhua Dong, and Ruiming Tang. CAM: A constructivist view of agentic memory for LLM-based reading comprehension. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[27] Boci Peng, Yun Zhu, Yongchao Liu, Xiaohe Bo, Haizhou Shi, Chuntao Hong, Yan Zhang, and Siliang Tang. Graph retrieval-augmented generation: A survey. *ACM Transactions on Information Systems*, 44(2):1–52, 2025.

[28] Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D Manning. Raptor: Recursive abstractive processing for tree-organized retrieval. In *The Twelfth International Conference on Learning Representations*, 2024.

[29] Charles Packer, Vivian Fang, Shishir G Patil, Kevin Lin, Sarah Wooders, and Joseph E Gonzalez. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

[30] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

[31] Minglai Yang, Ethan Huang, Liang Zhang, Mihai Surdeanu, William Yang Wang, and Liangming Pan. How is llm reasoning distracted by irrelevant context? an analysis using a controlled benchmark. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 13340–13358, 2025.

[32] Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H. Chi, Nathanael Schärli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, volume 202, pages 31210–31227, 2023.

[33] Yufeng Du, Minyang Tian, Srikanth Ronanki, Subendhu Rongali, Sravan Babu Bodapati, Aram Galstyan, Azton Wells, Roy Schwartz, Eliu A Huerta, and Hao Peng. Context length alone hurts LLM performance despite perfect retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 23281–23298, 2025.

[34] Brian Lester, Rami Al-Rfou, and Noah Constant. 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, 2021.

[35] Xiang Lisa Li and Percy Liang. 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, 2021.

[36] Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. In *International conference on machine learning*, pages 2793–2803, 2021.

[37] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.

[38] Tianle Li, Ge Zhang, Quy Duc Do, Xiang Yue, and Wenhu Chen. Long-context llms struggle with long in-context learning. *arXiv preprint arXiv:2404.02060*, 2024.

[39] Lei Liu, Xiaoyan Yang, Yue Shen, Binbin Hu, Zhiqiang Zhang, Jinjie Gu, and Guannan Zhang. Think-in-memory: Recalling and post-thinking enable llms with long-term memory. *arXiv preprint arXiv:2311.08719*, 2023.

[40] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*, 2024.

[41] Bernal Jimenez Gutierrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. HippoRAG: Neurobiologically inspired long-term memory for large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[42] Alireza Rezazadeh, Zichao Li, Wei Wei, and Yujia Bao. From isolated conversations to hierarchical schemas: Dynamic tree memory representation for LLMs. In *The Thirteenth International Conference on Learning Representations*, 2025.

[43] Stephen Robertson, Hugo Zaragoza, et al. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009.

[44] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[45] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 6769–6781, 2020.

[46] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*, 2022.

[47] Yucheng Li, Bo Dong, Frank Guerin, and Chenghua Lin. 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, 2023.

[48] Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 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, 2024.

[49] Fangyuan Xu, Weijia Shi, and Eunsol Choi. RECOMP: improving retrieval-augmented lms with context compression and selective augmentation. In *The Twelfth International Conference on Learning Representations*, 2024.

[50] Xinlei Yu, Zhangquan Chen, Yongbo He, Tianyu Fu, Cheng Yang, Chengming Xu, Yue Ma, Xiaobin Hu, Zhe Cao, Jie Xu, et al. The latent space: Foundation, evolution, mechanism, ability, and outlook. *arXiv preprint arXiv:2604.02029*, 2026.

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

[52] Muxin Fu, Xiangyuan Xue, Yafu Li, Zefeng He, Siyuan Huang, Xiaoye Qu, Yu Cheng, and Yang Yang. Latentmem: Customizing latent memory for multi-agent systems. *arXiv preprint arXiv:2602.03036*, 2026.

[53] Zeyu Zhang, Rui Li, Xiaoyan Zhao, Yang Zhang, Wenjie Wang, Xu Chen, and Tat-Seng Chua. Nextmem: Towards latent factual memory for llm-based agents. *arXiv preprint arXiv:2603.15634*, 2026.

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

[55] Jesse Mu, Xiang Li, and Noah Goodman. Learning to compress prompts with gist tokens. *Advances in Neural Information Processing Systems*, 36:19327–19352, 2023.

[56] Yujia Zheng, Zhuokai Zhao, Zijian Li, Yaqi Xie, Mingze Gao, Lizhu Zhang, and Kun Zhang. Thought communication in multiagent collaboration. *arXiv preprint arXiv:2510.20733*, 2025.

[57] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

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

[59] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115, 2023.

[60] Renxi Wang, Haonan Li, Xudong Han, Yixuan Zhang, and Timothy Baldwin. Learning from failure: Integrating negative examples when fine-tuning large language models as agents. *arXiv preprint arXiv:2402.11651*, 2024.

