# Memory OS of AI Agent

**Jiazheng Kang**
Beijing University of Posts and Telecommunications
kjz@bupt.edu.cn

**Mingming Ji**
Tencent AI Lab
matthhewj@tencent.com

**Zhe Zhao**
Tencent AI Lab
nlpzhezhao@tencent.com

**Ting Bai** *
Beijing University of Posts and Telecommunications
baiting@bupt.edu.cn

## Abstract

Large Language Models (LLMs) face a crucial challenge from fixed context windows and inadequate memory management, leading to a severe shortage of long-term memory capabilities and limited personalization in the interactive experience with AI agents. To overcome this challenge, we innovatively propose a **Memory O**perating **S**ystem, i.e., **MemoryOS**, to achieve comprehensive and efficient memory management for AI agents. Inspired by the memory management principles in operating systems, MemoryOS designs a hierarchical storage architecture and consists of four key modules: Memory Storage, Updating, Retrieval, and Generation. Specifically, the architecture comprises three levels of storage units: short-term memory, mid-term memory, and long-term personal memory. Key operations within MemoryOS include dynamic updates between storage units: short-term to mid-term updates follow a dialogue-chain-based FIFO principle, while mid-term to long-term updates use a segmented page organization strategy. Extensive experiments on the LoCoMo benchmark show an average improvement of 49.11% on F1 and 46.18% on BLEU-1 over the baselines on GPT-4o-mini, showing contextual coherence and personalized memory retention in long conversations. The implementation code is open-sourced at https://github.com/BAI-LAB/MemoryOS.

## 1 Introduction

Large Language Models (LLMs) demonstrate impressive capabilities in text comprehension and generation, but face inherent limitations in sustaining dialogue coherence due to their reliance on fixed-length contextual windows for memory management. This fixed-length design inherently struggles to preserve continuity in dialogues with significant temporal gaps, often resulting in disjointed memory that manifests as factual inconsistencies and reduced personalization. Long-term memory coherence is critical in scenarios requiring persistent user adaptation, multi-session knowledge retention, or stable persona representation across extended interactions, where the limitations of fixed-length memory management in default LLMs become particularly acute, constituting a significant open challenge in the field.

To address this challenge, current memory mechanisms in default LLMs can be broadly categorized into three methodological types: (1) *Knowledge-organization* methods (Xu et al., 2025; Liu et al., 2023; Zhang et al., 2024a), such as A-Mem structure memory into interconnected semantic networks or notes to enable adaptive management and flexible retrieval; (2) *Retrieval mechanism-oriented* approaches (Huang et al., 2024; Zhong et al., 2024; Li et al., 2024; Chen et al., 2024), e.g., MemoryBank integrates semantic retrieval with a memory forgetting curve mechanism to allow long-term memory updating; and (3) *Architecture-driven* methods (Packer et al., 2023; Chhikara et al., 2025; Zhang et al., 2024b), such as MemGPT use hierarchical structures with explicit read and write operations to dynamically manage context. Although these diverse strategies typically operate in isolation, i.e., each focusing on single dimensions such as storage structure, retrieval mechanism, or update strategies, no unified operating system has been proposed to enable systematic and comprehensive memory management for AI agents.

Inspired by memory management principles in operating systems, we pioneer the proposal of a comprehensive memory operating system, termed **MemoryOS**. As illustrated in Fig. 1, MemoryOS comprises four core functional modules: memory *Storage, Updating, Retrieval, and Generation*. Through their coordinated collaboration, the system establishes a unified memory management

*Corresponding author.



*Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 25961–25970
November 4-9, 2025 

framework encompassing hierarchical storage, dynamic updating, adaptive retrieval, and contextual generation. Specifically, *Memory Storage* organizes information into short-term, mid-term, and long-term storage units. *Memory Updating* dynamically refreshes via a segmented paging architecture based on the dialogue-chain and heat-based mechanisms. *Memory Retrieval* leverages semantic segmentation to query these tiers, then *Response Generation* integrates retrieved memory information to generate coherent and personalized responses. This synergistic workflow ensures holistic management of long-term conversational memory, enabling contextual coherence and personalized recall in extended dialogues. The primary contributions of our work are summarized as:

- We make the first innovative attempt to introduce a systematic operating system, termed MemoryOS, for memory management, empowering AI agents with long-term conversational coherence and user persona persistence in long conversational interactions.
- MemoryOS introduces a pioneering three-tier hierarchical memory storage architecture and integrates four core functional modules (i.e, storage, updating, retrieval, and generation) for memory management, enabling dynamic capture and evolution of user preferences across extended dialogues.
- Comprehensive experiments validate MemoryOS's effectiveness and efficiency in maintaining response correctness and coherence across diverse benchmark datasets, demonstrating its capability to handle long conversational interactions.

