FOCUS: MANAGING MEMORY RETRIEVAL SPACE WITH REVERSIBLE FORGETTING IN LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Long term memory enables LLM agents to retain information across interactions and reuse it in later responses. As interactions accumulate, memory stores grow, increasing the amount of information available for retrieval. However, the expanding retrieval space allows weakly related memories to compete with relevant evidence, which makes it harder to identify the information needed for each query. Removing memories can reduce this retrieval space, yet memories that have not been useful for an extended period may become relevant again in the future. To address this problem, we introduce FOCUS, a long term memory framework that uses a reversible memory lifecycle to control retrieval space growth. FOCUS assigns an activation value to each memory and organizes stored memories into an Active Set and a Dormant Set, with activation governing transitions between the two sets. Active memories participate directly in query matching, while Dormant memories remain stored in groups whose summaries are matched to the query to select relevant groups and guide retrieval of their full memory entries. Extensive experiments on LoCoMo and Memora with Qwen3-30B-A3B-Instruct and DeepSeek-V4-Flash show that FOCUS significantly improves long term memory performance over competing methods while maintaining the smallest retrieval space. Notably, in the longest Memora setting, FOCUS maintains a small retrieval space that is only 18.28% of all stored memories after 1,750 sessions.

1 INTRODUCTION

LLM agents are increasingly used for complex tasks across a range of domains such as software engineering, scientific research, and personal assistance (Yang et al., 2024; Lu et al., 2026; Boiko et al., 2023; Zhang et al., 2025). In many applications, agent interactions span multiple sessions, requiring continued access to information from earlier sessions to preserve continuity (Li et al., 2025). As these sessions accumulate, the resulting history can exceed the LLM context window (Liu et al., 2024; Wang et al., 2023; Packer et al., 2024). Memory systems address this limitation by retaining information from earlier interactions and retrieving relevant memories when needed (Zhong et al., 2024; Wu et al., 2025). Over time, this process produces a growing memory store, motivating research into how stored information is organized and maintained.

Recent work has improved long term memory systems through advances in memory construction, updating, organization, and retrieval. Some approaches continually refine stored information as new interactions arrive, while others introduce associative or hierarchical structures to organize memories and support later access (Chhikara et al., 2025; Xu et al., 2025; Kang et al., 2025). Efficiency has also become an important consideration, with methods using buffered processing or compact memory representations to reduce the costs of memory construction and maintenance (Fang et al., 2026; Liu et al., 2025). These advances address several stages of the memory pipeline. However, existing methods give limited attention to managing retrieval space growth as memories accumulate over long interaction histories. As illustrated in Figure 1(a), accumulated memories expand the retrieval space, requiring relevant information to be selected at each query from an increasingly large candidate set that contains many weakly related memories. A straightforward way to limit this growth is to discard memories that appear less relevant to current interactions. However, information irrelevant to current queries may become useful again in later interactions. This tension calls for long term memory systems that can control the retrieval space while allowing stored memories to reenter retrieval when they become relevant in later interactions.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

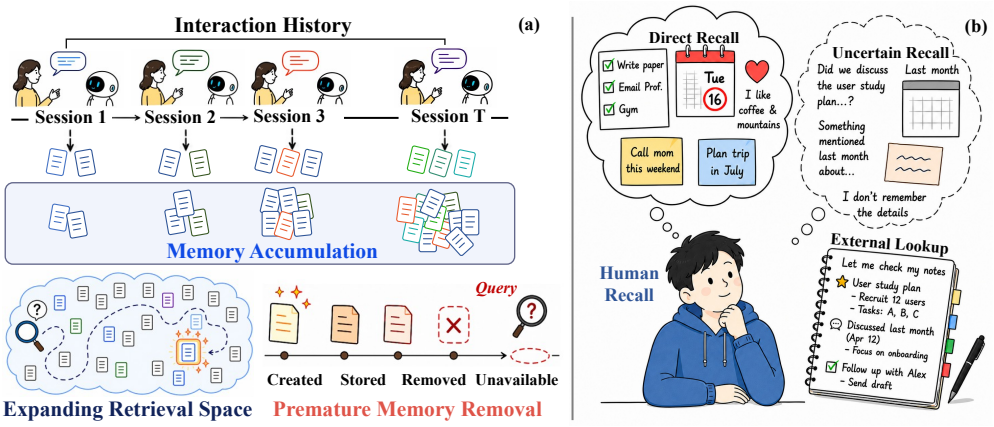


Figure 1: Motivation for reversible forgetting. (a) As interaction histories grow, accumulated memories enlarge the retrieval space. Permanent removal can reduce this space but may make previously stored information unavailable to future queries. (b) Readily accessible information can be recalled directly, while partial recollection can guide external lookup to recover missing details.