[61] Jian Guan, Wei Wu, Zujie Wen, Peng Xu, Hongning Wang, and Minlie Huang. Amor: A recipe for building adaptable modular knowledge agents through process feedback. *Advances in Neural Information Processing Systems*, 37:126118–126148, 2024.

[62] Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. Agenttuning: Enabling generalized agent abilities for llms. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3053–3077, 2024.

[63] Yaqing Wang, Sahaj Agarwal, Subhabrata Mukherjee, Xiaodong Liu, Jing Gao, Ahmed Hassan, and Jianfeng Gao. Adamix: Mixture-of-adaptations for parameter-efficient model tuning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5744–5760, 2022.

[64] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016.

[65] Jason Phang, Yi Mao, Pengcheng He, and Weizhu Chen. Hypertuning: Toward adapting large language models without back-propagation. In *International Conference on Machine Learning*, pages 27854–27875. pmlr, 2023.

[66] Yiming Zeng, Jinghan Cao, Zexin Li, Wanhao Yu, Zhankai Ye, Dawei Xiang, Ting Hua, Xin Liu, Shangqian Gao, and Tingting Yu. Hyperedit: Unlocking instruction-based text editing in llms via hypernetworks. *arXiv preprint arXiv:2512.12544*, 2025.

[67] Vinod Kumar Chauhan, Jiandong Zhou, Ping Lu, Soheila Molaei, and David A Clifton. A brief review of hypernetworks in deep learning. *Artificial Intelligence Review*, 57(9):250, 2024.

[68] Yuxin Jiang, Yufei Wang, Chuhan Wu, Wanjun Zhong, Xingshan Zeng, Jiahui Gao, Liangyou Li, Xin Jiang, Lifeng Shang, Ruiming Tang, et al. Learning to edit: Aligning llms with knowledge editing. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4689–4705, 2024.

[69] Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, et al. A comprehensive study of knowledge editing for large language models. *arXiv preprint arXiv:2401.01286*, 2024.

[70] Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, et al. Knowledge editing for large language models: A survey. *arXiv preprint arXiv:2310.16218*, 2023.

[71] Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.

[72] Rujikorn Charakorn, Edoardo Cetin, Yujin Tang, and Robert Tjarko Lange. Text-to-lora: Instant transformer adaption. *arXiv preprint arXiv:2506.06105*, 2025.

[73] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[74] Bowen Jiang, Zhuoqun Hao, Young Min Cho, Bryan Li, Yuan Yuan, Sihao Chen, Lyle Ungar, Camillo Jose Taylor, and Dan Roth. Know me, respond to me: Benchmarking LLMs for dynamic user profiling and personalized responses at scale. In *Second Conference on Language Modeling*, 2025.

[75] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, 2018.

[76] Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 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, 2023.

[77] Jizhan Fang, Xinle Deng, Haoming Xu, Ziyan Jiang, Yuqi Tang, Ziwen Xu, Shumin Deng, Yunzhi Yao, Mengru Wang, Shuofei Qiao, et al. Lightmem: Lightweight and efficient memory-augmented generation. *arXiv preprint arXiv:2510.18866*, 2025.

[78] Yuyang Hu, Jiongnan Liu, Jiejun Tan, Yutao Zhu, and Zhicheng Dou. Memory matters more: Event-centric memory as a logic map for agent searching and reasoning. *arXiv preprint arXiv:2601.04726*, 2026.

[79] Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Jinhua Zhao, Bryan Kian Hsiang Low, and Paul Pu Liang. Mem1: Learning to synergize memory and reasoning for efficient long-horizon agents. *arXiv preprint arXiv:2506.15841*, 2025.

[80] Yu Wang, Ryuichi Takanobu, Zhiqi Liang, Yuzhen Mao, Yuanzhe Hu, Julian McAuley, and Xiaojian Wu. Mem-α: Learning memory construction via reinforcement learning. *arXiv preprint arXiv:2509.25911*, 2025.

[81] Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Jinhe Bi, Kristian Kersting, Jeff Z Pan, et al. Memory-r1: Enhancing large language model agents to manage and utilize memories via reinforcement learning. *arXiv preprint arXiv:2508.19828*, 2025.

[82] Haoran Wei, Yaofeng Sun, and Yukun Li. Deepseek-ocr: Contexts optical compression. *arXiv preprint arXiv:2510.18234*, 2025.

[83] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020.

[84] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[85] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Advances in neural information processing systems*, volume 35, pages 27730–27744, 2022.

[86] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in neural information processing systems*, volume 36, pages 53728–53741, 2023.

[87] Jeffrey Cheng and Benjamin Van Durme. Compressed chain of thought: Efficient reasoning through dense representations. *arXiv preprint arXiv:2412.13171*, 2024.