## 2 Related Work

### 2.1 Memory for LLM Agents

Existing Large Language Models (LLMs) face fundamental challenges in handling complex scenarios requiring long-term coherence. These challenges stem from the inherent limitations of fixed-length designs, which struggle to maintain continuity in dialogues with significant temporal gaps, resulting in fragmented memory that manifests as factual inconsistencies and diminished personalization. Advancements in memory systems of LLMs addressing this problem can be broadly grouped into three categories: knowledge-organization, retrieval mechanism-oriented, and architecture-driven frameworks (Zhang et al., 2024c; Wu et al., 2025; Du et al., 2025). *Knowledge-organization* methods focus on capturing and structuring the intermediate reasoning states of large language models. For example, Think-in-Memory (TiM) (Liu et al., 2023) stores evolving chains-of-thought, enabling consistency through continual updates. A-Mem (Xu et al., 2025) organizes knowledge into an interconnected note network that spans sessions. Grounded Memory (Ocker et al., 2025) integrates vision-language models for perception, knowledge graphs for structured memory representation to enable context-aware reasoning in smart personal assistants. *Retrieval mechanism–oriented* approaches enrich the model with an external memory library. MemoryBank (Zhong et al., 2024) logs conversations, events, and user traits in a vector database and refreshes them using a forgetting-curve schedule; AI-town (Park et al., 2023) keeps memories in natural language and adds a reflection loop for relevance filtering. EmotionalRAG (Huang et al., 2024) retrieves memory entries by combining semantic similarity with the agent's current emotional state using a hybrid strategy. *Architecture-driven* designs alter the core control flow to manage context explicitly. For example, MemGPT (Packer et al., 2023) adopts an OS-like hierarchy with dedicated `read/write` calls, while Self-Controlled Memory (SCM) (Wang et al., 2025) introduces dual buffers and a memory controller that gates selective recall.

### 2.2 Memory Management in OS

Modern operating systems (OS) use combined segment-page memory management to balance logical structure with efficient physical utilization. Classic approaches like Multics (Bensoussan et al., 1972) organize memory into segments divided into pages, supporting efficient management, protection, and sharing. Segment metadata (size, access permissions) prevents external fragmentation (Bensoussan et al., 1972), while paging reduces internal fragmentation (Denning, 1970). Advanced OS use priority-based eviction (e.g., LRU, working-set models) to maintain hot data (Denning, 1970), and Zheng et al. (Zheng et al., 2020) show that combining coarse-grained segmentation with fine-grained paging minimizes overhead on many-core processors.

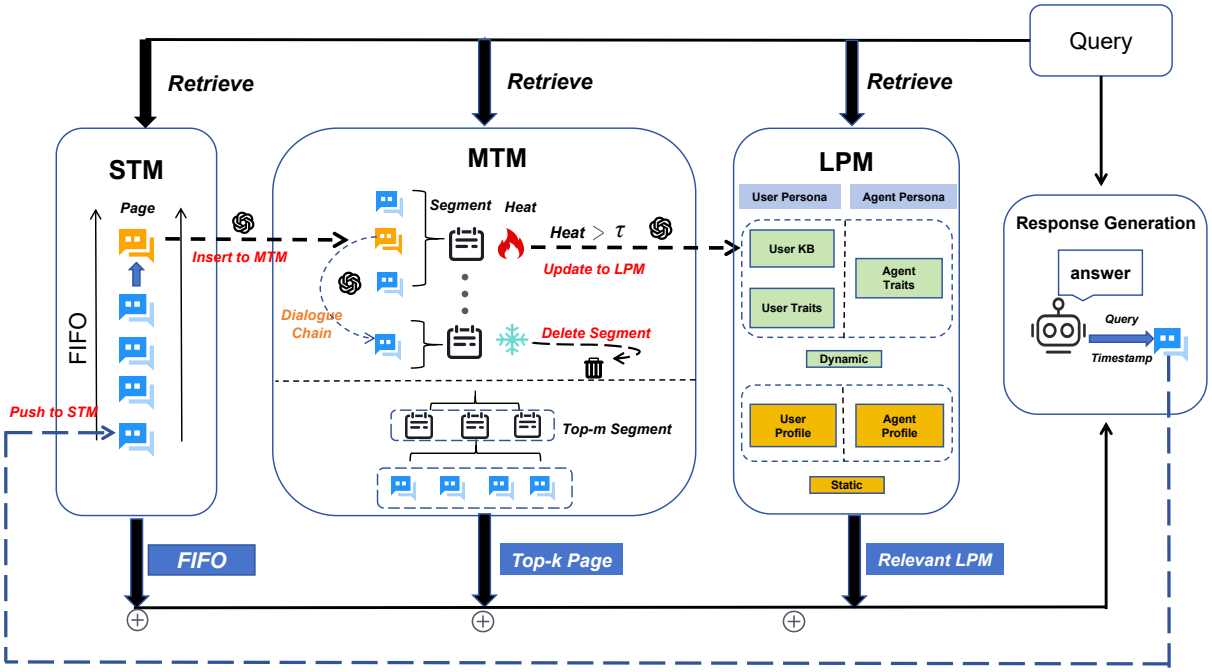



Figure 1: The overview architecture of MemoryOS, including memory Store, Updating, Retrieval, Response.

Inspired by the management in OS, our MemoryOS applies these principles by structuring its memory into logical segments (conversation topics) subdivided into pages. It uses heat-based prioritization to retain relevant content and efficiently discard or archive less-accessed information, enhancing context management and personalization.