The way people forget and later recover information offers a useful perspective on this problem. As illustrated in Figure 1(b), recently or frequently revisited memories are often easier to retrieve, while memories that go unrecalled for longer periods may become less distinct over time (Bjork & Bjork, 1992; Wixted, 2004). Even when details become difficult to recall, partial recollection can still guide recovery through external records. For example, someone may remember that a decision was made in an earlier meeting and refer to the meeting notes to recover its details.

Motivated by this principle, we introduce FOCUS, the first long term memory management framework to **control retrieval space growth** through **reversible memory state transitions** while preserving access to memories that may become relevant again. FOCUS organizes stored memories into an Active Set and a Dormant Set, with an activation score governing transitions between them. Active memories participate directly in retrieval, while Dormant memories remain stored in groups whose summaries guide retrieval of their full entries. Retrieved memories from both sets are then ranked together for response generation. By matching group summaries, FOCUS keeps individual Dormant entries outside the main retrieval space while retaining their full content. After generating a response, FOCUS uses memory use attribution to identify which retrieved memories contributed to it. Since actual use provides stronger evidence of continued relevance than retrieval alone, FOCUS refreshes activation for used memories and provides temporary activation for those retrieved but unused. With activation also decaying naturally over time, an Active memory moves to the Dormant Set when its score falls below the forgetting threshold, while a Dormant memory returns to the Active Set upon reaching the reactivation threshold and becomes directly accessible again.

We evaluate FOCUS on LoCoMo (Maharana et al., 2024) and Memora (Uddin et al., 2026), using Qwen3-30B-A3B-Instruct (Yang et al., 2025) and DeepSeek-V4-Flash (DeepSeek-AI et al., 2026) as backbones. On LoCoMo, FOCUS achieves overall accuracies of 75.97 and 77.86, outperforming the best baseline by 5.84 and 2.28 points, respectively. On Memora, it achieves further gains of 5.19 and 6.53 points in overall Forgetting Aware Memory Accuracy (FAMA), which evaluates valid information retention and penalizes obsolete information. Ablation results further demonstrate the contribution of key components, with consistent performance declines on both benchmarks when they are removed. Beyond these performance gains, FOCUS maintains a substantially smaller retrieval space across all settings. In the longest setting, Memora Quarterly, FOCUS uses only 3.16K and 2.99K retrieval units, compared with 14.06K and 9.11K for LightMem. After 1,750 sessions, its retrieval space accounts for just 18.28% of all stored memory entries.

In summary, our main contributions are as follows. **(1)** We introduce FOCUS, the first long term memory framework to manage retrieval space through reversible memory state transitions while preserving complete memory content. **(2)** We develop an activation mechanism driven by time and recent memory use, with memory use attribution identifying whether retrieved memories contribute to the agent's response during each interaction. **(3)** We conduct extensive evaluations on LoCoMo and Memora with two backbone models, demonstrating consistent improvements in long term memory performance while achieving the smallest retrieval space across all tested settings.

2 PRELIMINARIES

2.1 MEMORY SYSTEMS FOR LLM AGENTS

We consider an LLM agent that interacts with a user over multiple sessions. Before interaction i, the completed interactions form the history $\mathcal{H}_i = \big((q_1, y_1), \ldots, (q_{i-1}, y_{i-1})\big)$. A long term memory system retains selected information from this history for later interactions through writing, management, and reading operations (Zhang et al., 2025), denoted by $\mathcal{W}$, $\mathcal{M}$, and $\mathcal{R}$, respectively.

The memory store evolves as the interaction history grows. Given a segment of the history, $\mathcal{W}$ extracts information to retain as memory entries, which are then incorporated into the existing store by $\mathcal{M}$ through addition, revision, or reorganization. We denote the resulting store available before interaction i by Mem_i. For a new query q_i, $\mathcal{R}(q_i, \mathrm{Mem}_i)$ selects relevant memory content and provides it to the LLM together with the query. The LLM then generates the response $y_i = \mathrm{LLM}\big(q_i, \mathcal{R}(q_i, \mathrm{Mem}_i)\big)$. Once the interaction is completed, (q_i, y_i) is appended to the history to form $\mathcal{H}_{i+1}$ and may subsequently be incorporated into the memory store through $\mathcal{W}$ and $\mathcal{M}$.