[88] Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, Shuohang Wang, Yelong Shen, and Xin Eric Wang. Soft thinking: Unlocking the reasoning potential of llms in continuous concept space. *arXiv preprint arXiv:2505.15778*, 2025.

[89] Cong Jiang, Xiaofeng Zhang, Fangzhi Zhu, XiaoWei Chen, Junxiong Zhu, and Zheng Zhang. Rethinking LLM reasoning: From explicit trajectories to latent representations. In *The Fourteenth International Conference on Learning Representations*, 2026.

A Glossary of Symbols

Table 5 provides a glossary of commonly-used terms in this paper.

Table 5: Glossary of symbols used in this paper.

Symbol	Description
Inputs and Adaptation Operator.	
q	Input query specifying the agent's immediate task
$\mathcal{M}$	External memory bank
$\mathcal{D}$	Top-K retrieved subset, $\mathcal{D} = \{d_1, \ldots, d_K\}$
f_θ	Frozen inference LLM with parameters θ
$\mathcal{A}(q, \mathcal{D})$	Memory adaptation operator, $\big([q; \mathcal{D}], \mathcal{Z}, \Delta\theta\big)$ (Eq. 1)
$\mathcal{Z}$	Latent prompt injected alongside the input
$\Delta\theta$	Instance-specific weight modulation applied to selected layers of f_θ
UniMem Modules.	
$\mathcal{C}_\phi$	Compressor with LoRA adapter ϕ
Proj_ψ	Linear projection aligning H with the input embedding space of f_θ
$\mathcal{H}_\xi$	Modulator: hypernetwork generating per-(layer, type) LoRA factors
L	Learnable latent embeddings appended to the Compressor input, $L \in \mathbb{R}^{k \times h}$
L_c	Learnable latents reserved for the condition signal, $L_c \in \mathbb{R}^{k_c \times h}$
E	Learnable embedding dictionary of $\mathcal{H}_\xi$, indexed by a module type t or a layer index ℓ
Pathway Variables and Training.	
H	Latent representation matrix from $\mathcal{C}_\phi$, $H \in \mathbb{R}^{k \times h}$
k	Latent length, i.e., the number of vectors in H
c	Condition vector, hidden state of L_c from $\mathcal{C}_\phi$, $c \in \mathbb{R}^h$
$\Delta W_{\ell,t}$	Per-layer per-type weight perturbation, $\Delta W_{\ell,t} = \lambda A_{\ell,t} B_{\ell,t}^\top$ (Eq. 4)
r	Rank of the dynamic LoRA decomposition
λ	Scalar modulation strength, $\lambda > 0$
Φ	Trainable parameter set, $\Phi = \{\phi, \psi, \xi, L\}$
$\mathcal{T}^+$	Self-collected positive trajectory pool used for SFT warmup

B Discussion on More Related Works

We complement the condensed treatment of Section 2 by elaborating on two adjacent threads of research and clarifying the unique positioning of UniMem relative to them.

Reinforcement Learning for LLM Agents. Reinforcement learning has become a central tool for shaping LLM-agent behavior beyond what static supervision can achieve. Early efforts adopted PPO [84] with reward models distilled from human preferences [85], while subsequent direct preference optimization (DPO) [86] bypassed the reward model in favor of a contrastive objective over response pairs. Group-relative policy optimization (GRPO) [73], originally introduced for mathematical reasoning, has since become a widely adopted choice for agentic tasks because of its lightweight on-policy structure and stable convergence behavior. Within memory-augmented agents specifically, recent work treats memory operations themselves as a learned policy: Mem1 [79] optimizes the triggering of memory writes, Mem-α [80] learns the construction of structured entries, and Memory-R1 [81] jointly optimizes memory management and memory filtering. UniMem participates in this trend by adopting GRPO as the second stage of its training curriculum (Section 4.4), but the policy being shaped is qualitatively different. Rather than optimizing how memory is constructed or retrieved, the reward signal in UniMem sharpens how the latent and parametric pathways consume an already-retrieved memory subset, leaving upstream operators unchanged.

Implicit Reasoning with Latent Representations. A parallel line of research questions whether explicit, token-by-token reasoning trajectories are necessary at all. Coconut [71] introduced continuous chain-of-thought, in which the LLM reasons in a continuous latent space instead of decoding

Table 6: Principal training hyperparameters of UniMem.