## 3 MemoryOS

MemoryOS is a comprehensive memory management system for AI agents that dynamically updates memory and retrieves semantically relevant context, ensuring coherent and personalized interactions in long conversations.

### 3.1 Overview Architecture

The overview architecture of MemoryOS is illustrated in Fig. 1. It consists of four modules: **memory storage**, **update**, **retrieval**, and **generation**.

**Memory Storage:** This module is responsible for organizing and storing memory information by a three-tier hierarchical structure: Short-Term Memory (STM) for timely conversations, Mid-Term Memory (MTM) for recurring topic summaries, and Long-term Personal Memory (LPM) for user or agent preferences, ensuring memory integrity and effective utilization.

**Memory Updating:** This module manages dynamic memory refreshing, including STM-to-MTM updates via dialogue-chain FIFO and MTM-to-LPM updates using a segmented page strategy with heat-based replacement.

**Memory Retrieval:** This module retrieves relevant memory via specific queries, employing a two-tiered approach in MTM: semantic relevance identifies segments first, followed by retrieving pertinent dialogue pages. Finally, it combines persona attributes from LPM and contextual information from STM to generate responses, integrating all relevant memories for response generation.

**Response Generation:** It processes the data and generates appropriate responses. It integrates retrieval outcomes from STM, MTM, and LPM into a coherent prompt, enabling the generation of contextually coherent and personalized responses.

### 3.2 Memory Storage Module

Memory storage module is implemented via a hierarchical structure consisting of three type store units, i.e., Short-Term Memory (STM), Mid-Term Memory (MTM), and Long-term Personal Memory (LPM) store units.

**Short-Term Memory (STM):** It stores real-time conversation data in units called *dialogue pages*. Each dialogue page contains the user's query $Q$, the model's response $R$, and the timestamp $T$, structured as $page_i = \{Q_i, R_i, T_i\}$. To ensure contextual coherence, a dialogue chain is

constructed for each page to maintain contextual information in short-term continuous dialogue exchanges and ensure consistent context tracking. A **dialogue page** is defined as:

$$page_i^{chain} = \{Q_i, R_i, T_i, meta_i^{chain}\}, \quad (1)$$

where the meta information is generated by an LLM in two steps: first, evaluating a new page's contextual relevance to prior pages to determine chain linkage or resetting to the current page if semantically discontinuous; second, summarizing all chain pages into $meta_i^{chain}$.

**Mid-Term Memory (MTM):** Inspired by the memory management principles in Operating Systems, it adopts a *Segmented Paging* storage architecture. Dialogue pages with the same topic are grouped into segments, each containing multiple pages for a unique topic. The **segment** in MTM is defined as:

$$segment_i = \{page_i \mid \mathcal{F}_{\text{score}}(page_i, segment_i) > \theta\}, \quad (2)$$

where the content of a segment is summarized by a LLM based on the related dialogue pages. $\mathcal{F}_{\text{score}}$ measures the similarity between the dialogue page and the segment based on both semantic and keyword similarities, defined as:

$$\mathcal{F}_{\text{score}} = \cos(\mathbf{e}_s, \mathbf{e}_p) + \mathcal{F}_{Jaccard}(K_s, K_p), \quad (3)$$

where $\mathbf{e}_s$ and $\mathbf{e}_p$ denote the embedding vectors of a segment and the dialogue page, $K_s$ and $K_p$ are the keyword sets summarized by LLMs in the segment and page, respectively. $\mathcal{F}_{Jaccard}$ is the Jaccard similarity, defined as $\mathcal{F}_{Jaccard} = \frac{|K_s \cap K_p|}{|K_s \cup K_p|}$.

Pages with similarity scores to a segment exceeding the threshold $\theta$ are merged into the same segment, ensuring topic coherence and semantic consistency within the segment.

**Long-term Persona Memory (LPM):** This module ensures that both the user and the assistant maintain a persistent memory of important personal details and characteristics, ensuring consistency and personalization over long-term interactions. It consists of two components: the User Persona and the AI Agent Persona.

- **User Persona**. The **User Profile** comprises a static component with fixed attributes (gender, name, birth year), a User Knowledge Base (**User KB**) that dynamically stores factual information extracted and incrementally updated from past interactions, and **User Traits** that contains the evolving interests, habits, and preferences of users over time.
- **Agent Persona**. It contains the **Agent Profile**, which includes fixed settings like the role the AI agent assistant plays or its character traits, providing a consistent self-description. The **Agent Traits** are dynamic attributes that develop through interactions with the user, potentially including new settings added by the user or interaction history, e.g., recommended items, during conversations.

### 3.3 Memory Update Module

Key updating operations include updates within each unit itself and the update mechanism from STM to MTM, MTM to LPM store units.

**STM-MTM Update:** STM stores information in the form of dialogue pages in a queue with fixed length. We employ a First-In-First-Out (FIFO) update strategy for information migration to the Mid-Term Memory (MTM). New dialogue page is appended to the queue's end. When the STM queue reaches its maximum capacity, the oldest dialogue page is transferred from the STM to the MTM according to the FIFO principle.