2.2 CHALLENGES OF LONG TERM MEMORY GROWTH

As interactions accumulate, $\mathcal{W}$ continually adds new memories, causing $|\mathrm{Mem}_i|$ to grow over time. Consequently, $\mathcal{M}$ must resolve new information against more stored memories, while $\mathcal{R}$ must select relevant content at each query from a larger candidate pool in which older or weakly related memories compete with useful evidence. Discarding memories that appear less relevant can ease these problems, but memories unused for long periods may become relevant again and cannot be recovered once deleted. This raises a central question: Can a long term memory system preserve stored information while dynamically controlling which stored memories participate in $\mathcal{R}$?

3 FOCUS: MEMORY MANAGEMENT WITH REVERSIBLE FORGETTING

As illustrated in Figure 2(a), FOCUS manages long term memory through three operations. The writing operation $\mathcal{W}_F$ extracts memory entries from interaction histories, and the management operation $\mathcal{M}_F$ incorporates them into the memory store (Section 3.1). Inspired by half-life models of human recall (Settles & Meeder, 2016), $\mathcal{M}_F$ assigns each memory an activation value and organizes stored memories into an Active Set and a Dormant Set, with activation governing reversible transitions between the two sets (Section 3.2). The reading operation $\mathcal{R}_F$ retrieves Active memories as full entries, whereas Dormant memories are organized into groups represented by compact summaries that guide access to relevant full memories for subsequent response generation (Section 3.3).

3.1 MEMORY CONSTRUCTION

FOCUS adopts a topic based buffering strategy following LightMem (Fang et al., 2026). Incoming dialogue turns first accumulate in a temporary buffer. When the buffer reaches capacity, FOCUS identifies topic boundaries using attention patterns and semantic similarity between adjacent turns, partitioning the buffered dialogue into topic segments T_i. The resulting segments are transferred to a construction buffer $\mathcal{B} = \{T_1, \ldots, T_N\}$ and accumulated for memory extraction. Once $\mathcal{B}$ is full, its contents are passed to an LLM for extraction in a single batch. Grouping related turns before extraction reduces the number of LLM calls compared with processing each turn separately and limits the mixing of information across topics during memory construction. Further details of the topic segmentation and buffering procedure are provided in Appendix A.1.

For each extraction, $\mathcal{W}_F$ maps the topic segments in $\mathcal{B}$ to a set of memory entries $\mathcal{N}$, each representing an atomic fact. Before incorporating a new fact into the memory store, FOCUS retrieves relevant existing memories from both the Active Set and the Dormant Set using the method described in Section 3.3. Based on a comparison between the incoming fact and these retrieved memories, the LLM selects one of four operations, `ADD`, `UPDATE`, `REWRITE`, or `IGNORE`. `ADD` creates a new memory when the incoming fact provides distinct information that can coexist with existing memories. `UPDATE` merges the new fact with an existing memory when one contains information that extends the other, producing a revised entry that preserves both. `REWRITE` handles apparent conflicts arising from implicit temporal or contextual conditions by revising the new fact and the existing memory to

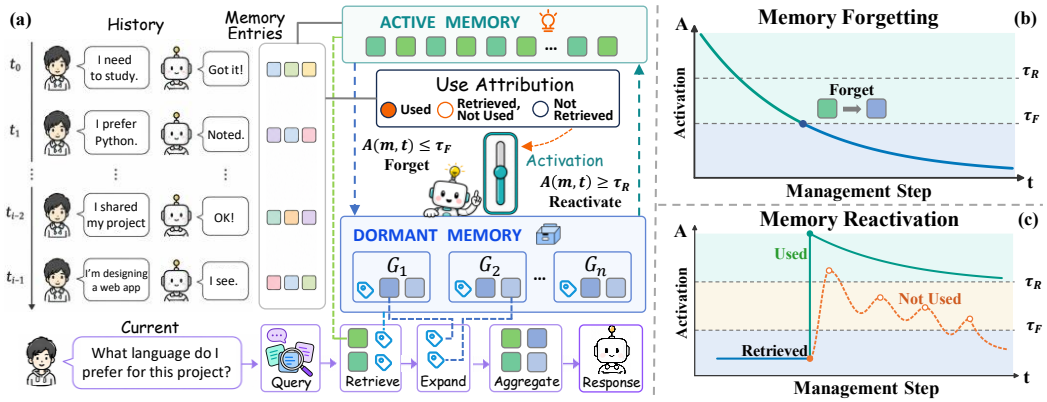


Figure 2: Overview of FOCUS, the first framework to manage retrieval space through reversible memory state transitions. (a) FOCUS retrieves Active memories directly and accesses Dormant memories through group summaries, with memory use attribution guiding activation updates that govern reversible transitions between the two sets. (b) An Active memory enters the Dormant Set when its activation falls below τ_F, while its full content remains stored. (c) Memory use refreshes activation, while retrieval without use provides temporary activation; a Dormant memory returns to the Active Set when activation reaches τ_R.

make the relevant context or historical state explicit. `IGNORE` leaves the memory store unchanged when the new item is redundant or lacks distinct durable information. Memories created through `ADD` and `UPDATE` enter the Active Set for subsequent retrieval. For `REWRITE`, the new fact enters the Active Set, while the rewritten memory retains its previous lifecycle state.