Component	Hyperparameter	Value
Compressor $\mathcal{C}_\phi$	LoRA rank r_e	16
	LoRA α	32
	Target modules	$\{\text{q_proj}, \text{v_proj}\}$
Latents L	Latent length k	16
	Condition latents k_c	8
Modulator $\mathcal{H}_\xi$	Target modules $\mathcal{S}$	$\{\text{q_proj}, \text{v_proj}, \text{o_proj}\}$
	Dynamic-LoRA rank r	32
	Modulation strength λ	1.0
	Condition dimension d_c	128
SFT Stage	Optimizer	AdamW
	Learning rate	1×10^{-4} / 1×10^{-5}
	LR schedule	cosine, 10% warmup
	Batch size	32
GRPO Stage	Group size G	16
	Sampling	$T=1.0$, top-$p=0.95$
	Clip ratio ε	0.2
	KL coefficient β	0.04
	Learning rate	1×10^{-5} / 3×10^{-5}
	Batch size (queries/step)	16

verbose textual rationales. Subsequent work has extended this idea along several axes, including compressed chain-of-thought [87], soft-thinking variants that anneal between latent and token-level reasoning [88], and most recently latent reasoning tuning [89], where a lightweight reasoning network emits implicit vectors that condition the LLM during the answer pass. The shared theme across this family is that compact latent representations can subsume what was previously an expensive sequence of decoded reasoning tokens.

Positioning of UniMem. While UniMem and the methods above all operate in continuous latent space, they target distinct stages of the LLM computation graph. Implicit-reasoning latents replace the model's intermediate *thinking* between query reading and answer generation, whereas the latent prompt $\mathcal{Z}$ and the modulated weights $\theta + \Delta\theta$ in UniMem operate on the *input* side, replacing the verbose retrieved memory that would otherwise enter the network as plain text. A symmetric distinction holds for RL-driven memory operators: they shape upstream construction and retrieval policies, whereas UniMem keeps these stages untouched and re-designs the downstream adaptation step. The two families of methods are therefore stackable rather than competitive with UniMem: a stronger upstream memory policy supplies cleaner $\mathcal{D}$ for the Compressor to digest, while a stronger downstream reasoning model consumes the modulated $\theta + \Delta\theta$ exactly as it would consume any frozen LLM. Memory adaptation, in this sense, is a distinct frontier rather than a substitute for advances in RL-driven memory operations or implicit reasoning.

C Dataset Details

We evaluate UniMem on four memory-augmented benchmarks spanning open-domain question answering (HotpotQA, PopQA), long-term dialogue (LoCoMo), and memory personalization (PersonaMem). Each benchmark is divided into disjoint training, development, and test sets, with the development set used for checkpoint selection and all results reported on the test set. The fine-grained results in Tables 3 and 7 are computed per category on the test set.

LoCoMo Categories. LoCoMo [9] evaluates memory over long-term multi-session dialogues, and we adopt the four non-adversarial reasoning categories of its official taxonomy. *Single-hop* questions probe factual recall from a single utterance, and *Multi-hop* questions require synthesis across multiple utterances. *Temporal* questions involve reasoning over time-stamped events, while *Open-domain* questions combine the dialogue history with external commonsense.

Table 7: Performance breakdown by question type on PersonaMem (Accuracy). Column abbreviations: **Recall** = recalling user-shared facts; **Suggest** = suggesting new things; **Acknowledge** = acknowledging latest preferences; **Track** = tracking full preference evolution; **Revisit** = revisiting reasons behind preference updates; **Recommend** = providing preference-aligned recommendations; **Generalize** = generalizing to new scenarios.

Method	Recall	Suggest	Acknowledge	Track	Revisit	Recommend	Generalize	Overall
Vanilla RAG [22]	65.5	24.4	56.2	30.4	73.7	38.0	43.7	45.3
Mem0 [24]	46.2	24.4	42.0	50.3	79.1	51.6	33.7	43.2
A-Mem [23]	59.1	39.2	59.7	54.8	82.6	62.9	53.8	56.5
Mem-α [80]	82.5	51.7	82.1	70.6	83.2	76.3	75.8	72.9
Memory-R1 [81]	80.7	43.8	81.6	66.5	82.9	73.4	71.3	69.6
UniMem	**87.6**	**66.7**	**85.3**	**78.7**	**88.5**	**83.1**	**80.8**	**80.2**

PersonaMem Types. PersonaMem [74] builds on long-horizon multi-session conversations to probe seven facets of memory personalization that span fact-level recall, preference-state tracking, action-oriented suggestion, and forward-looking generalization. These facets include *Recall* (retrieving explicitly stated facts), *Suggest* (proposing novel items consistent with implicit preferences), *Acknowledge* (acknowledging the user's latest preferences), *Track* (tracking the full trajectory of preference change), *Revisit* (revisiting the reasons behind preference updates), *Recommend* (providing preference-aligned recommendations), and *Generalize* (transferring to previously unseen scenarios), tracing a continuum from low-level fact lookup to forward-looking inference.

D Implementation Details

Backbone and Hardware. Unless otherwise stated, the inference LLM f_θ is Qwen3-4B-Instruct [5], and the Compressor $\mathcal{C}_\phi$ shares the same backbone weights via a dedicated LoRA adapter. The variant analysis additionally uses LLaMA3.1-8B-Instruct [4] as an alternative backbone. All experiments are conducted on a single node equipped with $8\times$ NVIDIA H100 80 GB GPUs. Inference runs in BF16 with greedy decoding under a fixed generation budget.