**MTM-LPM Update:** MTM updates involve two operations, i.e., segment deletion and segment-to-LPM updates, both based on the Heat score of segments, defined as:

$$Heat = \alpha \cdot N_{\text{visit}} + \beta \cdot L_{\text{interaction}} + \gamma \cdot R_{\text{recency}}, \quad (4)$$

where coefficients $\alpha$, $\beta$, and $\gamma$ determine the relative importance of each factor. $N_{\text{visit}}$ is the number of times the segment has been retrieved, $L_{\text{interaction}}$ denotes the total number of dialogue pages within the segment, and $R_{\text{recency}}$ is the time decay coefficient represents the duration since the last retrieval time of the current segment, defined as: $R_{\text{recency}} = \exp\left(-\frac{\Delta t}{\mu}\right)$, where $\Delta t$ is the time elapsed since the last access, measured in seconds, and $\mu$ is a configurable time constant (i.e., 1e+7).

These three metrics, i.e., retrieval count ($N_{\text{visit}}$), total dialogue pages ($L_{\text{interaction}}$), and time decay coefficient ($R_{\text{recency}}$), collectively represent the frequent access, high engagement, and recent use as core indicators of segment heat. When the length of segments exceeds the maximum capacity, segments with the lowest heat are evicted.

This mechanism ensures that the content stored in MTM can retain topics with high engagement frequency in long user conversations, preserving detailed dialogue content under these topics through a segmented paging structure.

**LPM Update:** Segments with heat exceeding a threshold $\tau$ (i.e., 5) are transferred to LPM. Segments and their dialogue pages update User Traits, User KB, and Agent Traits. Following the user traits in (Li et al., 2025), we construct personalized User Traits with 90 dimensions across three categories: basic needs and personality, AI alignment dimensions, and content platform interest tags. Then, extract and update these dimensions from segments and dialogue pages to autonomously evolve traits by LLMs. Meanwhile, factual information relevant to the user and agent assistant is extracted and recorded into the User KB and Agent Traits, respectively. Both the User KB and Assistant Traits maintain a fixed-size queue (i.e., 100), employing a First-In-First-Out (FIFO) strategy. After memory transition, the number of pages $L_{interaction}$ in Eq. 4 is reset to zero, causing the heat score of the segment to decline. This ensures continuous persona evolution without redundancy.

### 3.4 Memory Retrieval Module

The Memory Retrieval Module retrieves information from three parts: STM for recent context, MTM using a two-stage retrieval (segment and page level), and LPM for personalized knowledge. Given a query from the user, the memory retrieval module retrieves from the stored memory, i.e., STM, MTM, and LPM, to return the most relevant information to generate the responses, defined as:

$$\mathcal{F}_{\text{Retrieval}}(STM, MTM, LPM|Q), \quad (5)$$

where $\mathcal{F}_{\text{Retrieval}}$ is the retrieval strategies applied in three memory store units.

**STM retrieval:** All dialogue pages are retrieved as STM holds the most recent contextual memory for the current conversation.

**MTM retrieval:** Inspired by psychological memory recall mechanisms (Yuan et al., 2024), a two-stage retrieval process is employed: first selecting segments via a matching score (defined in Eq. 3) to select top-m candidate segments, then selecting the top-k most relevant dialogue pages within these segments based on semantic similarity. After retrieval, the segment's visit counter $N_{\text{visit}}$ and recency factor $R_{\text{recency}}$ are updated.

**LPM retrieval:** The User KB and Assistant Traits each retrieve the top-10 entries with the highest semantic relevance to the query vector as background knowledge. All information in the User Profile, Agent Profile, and User Traits is utilized, as they store user preference information, agent characteristic information, and user-specific trait information.

### 3.5 Response Generation Module

Given the user query, the final prompt is constructed by integrating the above three types of retrieved content from STM, MTM and LPM, along with the user's query, form the final prompt input for the LLM to generate the final response. The incorporation of memory from recent dialogue (STM), relevant conversation pages (MTM), and persona information (LPM) ensures responses remain contextually coherent with current interactions, draw on historical dialogue details and summaries for depth, and align with user and assistant identities, respectively, enabling coherent, accurate, and personalized interaction experiences of AI agent systems.

## 4 Experiments

### 4.1 Experimental Settings

**Datasets.** We conduct our experiments on GVD (Zhong et al., 2024) and LoCoMo benchmark (Maharana et al., 2024) datasets. The GVD dataset consists of multi-turn dialogues simulated from interactions between 15 virtual users and an assistant over a 10-day period, covering at least two topics per day. The LoCoMo benchmark is specifically designed for assessing long-term conversational memory capabilities, consisting of ultra-long dialogues averaging 300 turns and about 9K tokens per conversation. Questions are categorized into four types: Single-hop, Multi-hop, Temporal, and Open-domain, to systematically evaluate the memory abilities of LLMs.

**Evaluation Metrics.** For the GVD dataset, we use three evaluation metrics: Memory Retrieval Accuracy (Acc.), Response Correctness (Corr.), and Contextual Coherence (Cohe.). Memory Retrieval Accuracy is evaluated as a binary indicator (0 or 1), while Correctness and Coherence are assessed on a three-point scale (0, 0.5, or 1). All

evaluations on the GVD dataset are automatically scored by the DeepSeek-R1 (Guo et al., 2025). On the LoCoMo benchmark, standard F1 and BLEU-1 (Papineni et al., 2002) are employed to evaluate the model's performance.