3.2 Activation Driven Memory Lifecycle Management

FOCUS manages the memory lifecycle by partitioning the memory store into an Active Set and a Dormant Set. Active memories participate directly in retrieval, while Dormant memories remain stored and are accessed through the mechanism described in Section 3.3. This separation decouples memory retention from retrieval participation, allowing the retrieval space to change without deleting stored content. Each memory is assigned an activation value that governs reversible transitions between the two sets. This activation changes over time and is further updated according to how the memory is retrieved and used during subsequent interactions.

We index lifecycle updates by step t, and let $\mathcal{A}_t$ and $\mathcal{D}_t$ denote the Active Set and Dormant Set at that step, with $\mathrm{Mem}_t = \mathcal{A}_t \cup \mathcal{D}_t$. For the current interaction, retrieval first produces a set of memories $\mathcal{R}_t$, representing the memories provided as context for response generation. After the response is generated, memory use attribution is then performed by an LLM to identify which retrieved memories contributed to the response, forming $\mathcal{U}_t \subseteq \mathcal{R}_t$. Retrieval records which memories were accessed, while memory use attribution estimates which retrieved memories support the generated response. Membership in $\mathcal{R}_t$ and $\mathcal{U}_t$ classifies each memory as used, retrieved without use, or not retrieved, providing the signals required for subsequent activation updates at each step.

The first component of activation captures how memory strength changes with time after attributed use. For a memory m, activation gradually decreases as the interval since its most recent contribution grows. Following the half life formulation used to model memory retention (Settles & Meeder, 2016), we define the natural decay as $A_d(m,t) = \exp[-\lambda(t - a(m))]$, where $a(m)$ denotes the most recent step at which m contributed to a response, or its creation step if it has never been used. The decay rate $\lambda > 0$ corresponds to a half life of $H_{\mathrm{idle}} = \ln 2/\lambda$, meaning that activation is halved after every H_{idle} management steps without use. Retrieval alone leaves $a(m)$ unchanged, preserving the most recent attributed use as the reference point for subsequent decay.

The second component reflects how the memory is used in the current interaction. When $m \in \mathcal{U}_t$, its activation is restored to 1, and the reference step is updated to $a(m) = t$. A memory that is retrieved without contributing to the response receives a temporary activation $A_r(m,t) = A_0 \exp[-\lambda_r(k_m(t) - 1)]$, where A_0 is the initial temporary activation and $k_m(t)$ counts consecutive retrievals of m without use since its most recent use. The decay rate λ_r corresponds to a retrieval half

life of $H_{\text{miss}} = \ln 2/\lambda_r$. As the miss streak grows, the temporary activation decreases, preventing repeated retrieval alone from sustaining high activation.

The final activation combines natural decay and temporary activation according to

$$A(m,t) = \begin{cases} 1, & m \in \mathcal{U}_t, \\ \max\{A_d(m,t), A_r(m,t)\}, & m \in \mathcal{R}_t \setminus \mathcal{U}_t, \\ A_d(m,t), & m \notin \mathcal{R}_t. \end{cases}$$

As shown in Figure 2(b), an Active memory moves to the Dormant Set when $A(m,t) \leq \tau_F$. A Dormant memory returns to the Active Set when $A(m,t) \geq \tau_R$, as illustrated in Figure 2(c). We set $\tau_R > \tau_F$ to reduce frequent transitions between the two lifecycle states.

Maintaining these activation states by scanning the complete memory store at every step would incur a cost of $O(|\text{Mem}_t|)$. FOCUS exploits the monotonic decay of $A_d(m,t)$ to avoid this full scan. For each Active memory, the next step at which its natural decay will fall below τ_F is computed in advance and maintained in a priority queue. Retrieval or use updates the corresponding schedule when the activation state changes, while version records in a hash table allow outdated queue entries to be ignored. At step t, $\mathcal{M}_F$ processes the retrieved memories together with Active memories whose scheduled transition time has arrived. The resulting lifecycle maintenance cost is approximately $O((|\mathcal{R}_t| + k_t)\log|\mathcal{A}_t|)$, where k_t denotes the number of scheduled Active memories processed at step t. Further implementation details are provided in Appendix A.2.