UniMem Hyperparameters. Table 6 summarizes the principal hyperparameters of UniMem. The Compressor LoRA targets the attention projection matrices $\{\mathrm{q_proj}, \mathrm{v_proj}\}$ with rank $r_e = 16$ and scaling factor $\alpha = 32$. The Modulator follows the Text-to-LoRA-style architecture [72] with a four-block MLP body and per-type heads; LoRA factors are emitted for the target set $\mathcal{S} = \{\mathrm{q_proj}, \mathrm{v_proj}, \mathrm{o_proj}\}$ at every transformer layer, with rank $r = 32$ and modulation strength $\lambda = 1$. The default latent length is $k = 16$, with $k_c = 8$ latents reserved for the condition vector c.

Training Curriculum. We optimize Φ in two stages while keeping f_θ frozen. The first stage collects a pool of positive trajectories $\mathcal{T}^+$ by sampling rollouts from the underlying LLM and retaining those whose task metric (F1 or exact match) against the gold response exceeds a benchmark-specific threshold, and then runs SFT on $\mathcal{T}^+$ as a warmup. The second stage applies GRPO [73], treating the composition $f_\theta \circ \mathcal{A}(\cdot; \Phi)$ as the policy, the SFT-warmup checkpoint as the frozen reference policy, and the task metric as the reward. The principal hyperparameters of both stages are listed in Table 6.

Reproducibility. The empirical results reported in Tables 1–7 are averaged over three random seeds. Memory retrieval indices are pre-built and cached so that the randomness across seeds is concentrated in optimization rather than in the upstream pipeline.

E Fine-Grained Performance on PersonaMem

Table 7 reports the per-type breakdown of PersonaMem [74]. UniMem leads every baseline in all seven categories, with the gap widening on the types that demand richer integration of the persona history. The largest improvements over the strongest baseline Mem-α arise on *Suggest* ($+15.0$, novel suggestions grounded in implied preferences), *Track* ($+8.1$, multi-step preference evolution), and *Recommend* ($+6.8$, preference-aligned recommendations under shifting context); these are precisely the categories whose answers cannot be lifted from a single recent utterance and instead require synthesizing dispersed evidence across the persona corpus. Categories anchored to a localized fact,

such as *Recall* (+5.1) and *Revisit* (+5.3), see narrower yet still consistent gains, indicating that even a small relative improvement is informative once the textual pathway approaches its ceiling on simpler queries. The fine-grained pattern thus mirrors the LoCoMo analysis in Section 5.4: latent compression sharpens the signal-to-noise ratio of long persona transcripts, while parametric modulation propagates the salient evidence into the depth of f_θ where multi-step preference reasoning unfolds.

F Generalization to Experiential Memory

Table 8: Generalization to experiential memory for UniMem.

Method	HotpotQA (F1)	ALFWorld (SR)
Vanilla RAG [22]	29.4	43.5
Mem0 [24]	30.2	49.8
A-Mem [23]	33.5	47.0
Memory-R1 [81]	35.2	52.6
UniMem	**42.7**	**59.3**

Table 8 reports the full results of the experiential generalization experiment described in Section 5.4. The protocol follows the experiential-learning paradigm [18], where the memory bank is populated online from the agent's training-time rollouts before evaluation on held-out queries; the same pipeline is applied to both HotpotQA (open-domain QA) and ALFWorld [83] (embodied control). UniMem outperforms every baseline on both metrics, lifting HotpotQA F1 from 35.2 to 42.7 (+7.5) over the strongest experiential baseline Memory-R1 [81], and ALFWorld success rate from 52.6 to 59.3 (+6.7). The advantage is consistent with the design intuition behind UniMem: agent-generated trajectories are noisier and more idiosyncratic than curated passages, and our latent compressor effectively suppresses low-utility signals while the parametric modulator propagates the salient remainder into deeper network layers, thus enabling robust exploitation of noisy experiential memory.

G Discussion and Future Work

Where the Gains Come From. All main experiments pair UniMem with raw passages and BM25 retrieval, as in Vanilla RAG, such that its improvements do not rely on stronger memory construction or retrieval. The gains over Vanilla RAG are most pronounced on question types that require integrating dispersed or time-dependent evidence, such as Temporal questions on LoCoMo and preference tracking on PersonaMem, where the textual pathway is most affected by context overload and semantic dilution. Moreover, removing the latent pathway, the parametric pathway, or the GRPO stage consistently degrades performance (Table 4). These observations suggest that the decisive factor is how faithfully retrieved memory reaches the internal computation of the LLM, which is precisely the bottleneck that UniMem is designed to address.