**Compared Methods.** We compare MemoryOS with representative memory methods, including:

**TiM (Think-in-Memory):** This approach mimics human memory by storing reasoning outcomes instead of raw dialogues. It uses locality-sensitive hashing (LSH) to retrieve relevant context before generating responses and updates memory through post-hoc reflection. TiM manages memory via insertion, forgetting, and merging to reduce redundant reasoning and improve consistency.

**MemoryBank (Zhong et al., 2024):** This framework dynamically adjusts memory strength based on the Ebbinghaus Forgetting Curve, prioritizing important content over time. It further builds a user portrait through continuous interaction analysis to support personalized responses.

**MemGPT (Packer et al., 2023):** This method introduces a dual-tier memory, featuring a main context for fast access and an external context for long-term storage. This design aims to enable scalable memory extension beyond the fixed context window of LLMs.

**A-Mem (Agentic Memory) (Xu et al., 2025):** it dynamically generates structured notes and links them to form interconnected knowledge networks, enabling continuous memory evolution and adaptive management for LLMs.

**MemoryOS:** It is a comprehensive memory management framework. Through coordinated collaboration with four core functional modules: memory Storage, Updating, Retrieval, and Generation. MemoryOS achieves dialogue coherence and user persona persistence in long interactions.

**Implementation Details.** The experiments are conducted on hardware equipped with 8-H20 GPUs. The fixed length of the dialogue page queue in STM is 7. The maximum length of segments in MTM is set to 2000. The maximum capacity for both the User KB and Agent Traits is set to 100 entries. The predefined $Heat$ threshold $\tau$, which controls the information from the MTM to the LPM, is set to 5. The values of $\alpha$, $\beta$, and $\gamma$ in Eq. 4 are equality set to 1. For memory retrieval, the number of retrieval top-$m$ segments was set to 5, and the hyperparameter top-$k$ for retrieved dialogue page was set to 5 and 10 on the GVD and LoCoMo datasets. The similarity value of $\theta$ in Eq. 2 is 0.6, and the time constant $\mu$ is 1e+7.

Table 1: Comparison results on the GVD dataset.

| Model | Method | Acc. ↑ | Corr. ↑ | Cohe.↑ |
|---|---|---|---|---|
| GPT-4o-mini | TiM | 84.5 | 78.8 | 90.8 |
| | MemoryBank | 78.4 | 73.3 | 91.2 |
| | MemGPT | 87.9 | 83.2 | 89.6 |
| | A-Mem | 90.4 | 86.5 | 91.4 |
| | **Ours** | **93.3** | **91.2** | **92.3** |
| Improvement (%) | | 3.2% ↑ | 5.4% ↑ | 1.0% ↑ |
| Qwen2.5-7B | TiM | 82.2 | 73.2 | 85.5 |
| | MemoryBank | 76.3 | 70.3 | 82.7 |
| | MemGPT | 85.1 | 80.2 | 86.9 |
| | A-Mem | 87.2 | 79.5 | 87.8 |
| | **Ours** | **91.8** | **82.3** | **90.5** |
| Improvement (%) | | 5.3% ↑ | 3.5% ↑ | 3.1% ↑ |

## 4.2 Main Results

The experimental results in GVD and LoCoMo benchmark datasets are shown in Table 1 and Table 2. We have the following observations:

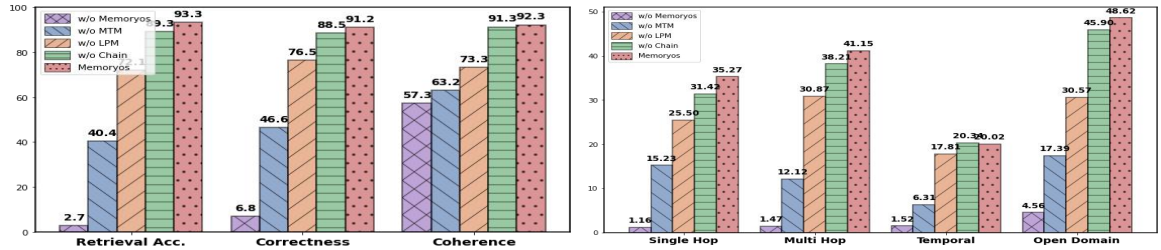
(1) Among all memory methods, MemoryBank performs the worst. This indicates that simply applying memory decay mechanisms is insufficient for managing conversational memory effectively. TiM outperforms MemoryBank by mitigating repetitive reasoning by saving "thoughts" rather than raw turns, but its single-stage hash retrieval cannot preserve cross-topic dependencies.