3.3 Memory Access across Lifecycle States

The Active Set and the Dormant Set use different access mechanisms. Active memories are matched directly against the query using embedding similarity, while Dormant memories remain stored and are accessed through compact group summaries that guide retrieval of relevant full memories when needed. This organization reduces the number of individual memory entries involved in direct query matching while preserving the full content of Dormant memories in the memory store.

Dormant memories are organized into semantic groups for later access. Let $\mathcal{G}_t = \{G_1, \ldots, G_{M_t}\}$ denote the groups maintained at step t. Each group G_j has a centroid $\mathbf{c}_j = \frac{1}{|G_j|}\sum_{m' \in G_j} \mathbf{e}_{m'}$, computed from the embeddings of its current Dormant members and updated as membership changes. When a memory m first enters the Dormant Set, its embedding is compared with the group centroids, and the K_G most similar groups are retained as candidates. An LLM examines m and the memories in these candidate groups to determine whether m belongs to an existing group. A new group is created when no group candidate satisfies the assignment threshold.

Each Dormant group G_j is represented in the retrieval space by a compact summary r_j, generated by an LLM from the full content of its current members. The summary captures the main semantic content of the group, and its embedding $\mathbf{e}_{r_j}$ is stored for efficient query matching during retrieval. Retrieval identifies relevant groups through these summary embeddings before accessing individual Dormant memories within selected groups. For space measurements, each Active memory entry and each Dormant group summary counts as one retrieval unit.

Group summaries are updated in batches to avoid repeated LLM calls during frequent lifecycle transitions, separating routine group maintenance from LLM based summary regeneration. When a Dormant memory is reactivated, the group centroid is updated immediately, while the existing summary is retained until the next refresh. If the memory returns to the same group before the refresh, the pending membership change is canceled. A new Dormant memory, or an `REWRITE` that changes the content of an existing Dormant memory, triggers a refresh that incorporates the accumulated membership changes and regenerates the summary from the current group members. A group summary is removed once the group contains no Dormant memories.

For a retrieval query x, $\mathcal{R}_F$ selects the K_A most similar memories from the Active Set. Dormant retrieval first matches x against the group summary embeddings to select the top K_R groups, and then retrieves the K_D most similar Dormant memories from each selected group. The Active and Dormant candidates are merged and reduced to the final top-k memories. During memory construction, each newly extracted atomic fact serves as x, and the retrieved memories provide candidates for `ADD`, `UPDATE`, `REWRITE`, or `IGNORE`. During response generation, the user query q_i serves as x, and the retrieved memories form $\mathcal{R}_t$ for response generation and subsequent memory use attribution.

Table 1: Main results on LoCoMo and Memora with two backbone models. LoCoMo reports accuracy on **single**-hop, **temporal**, **multi**-hop, and **open**-domain questions, while Memora reports Forgetting Aware Memory Accuracy (FAMA) under **Week**ly, **Month**ly, and **Quarter**ly settings. All values are reported on a 0–100 scale and averaged over three independent runs. Higher is better; boldface and underlining indicate the best and second best results within each backbone.

	LoCoMo Accuracy (%)					Memora FAMA (%)			
Method	**Single**	**Temporal**	**Multi**	**Open**	**Overall**	**Week**	**Month**	**Quarter**	**Overall**
Qwen3-30B-A3B-Instruct									
NativeRAG	76.10	32.71	59.93	51.04	62.53	35.67	29.35	27.72	30.91
A-MEM	62.07	41.74	46.45	52.08	54.35	39.61	31.25	34.40	35.09
LightMem	<u>81.57</u>	<u>47.98</u>	65.96	<u>56.25</u>	<u>70.13</u>	41.88	29.61	29.44	33.64
Mem0	54.34	32.09	40.78	52.08	47.08	<u>45.77</u>	<u>35.64</u>	33.93	<u>38.45</u>
MemoryOS	69.32	31.15	<u>66.31</u>	54.17	59.87	42.87	33.99	<u>35.72</u>	37.53
FOCUS	**85.14**	**54.83**	**78.01**	**60.42**	**75.97**	**51.82**	**40.60**	**38.50**	**43.64**
DeepSeek-V4-Flash									
NativeRAG	72.18	47.35	48.94	40.62	60.78	35.31	30.19	33.70	33.07
A-MEM	57.07	48.60	44.68	39.58	51.95	41.51	34.03	<u>37.45</u>	37.66
LightMem	**81.09**	<u>71.96</u>	<u>71.28</u>	<u>52.08</u>	<u>75.58</u>	43.72	30.49	28.00	34.07
Mem0	56.72	58.57	39.01	37.50	52.66	<u>56.43</u>	<u>44.55</u>	33.55	<u>44.84</u>
MemoryOS	66.11	59.81	58.87	43.75	62.08	39.94	36.32	29.74	35.33
FOCUS	<u>80.02</u>	**80.37**	**71.63**	**68.75**	**77.86**	**61.98**	**52.50**	**39.63**	**51.37**

4 EXPERIMENTS

We evaluate FOCUS on LoCoMo and Memora with two backbone models, comparing its memory performance and retrieval space with existing memory systems. Results on Memora further examine whether the method can retain useful information while excluding information that should be forgotten as interactions accumulate. To understand the source of its performance gains, we conduct ablation studies on the main lifecycle components. Further analysis examines the evolution of retrieval space over long interaction histories and the cost of memory construction.