Implications for Memory-Augmented Agents. Once the textual pathway becomes the limiting factor, further improvements in memory construction or retrieval alone yield diminishing returns, since downstream reasoning is constrained by how much memory enters the network and how deeply it propagates, neither of which the textual pathway can adjust. By making memory adaptation learnable and instance-conditioned, UniMem opens a design axis that is largely orthogonal to memory construction and retrieval. The compatibility analysis in Section 5.3 corroborates this orthogonality, as integrating UniMem with off-the-shelf memory mechanisms consistently improves upon both components. UniMem thus serves as a downstream layer whose gains accumulate with those of upstream memory designs. This adaptation stage is particularly valuable for noisy memories, as evidenced by the results on agent-generated trajectories in Appendix F.

Open Challenges and Future Directions. Two challenges remain open. First, the latent and parametric pathways are inherently opaque despite their expressive capacity, which makes it difficult to inspect what these pathways encode and how they interact with the textual pathway. Interpretability analyses tailored to memory adaptation, or hybrid representations that remain human-readable, are promising remedies. Second, although the adaptation principles of UniMem are agnostic to the modality and source of memory entries, all experiments in this paper involve textual memories. A

natural next step is to extend UniMem to interleaved multi-modal streams and to lifelong experience pools that the agent accumulates through tool usage or embodied interaction.

H Algorithms of UniMem

Algorithm 1 Inference pipeline of UniMem.

Require: Query q; memory bank $\mathcal{M}$; retriever $\mathrm{Retrieve}$; core parameters $\Phi = \{\phi, \psi, \xi, L\}$; frozen LLM f_θ; latent length k; target set $\mathcal{S}$.
Ensure: Response y.

$\mathcal{D} \leftarrow \mathrm{Retrieve}(q, \mathcal{M})$ ▷ Top-K memory subset
$H,\ c \leftarrow \mathcal{C}_\phi\big([q; \mathcal{D}],\ L,\ L_c\big)$ ▷ Single forward pass; $H \in \mathbb{R}^{k\times h}$ for $\mathcal{Z}$, $c \in \mathbb{R}^h$ for $\mathcal{H}_\xi$
$\mathcal{Z} \leftarrow \mathrm{Proj}_\psi(H) \in \mathbb{R}^{k\times h}$ ▷ Soft prompt
$\{\Delta W_{\ell,t}\}_{(\ell,t)\in\mathcal{S}} \leftarrow \mathcal{H}_\xi(c,\ E)$ ▷ Per-(layer, type) LoRA factors via Eq. (3)
$y \sim p_\theta\big(\cdot \mid [q; \mathcal{D}],\ \mathcal{Z},\ \theta + \Delta\theta\big)$ ▷ Generate with latent context and modulated weights
return y

Algorithm 2 End-to-end optimization procedure of UniMem.

Require: Frozen LLM f_θ; dataset $\mathcal{T} = \{(q, \mathcal{D}, y^\star)\}$; reward R; filter threshold τ; group size G.
Ensure: Trainable parameters $\Phi = \{\phi, \psi, \xi, L\}$.

function $\textsc{Adapt}(q, \mathcal{D}; \Phi)$ ▷ Adaptation operator $\mathcal{A}$, see Algorithm 1
  $H,\ c \leftarrow \mathcal{C}_\phi\big([q; \mathcal{D}],\ L,\ L_c\big)$; $\mathcal{Z} \leftarrow \mathrm{Proj}_\psi(H)$; $\Delta\theta \leftarrow \mathcal{H}_\xi(c,\ E)$
  return $\big([q; \mathcal{D}],\ \mathcal{Z},\ \theta + \Delta\theta\big)$
end function
Stage 1: Supervised warmup on self-collected positive trajectories.
$\mathcal{T}^+ \leftarrow \big\{(q, \mathcal{D}, \hat{y}) \ :\ (q, \mathcal{D}, y^\star) \in \mathcal{T},\ \hat{y} \sim p_\theta(\cdot \mid [q; \mathcal{D}]),\ R(\hat{y}, y^\star) \geq \tau\big\}$
Minimize $\mathcal{L}_{\mathrm{SFT}}(\Phi) = -\,\mathbb{E}_{(q,\mathcal{D},y)\sim\mathcal{T}^+} \log p_\theta\big(y \mid \textsc{Adapt}(q, \mathcal{D}; \Phi)\big)$ until convergence
Stage 2: Group-relative policy optimization on $\mathcal{T}$.
for $(q, \mathcal{D}, y^\star) \in \mathcal{T}$ **do**
  Sample a group $\{\hat{y}_i\}_{i=1}^G \sim p_\theta\big(\cdot \mid \textsc{Adapt}(q, \mathcal{D}; \Phi)\big)$ with rewards $r_i = R(\hat{y}_i, y^\star)$
  Group-relative advantages $A_i = (r_i - \bar{r})/\sigma_r$
  Update Φ via the GRPO objective with KL anchoring to the SFT reference policy
end for
return Φ

I Prompts

This section provides the prompt templates used throughout our experiments. We organize the templates into two groups. The first group (Compressor Input) describes how the verbose retrieved context is fed into $\mathcal{C}_\phi$ at the encoding stage. The second group (Inference Templates) describes how the inference LLM f_θ is queried at generation time, with one instantiation per benchmark.