(2) A-Mem and MemGPT demonstrate relatively strong performance in long-form dialogue, But both of them lack systematic memory management mechanisms, giving rise to certain issues. For instance, MemGPT extends context via OS-style paging, yet its flat FIFO queue causes topic mixing as dialogue length grows; A-Mem organizes memories into a graph that enriches semantics, but the heavy, multi-step link generation inflates latency and error accumulation. By contrast, our Memory OS fuses a hierarchical STM/MTM/LPM architecture via segmented paging with heat-based eviction and a persona module, thereby ensuring that topic-aligned content remains accessible while maintaining consistency with users' specific preferences.

(3) Our proposed MemoryOS achieves superior performance across all benchmark datasets due to its hierarchical storage design, semantic retrieval

Table 2: LoCoMo dataset comparison with per-category scores and average ranks. A-Mem refers to the results reported in the original paper. A-Mem* represents our implementation results under the same experimental environment as our model.

| Model | Method | Single Hop | | Multi Hop | | Temporal | | Open Domain | | Avg. Rank ↓ | Avg. Rank ↓ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | F1 ↑ | BLEU-1 ↑ | F1 ↑ | BLEU-1 ↑ | F1 ↑ | BLEU-1 ↑ | F1 ↑ | BLEU-1 ↑ | (F1) | (BLEU-1) |
| GPT-4o-mini | TiM | 16.25 | 13.12 | 18.43 | 17.35 | 8.35 | 7.32 | 23.74 | 22.05 | 3.8 | 4.0 |
| | MemoryBank | 5.00 | 4.77 | 9.68 | 6.99 | 5.56 | 5.94 | 6.61 | 5.16 | 5.0 | 5.0 |
| | MemGPT | 26.65 | 17.72 | 25.52 | 19.44 | 9.15 | 7.44 | 41.04 | 34.34 | 2.2 | 2.5 |
| | A-Mem | 27.02 | 20.09 | 45.85 | 36.67 | 12.14 | 12.00 | 44.65 | 37.06 | – | – |
| | A-Mem* | 22.61 | 15.25 | 33.23 | 29.11 | 8.04 | 7.81 | 34.13 | 27.73 | 3.0 | 2.5 |
| | **Ours** | **35.27** | **25.22** | **41.15** | **30.76** | **20.02** | **16.52** | **48.62** | **42.99** | **1.0** | **1.0** |
| Improvement (%) | | 32.35%↑ | 42.33%↑ | 23.83%↑ | 5.67%↑ | 118.80%↑ | 111.52%↑ | 18.47%↑ | 25.19%↑ | – | – |
| Qwen2.5-3B | TiM | 4.37 | 5.01 | 2.54 | 3.21 | 6.20 | 5.37 | 6.35 | 7.34 | 4.3 | 3.5 |
| | MemoryBank | 3.60 | 3.39 | 1.72 | 1.97 | 6.63 | 6.58 | 4.11 | 3.32 | 4.8 | 4.8 |
| | MemGPT | 5.07 | 4.31 | 2.94 | 2.95 | 7.04 | 7.10 | 7.26 | 5.52 | 2.8 | 3.8 |
| | A-Mem | 12.57 | 9.01 | 27.59 | 25.07 | 7.12 | 7.28 | 17.23 | 13.12 | – | – |
| | A-Mem* | 10.31 | 8.76 | 16.31 | 11.07 | 6.94 | 7.31 | 12.34 | 10.62 | 2.3 | 2.0 |
| | **Ours** | **23.26** | **15.39** | **21.44** | **14.95** | **10.18** | **8.18** | **26.23** | **22.39** | **1.0** | **1.0** |
| Improvement (%) | | 125.61% ↑ | 75.68% ↑ | 31.45% ↑ | 35.05% ↑ | 46.69% ↑ | 11.90% ↑ | 112.56% ↑ | 110.83% ↑ | – | – |




(a)Ablation results on the GVD dataset with GPT-4o-mini

(b)Ablation results on the LoCoMo dataset with GPT-4o-mini using the F1 score.

Figure 2: The ablation study on the GVD and LoCoMo benchmark datasets.

Table 3: Efficiency analysis on LoCoMo benchmark (quantified by LLM call and recalled tokens counts).

| Method | Tokens | Avg.Calls | Avg. F1 |
|---|---|---|---|
| MemoryBank | 432 | 3.0 | 6.84 |
| TiM | 1,274 | 2.6 | 18.01 |
| MemGPT | 16,977 | 4.3 | 29.13 |
| A-Mem* | 2,712 | 13.0 | 26.55 |
| **Ours** | **3,874** | **4.9** | **36.23** |

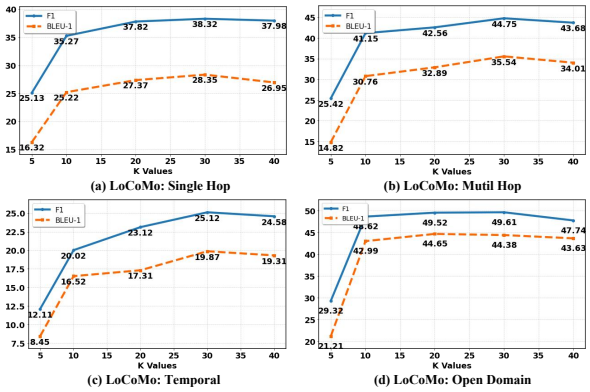



Figure 3: Impact of hyperparameter $k$ (retrieved pages in MTM) on LoCoMo benchmark.

capabilities, and persona-driven dynamic updating, which ensure coherent and accurate memory management. Notably, the model's advantages are particularly pronounced in more challenging memory management tasks. For example, on the LoCoMo benchmark with gpt-4o-mini, it achieves average improvements of 49.11% on F1 score and 46.18% on BLEU-1, while on the easier GVD dataset, in which all methods achieve higher baseline accuracy, our MemoryOS still surpasses the SOTA baseline A-Mem by 3.2% in accuracy, showing robust handling of complex long-context tasks requiring semantic consistency.