4.1 EXPERIMENTAL SETUP

Experimental Details. Following LightMem (Fang et al., 2026), we process dialogue turns incrementally in chronological order throughout all benchmark runs. FOCUS adopts the same buffering and topic segmentation setup as LightMem, with a 512 token sensory buffer and a 768 token construction buffer, and uses LLMLingua-2 for topic segmentation (Pan et al., 2024). We evaluate all methods with Qwen3-30B-A3B-Instruct-2507 (Yang et al., 2025) and DeepSeek-V4-Flash (DeepSeek-AI et al., 2026) as backbone models. For all methods, we use all-MiniLM-L6-v2 as the embedding model, which produces embeddings of dimension 384 (Reimers & Gurevych, 2019; Wang et al., 2020). Additional implementation details are provided in Appendix B.3.

Datasets and Metrics. We evaluate memory performance over long interaction histories using LoCoMo (Maharana et al., 2024) and Memora (Uddin et al., 2026). LoCoMo contains extended conversations spanning multiple sessions, with about 588 dialogue turns per conversation on average, and we report Accuracy following its official evaluation protocol. Memora provides longer interaction histories, averaging approximately 2.5K, 9.6K, and 31.3K turns under its Weekly, Monthly, and Quarterly settings, and is evaluated using its official Forgetting Aware Memory Accuracy (FAMA), which evaluates the use of valid information and penalizes responses that include obsolete information. GPT-4o-mini is used as the LLM judge for both benchmarks following their evaluation protocols. We measure retrieval space by the number of units involved in query matching. For memory construction and maintenance, we report LLM call counts alongside input and output token consumption. Further details are provided in Appendix B.1.

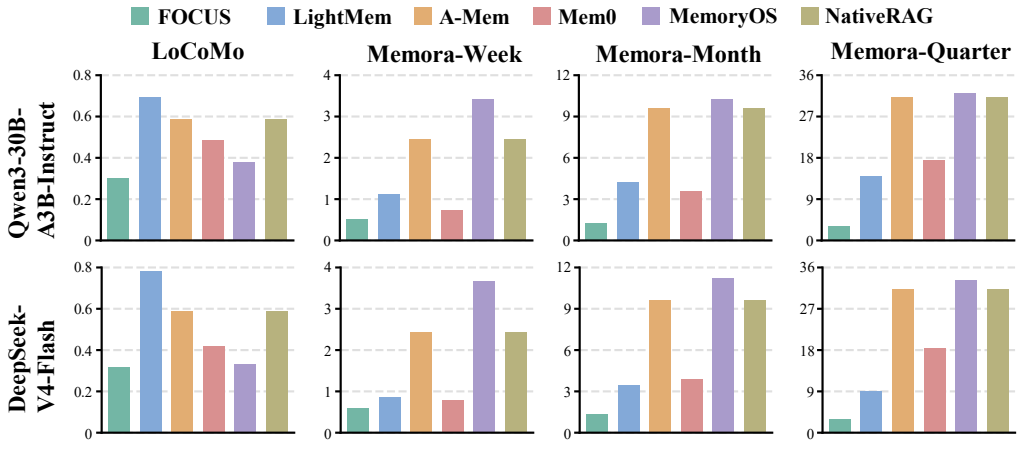


Figure 3: Retrieval space size across memory methods on LoCoMo and Memora. The horizontal axis represents different methods, while the vertical axis reports retrieval units in thousands (10^3). For FOCUS, retrieval units include Active memory entries and Dormant group summaries. FOCUS maintains the smallest retrieval space across all settings under both backbones.

Baseline Methods. We compare FOCUS with representative methods covering several directions in long term memory design. NativeRAG (Lewis et al., 2020) serves as a conventional retrieval baseline without explicit memory organization, while A-MEM (Xu et al., 2025) represents structured associative memory organization. LightMem (Fang et al., 2026) is included as a method that jointly considers memory performance and efficiency. We also compare with Mem0 (Chhikara et al., 2025) and MemoryOS (Kang et al., 2025), which provide more comprehensive memory management through explicit memory operations, with MemoryOS further introducing hierarchical organization. Further details on the baseline implementations are provided in Appendix B.2.