Compressor Input

The Compressor consumes the concatenation of the query and the retrieved entries, prefixed with a brief instruction that orients its causal attention toward extracting query-relevant signals. Notably, the trailing k learnable latent representations L are appended at the embedding layer, and their hidden states after a single forward pass are read out as H. The memory descriptors in the template (e.g., reference documents) are adapted to each benchmark.

Compressor Input Template

```
You are an information extraction assistant.  Your task is to read the reference
documents below and extract key information that is most useful for accurately
answering the given question.
```

Input Question
⟨q⟩
Reference Documents
⟨d_1⟩
⟨d_2⟩
...
⟨d_K⟩
Using no more than k tokens, extract the most relevant information from the documents above that will help answer the question effectively.
[LATENTS] ⟨L_1⟩⟨L_2⟩ $\cdots$ ⟨L_k⟩

Inference Templates

At inference, the prompt fed to f_θ takes the same chat skeleton (system message + user turn + assistant turn opener), while the system message, per-document formatting, and post-question instruction vary across different benchmarks. The latent prompt $\mathcal{Z}$ is directly appended to the inference template. The corresponding templates for the three task families in our experiments are listed below.

HotpotQA / PopQA (Open-Domain QA)

[SYSTEM] You are a helpful assistant. Answer questions based on the provided documents.
[USER]
[1] Title: ⟨title⟩
Text: ⟨passage⟩
[2] Title: ⟨title⟩
Text: ⟨passage⟩
...
Question: ⟨q⟩
First write one or two sentences of reasoning about the relevant documents, then on a new line give the final answer as a short phrase in the form "Answer: ⟨answer⟩".

LoCoMo (Long-Term Dialogue)

[SYSTEM] You are a helpful assistant. Answer questions based on the provided documents.
[USER]
[1] Date: ⟨date⟩ | Turn: ⟨id⟩
Dialogue: ⟨speaker⟩ said, "⟨utterance⟩"
[2] Date: ⟨date⟩ | Turn: ⟨id⟩
Dialogue: ⟨speaker⟩ said, "⟨utterance⟩"
...
Question: ⟨q⟩
First write one or two sentences of reasoning about the relevant dialogue turns, then on a new line give the final answer as a short phrase in the form "Answer: ⟨answer⟩".

PersonaMem (Memory Personalization)

[SYSTEM] You are a helpful assistant with a long-term memory. Answer questions based on the provided conversation history.
[USER]
[1] Turn ⟨n_1⟩ (User): ⟨utterance⟩
[2] Turn ⟨n_2⟩ (Assistant): ⟨utterance⟩
...
⟨q⟩ with options (a)/(b)/(c)/(d)
First write one or two sentences of reasoning about the relevant conversation, then on a new line give the letter of the most appropriate model response in the form "Answer: (⟨letter⟩)".

I.1 Case Study

To complement the aggregate metrics with a qualitative view, we contrast Vanilla RAG (the textual pathway in isolation) with UniMem on two illustrative cases from LoCoMo and PersonaMem, where both systems share an identical retrieval set and prompt. The two cases highlight *relative-to-absolute time grounding* and *preference-aware memory acknowledgment*, two scenarios in which the combined burden of context overload and semantic dilution makes the textual pathway visibly brittle. In both cases, the textual pathway reaches the correct conclusion in its reasoning yet fails to reflect it in the final answer, whereas UniMem keeps its reasoning and answer consistent.

***Case 1.* Relative-to-absolute time grounding (LoCoMo Temporal, conv-41)**

Query: *When did John help renovate his hometown community center?*
Evidence turn (D28:11, 5 August 2023): "*. . . Last year, I helped renovate a rundown community center back home. . . .*"
Gold answer: 2022
Vanilla RAG: . . . Since the date of the dialogue is 5 August, 2023, and the event occurred "last year," it would have taken place in 2022. Answer: **Last year** *(incorrect)*
UniMem: . . . Since the date of the dialogue is 5 August, 2023, and it refers to "last year," the renovation must have taken place in 2022. Answer: **2022** *(EM with the gold answer)*
Analysis: The retrieved turn contains the relative expression "last year" together with the anchor date "5 August 2023" in its timestamp, and answering the query requires resolving the former against the latter. Vanilla RAG performs this resolution in its reasoning, yet its final answer reverts to the verbatim expression and omits the inferred year. UniMem arrives at the same intermediate conclusion and outputs the resolved year as its final answer. This mismatch between reasoning and answer is consistent with the semantic dilution of textual memories discussed in Section 3. UniMem is designed to counteract this effect by additionally conditioning the generation on the memory through the latent prompt and the memory-conditioned weights at every layer.