(4) To evaluate model efficiency, we employed two metrics: tokens consumed and average LLM calls in each response. As shown in Table 3, our method outperforms the Top-2 baselines (i.e., MemGPT and A-Mem) in both aspects, requiring significantly fewer LLM calls than A-Mem* (4.9 vs.13) and much lower token consumption than MemGPT (3,874 vs. 16,977).

### 4.3 Ablation Study

To assess the contribution of each core module in our framework, we perform an ablation study by individually removing three key components: the Mid-Term Memory (-MTM), the Long-term

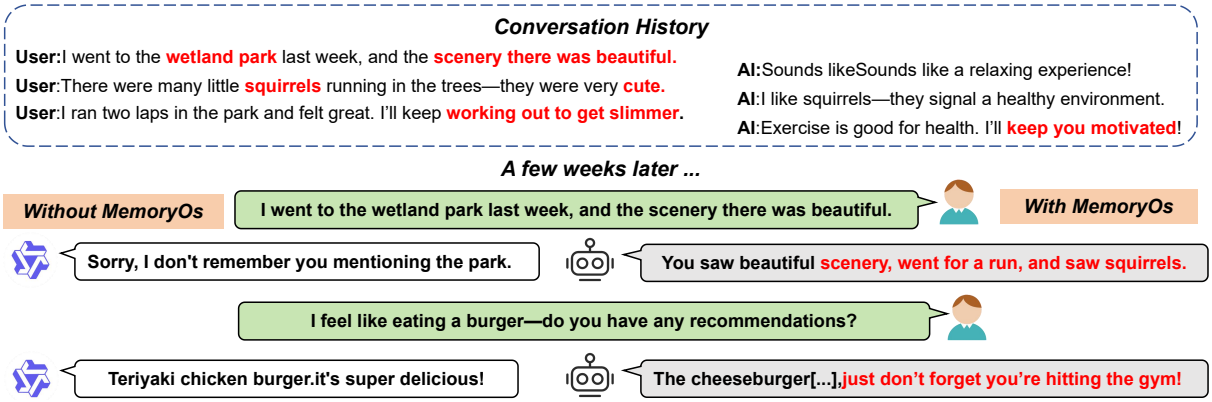



Figure 4: Case study demonstrating the positive impact of introducing our memory management system. Left: default LLMs; Right: with MemoryOS.

Persona Module (-LPM), and the Dialogue page Chain (-Chain) and the entire memory system (-MemoryOS). The results are presented in Figure 2. We can see that the memory system plays a pivotal role in the quality of responses during long dialogues. Without MemoryOS, the model's performance drastically reduces. In MemoryOS, the Mid-Term Memory (MTM) has the most significant impact, followed by the Long-Term Memory (LPM), while the Chain has the least impact.

### 4.4 Hyperparameter Analysis

We analyze the impact of top-k retrieved dialogue pages from Mid-Term Memory (MTM) on model performance. As shown in Fig. 3, by setting the hyperparameter $k$ with different values $k = \{5, 10, 20, 30, 40\}$ on the LoCoMo benchmark, we can see that the model's performance improves as $k$ increases, but the improvements diminish when exceeding a threshold. Retrieving more pages can enhance model performance, but excessive content may introduce noise, adversely affecting performance. We set $k = 10$ to achieve a relatively favorable performance while minimizing computational overhead.

### 4.5 Case Study

To visually demonstrate the role of our memory system, specifically how user long-term memory maintenance enhances conversational consistency. We present case studies in Fig. 4. Based on the conversation history, we show the responses of default LLMs and our LLMs with MemoryOS. We can see that MemoryOS exhibits excellent capabilities in recalling users' long-term conversations and preferences. For example, MemoryOS recalls details like "seeing the scenery, running, and spotting squirrels in the wetland park." from the initially mentioned a few weeks ago, "I went to the wetland park ...". These details are retrieved through the interplay and mutual support of mid-term memory's segment-page storage and the dialogue page chain. In addition, since the system integrates a personalization module, it can remember the user's goal of "wanting to get fit" and later proactively reminds the user when they express a desire to eat a burger: "Don't forget you want to get slimmer". This highlights the crucial role of the memory module in enhancing dialogue coherence and user experience.