4.2 Main Results

Table 1 presents the main results on LoCoMo and Memora with Qwen and DeepSeek as backbones, averaged over three independent runs. It reports LoCoMo accuracy across four question categories and Memora FAMA across the Weekly, Monthly, and Quarterly settings. On LoCoMo, FOCUS achieves consistent overall improvements, exceeding the best competing baseline by 5.84 points with Qwen and 2.28 points with DeepSeek. On Memora, FOCUS delivers further gains of 5.19 and 6.53 points in overall FAMA with Qwen and DeepSeek.

The LoCoMo results demonstrate broad improvements across question categories. With Qwen, FOCUS achieves the highest accuracy in all four categories, exceeding the best competing baseline by 3.57–11.70 points. With DeepSeek, it leads in three categories, with gains ranging from 0.35 to 16.67 points. These results suggest that FOCUS supports effective memory access and reasoning across diverse long term memory queries under both backbones.

Memora provides a complementary evaluation across increasing interaction lengths. FOCUS achieves the highest FAMA in the Weekly, Monthly, and Quarterly settings under both backbones. Compared with the best competing baseline at each time span, the gains range from 2.77 to 6.05 points with Qwen and from 2.18 to 7.95 points with DeepSeek. Although performance generally declines as histories grow longer, FOCUS maintains positive margins throughout, suggesting that its advantage persists across the tested interaction lengths.

Figure 3 compares retrieval space size across the same benchmark settings. FOCUS consistently maintains the smallest retrieval space under both backbones on LoCoMo and all three Memora settings. The gap widens as interaction histories grow, with the largest separation appearing in Memora Quarterly. In this longest setting, FOCUS uses only 3.16K retrieval units with Qwen and 2.99K with DeepSeek, compared with 14.06K and 9.11K for LightMem, which has the smallest competing retrieval space under both backbones. These results represent substantial retrieval space reductions of 77.5% with Qwen and 67.2% with DeepSeek.

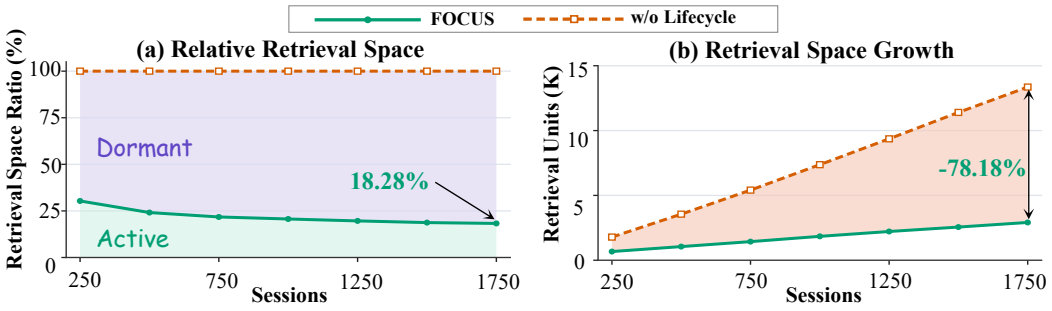


Figure 4: Retrieval space dynamics on Memora Quarterly using Qwen3-30B-A3B-Instruct. (a) Ratio of retrieval units to stored memory entries as interaction sessions accumulate. (b) Growth in the number of retrieval units for FOCUS and the variant without Lifecycle.

4.3 Memory Lifecycle Analysis

To examine the behavior of FOCUS over long interaction histories, we use Memora Quarterly with Qwen and report results up to 1,750 sessions, the common range across all Quarterly histories. The retrieval space of FOCUS consists of Active memory entries and Dormant group summaries, while the full content of Dormant memories remains stored. To isolate the effect of lifecycle management, we compare FOCUS with *w/o Lifecycle*, which preserves memory construction but keeps all stored memories directly searchable in a single set. As shown in Figure 4(a), the retrieval space accounts for 30.34% of the complete memory store at 250 sessions and gradually decreases to 18.28% at 1,750 sessions. This declining ratio indicates that the retrieval space grows more slowly than the complete memory store as interactions accumulate.

Figure 4(b) compares the number of retrieval units under both settings. The retrieval space grows in both cases, but its increase under FOCUS is considerably slower, with the gap widening over longer interaction histories. At 1,000 sessions, FOCUS has 1.84K retrieval units compared with 7.37K for *w/o Lifecycle*, corresponding to a reduction of 75.00%. At 1,750 sessions, these values reach 2.91K and 13.36K retrieval units, increasing the reduction to 78.18%.