***Case 2.* Preference-aware memory acknowledgment (PersonaMem Acknowledge, persona 18)**

Query: *I once started writing regularly to explore my thoughts and emotions.*
Options:
(a) "*I remember you've mentioned enjoying drawing. . . .*"
(b) "*I remember you've mentioned finding journaling unproductive. . . .*"
(c) "*I don't think you ever shared this aspect with me. . . .*"
(d) "*I remember you've mentioned enjoying journaling. . . .*"
Evidence turn (Turn 557, User): "*Sure! I started a daily journaling practice back in early 2011 to explore my personal thoughts and emotions. . . .*"
Gold answer: (d)
Vanilla RAG: The user explicitly states in Turn 557 that they started a daily journaling practice in early 2011 . . . Therefore, the most accurate and supported response is one that acknowledges this shared experience. Answer: **(c)** *(incorrect)*
UniMem: The user explicitly mentions starting a daily journaling practice in early 2011 . . . This indicates a clear and consistent engagement with journaling as a personal and reflective practice. Answer: **(d)** *(matches the gold answer)*
Analysis: A correct reply must recognize that the user has already shared this habit and recall the user's positive attitude toward it, which rules out the wrong activity in (a), the reversed preference in (b), and the memory-agnostic reply in (c). Vanilla RAG cites the right turn and even concludes that the response should acknowledge the shared experience, yet it selects (c), which claims that the user never mentioned the habit. Its final choice therefore contradicts the evidence it cites, mirroring the mismatch between reasoning and answer observed in Case 1. UniMem instead bases its decision on the recalled preference and selects (d), the only option that both acknowledges the memory and reflects the user's positive attitude. This result again reflects the design of UniMem, in which the memory enters the inference LLM through the latent prompt and the layer-wise weight modulation in addition to the textual input.

J Compressor Design Analysis

To validate the Compressor instantiation in Section 4.2, we examine its two design axes, the compressor backbone and the compression mode, on LoCoMo. As shown in Table 9, autoregressive

Table 9: Analysis of different compressor designs on LoCoMo.

Actor backbone	Compressor backbone	Compression mode	B1	F1
Qwen3-4B-Instruct	Shared	Parallel	49.3	55.7
Qwen3-4B-Instruct	Shared	Autoregressive	49.2	56.3
Qwen3-4B-Instruct	Qwen3-14B	Parallel	50.4	58.5
Qwen3-4B-Instruct	Qwen3-14B	Autoregressive	50.8	59.2

encoding delivers only marginal gains over parallel encoding for both compressor backbones, while requiring one forward step for each latent. An independent Qwen3-14B compressor further improves performance but requires loading an additional LLM alongside the inference model. In light of these trade-offs, we adopt the shared compressor with parallel encoding as the default configuration.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the latent and parametric pathway contributions, all of which are supported by the experiments in Sections 5.2–5.4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Appendix G discusses the opacity of the latent and parametric pathways and the text-only modality scope of our experiments as open challenges.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not contain formal theorems or proofs; its contributions are an architectural framework and an empirical study.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 4 formalizes the architecture of UniMem, and Appendix D reports all hyperparameters, the SFT and GRPO training schedule, random seeds, and hardware.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: All four benchmarks (LoCoMo, PersonaMem, HotpotQA, PopQA) are publicly available and cited; the code repository link is provided in the abstract.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Section 5.1 reports benchmarks, baselines, backbones, and retrieval; Appendix D lists all hyperparameters together with the optimizer, learning-rate schedule, and batch sizes for both SFT and GRPO stages.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: The results are averaged over three random seeds, and UniMem outperforms the baselines on each benchmark by substantial margins.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix D reports the hardware (a single node with $8\times$ NVIDIA H100 80 GB GPUs) and the per-stage hyperparameters that determine the compute footprint.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The work uses only publicly released benchmarks and pretrained LLMs and complies with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This paper studies a general memory adaptation mechanism for LLM agents and is foundational research with no direct path to a specific deployed application. The potential downstream impacts inherit those of the underlying LLMs.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: We release the code base built on publicly available LLMs and do not introduce datasets or models with elevated misuse risk.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All benchmarks, baselines, and pretrained backbones are cited to their original sources and used in accordance with their respective public licenses.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The code base is available at the repository linked in the abstract, with documentation and configurations described in Appendix D.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This work involves neither crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This work involves neither crowdsourcing nor research with human subjects, and therefore no IRB review is required.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs appear in this work as the object of study (frozen backbones for memory adaptation), and are not used as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.