## 5 Conclusion

Inspired by memory management mechanisms in operating systems, we pioneers propose a novel memory management system, MemoryOS, for AI agents. Implemented by a hierarchical memory storage architecture, MemoryOS addresses the fixed context window limitations in long conversations. By adapting OS-style segment-paging storage for dialogue history, MemoryOS enables efficient memory storage, updating, and semantic retrieval using heat-driven eviction to dynamically prioritize critical information across memory tiers. The integrated persona module captures evolving user preferences via personalized trait extraction, ensuring responses align with long conversation contexts. By bridging OS principles with AI memory management, MemoryOS empowers LLMs to sustain coherent, personalized conversations over extended interactions, enhancing human-like dialogue capabilities in real-world applications.

## 6 Limitation

While Memory OS represents a significant advancement in LLM-based agent memory management, several key limitations should be acknowledged. First, the current design relies on empirically set parameters for critical memory capacities, such as the short-term memory window size and the number of mid-term memory segments, without theoretical grounding in cognitive models. Future work will explore human memory mechanisms to derive more principled capacity configurations. Second, the topic within memory segments currently depends on the LLM's intrinsic extraction capabilities, lacking dynamic merging mechanisms to resolve redundancies arising from overlapping or evolving conversational themes. Future research may focus on developing adaptive merging strategies to refine segment organization and improve memory efficiency. Despite these limitations, we believe MemoryOS provides a fundamental foundation for memory management in AI agents, addressing the challenges of long-term dialogue coherence and personalization to enable more intelligent and adaptive memory systems for future AI agents.

## 7 Acknowledgments

This work is sponsored by the CCF-Tencent RAGR20240108, the National Natural Science Foundation of China under Grant No. 62236003, and the Tencent Basic Platform Technology Rhino-Bird Focused Research Program.

## References

André Bensoussan, C. T. Clingen, and R. C. Daley. 1972. The multics virtual memory: Concepts and design. *Communications of the ACM*, 15(5):308–318.

Weijie Chen, Ting Bai, Jinbo Su, Jian Luan, Wei Liu, and Chuan Shi. 2024. Kg-retriever: Efficient knowledge indexing for retrieval-augmented large language models. *arXiv preprint arXiv:2412.05547*.

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

Peter J. Denning. 1970. Virtual memory. *ACM Computing Surveys*, 2(3):153–189.

Yiming Du, Wenyu Huang, Danna Zheng, Zhaowei Wang, Sebastien Montella, Mirella Lapata, Kam-Fai Wong, and Jeff Z Pan. 2025. Rethinking memory in ai: Taxonomy, operations, topics, and future directions. *arXiv preprint arXiv:2505.00675*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Le Huang, Hengzhi Lan, Zijun Sun, Chuan Shi, and Ting Bai. 2024. Emotional rag: Enhancing role-playing agents through emotional retrieval. In *2024 IEEE International Conference on Knowledge Graph (ICKG)*, pages 120–127.

Hao Li, Chenghao Yang, An Zhang, Yang Deng, Xiang Wang, and Tat-Seng Chua. 2024. Hello again! llm-powered personalized agent for long-term dialogue. *arXiv preprint arXiv:2406.05925*.

Jia-Nan Li, Jian Guan, Songhao Wu, Wei Wu, and Rui Yan. 2025. From 1,000,000 users to every user: Scaling up personalized preference for user-level alignment. *Preprint*, arXiv:2503.15463.

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

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. 2024. 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, Bangkok, Thailand. Association for Computational Linguistics.

Felix Ocker, Jörg Deigmöller, Pavel Smirnov, and Julian Eggert. 2025. A grounded memory system for smart personal assistants. *Preprint*, arXiv:2505.06328.

Charles Packer, Vivian Fang, Shishir_G Patil, Kevin Lin, Sarah Wooders, and Joseph_E Gonzalez. 2023. Memgpt: Towards llms as operating systems.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22.

Bing Wang, Xinnian Liang, Jian Yang, Hui Huang, Shuangzhi Wu, Peihao Wu, Lu Lu, Zejun Ma, and Zhoujun Li. 2025. Scm: Enhancing large language model with self-controlled memory framework. *Preprint*, arXiv:2304.13343.

Yaxiong Wu, Sheng Liang, Chen Zhang, Yichao Wang, Yongyue Zhang, Huifeng Guo, Ruiming Tang, and Yong Liu. 2025. From human memory to ai memory: A survey on memory mechanisms in the era of llms. *arXiv preprint arXiv:2504.15965*.

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

Peiwen Yuan, Xinglin Wang, Shaoxiong Feng, Boyuan Pan, Yiwei Li, Heda Wang, Xupeng Miao, and Kan Li. 2024. Generative dense retrieval: Memory can be a burden. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, St. Julian's, Malta. Association for Computational Linguistics.

Kai Zhang, Yangyang Kang, Fubang Zhao, and Xiaozhong Liu. 2024a. LLM-based medical assistant personalization with short- and long-term memory coordination. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2386–2398, Mexico City, Mexico. Association for Computational Linguistics.

Kai Zhang, Yejin Kim, and Xiaozhong Liu. 2024b. Personalized llm response generation with parameterized memory injection. *arXiv preprint arXiv:2404.03565*.

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

Yan Zheng, Tong Zou, and Xingyan Wang. 2020. Segment-page-combined memory management technology based on a homegrown many-core processor. *CCF Transactions on High Performance Computing*, 2(4):376–381.

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