4.4 Ablation Study

We conduct ablation studies on LoCoMo and Memora Quarterly using Qwen to evaluate the contributions of key FOCUS components. *w/o Lifecycle* follows the configuration described in Section 4.3; *w/o Attribution* omits the step identifying retrieved memories contributing to the response; *w/o Reactivation* prevents Dormant memories from returning to the Active Set while preserving their retrieval; and *w/o Op Selection* applies `ADD` to every extracted fact, disabling all other operations.

As shown in Table 2(a), all ablations reduce LoCoMo accuracy and Memora FAMA. Disabling operation selection produces the largest declines of 11.59 and 5.91 points, highlighting the importance of maintaining memory quality as information accumulates. Removing lifecycle management reduces the two scores by 6.14 and 2.99 points. Disabling reactivation leads to decreases of 2.81 and 2.03 points despite preserving access to Dormant memories, suggesting that restoring their direct participation in retrieval benefits subsequent interactions. Removing memory use attribution results in declines of 3.62 and 1.56 points, supporting its role in lifecycle management.

4.5 Memory Construction and Maintenance Cost

Table 2(b) compares the LLM call and token costs of memory construction and maintenance. LightMem has the lowest construction cost, while FOCUS requires additional LLM calls for memory updating and memory use attribution. FOCUS adopts the same topic based buffering strategy as LightMem, allowing memory extraction to be performed in batches across multiple interaction turns. FOCUS uses fewer tokens and LLM calls than A-MEM, Mem0, and MemoryOS on both LoCoMo and Memora. The additional memory management operations increase construction cost relative to LightMem, while the overall cost remains lower than the other competing methods.

Table 2: (a) Ablation results on LoCoMo and Memora Quarterly using Qwen3-30B-A3B-Instruct. (b) Memory construction cost on LoCoMo and Memora Quarterly using Qwen3-30B-A3B-Instruct. Input and output tokens are reported in millions (M), and LLM calls in thousands (k).

(a) Ablation study

Setting	LoCoMo	Memora
	Acc	FAMA
Full	**75.97**	**38.50**
w/o Lifecycle	69.83	35.51
w/o Attribution	72.35	36.94
w/o Reactivation	73.16	36.47
w/o Op Selection	64.38	32.59

(b) LLM cost for memory construction and maintenance

Method	LoCoMo			Memora		
	In (M)	Out (M)	Calls (k)	In (M)	Out (M)	Calls (k)
A-MEM	9.15	3.80	11.75	622.73	87.94	428.19
LightMem	0.79	0.38	0.48	38.20	18.36	31.60
Mem0	10.30	1.29	8.70	503.39	50.78	309.23
MemoryOS	5.42	2.31	11.43	363.10	70.41	448.72
FOCUS	5.42	0.80	8.02	330.54	25.10	286.22

5 Related Work

Memory Systems for LLM Agents. Long term memory enables LLM agents to preserve information across interactions and maintain continuity beyond the current context (Park et al., 2023; Packer et al., 2024; Zhong et al., 2024). Early approaches mainly stored interaction history or extracted experiences in persistent memory for later retrieval. As interaction histories become longer, recent systems increasingly use structured memory organizations to maintain accumulated information and support continued updating. Associative and graph based methods connect related memories or distributed evidence (Xu et al., 2025; Gutiérrez et al., 2024; Xu et al., 2026), while hierarchical methods organize memories across different levels (Kang et al., 2025; Sun et al., 2026). As memory systems become more sophisticated, the cost of constructing and maintaining them over long interaction histories has received increasing attention. LightMem (Fang et al., 2026) addresses this issue through buffered processing and topic segmentation, reducing repeated model calls during memory construction. SimpleMem (Liu et al., 2025) further improves efficiency through compact memory representations and streamlined consolidation. However, existing work has focused mainly on how memories are represented, updated, and retrieved, with less attention to controlling which stored memories remain part of the retrieval space as interaction histories grow.

Memory Updating and Forgetting. Another line of research studies how stored memories are revised. Some approaches refine or consolidate existing memories through reflection or evidence accumulated from later interactions (Tan et al., 2025; Dai et al., 2026). More recent work formulates memory maintenance as explicit decisions over how stored information should be updated, retained, or removed (Yan et al., 2026; Yu et al., 2026; Ma et al., 2026). This progression extends memory maintenance from revising stored content to determining how information should be retained over time. Forgetting has also received increasing attention as part of long term memory management (Zhong et al., 2024; Honda et al., 2026). Recent evaluations recognize selective forgetting as a distinct capability of memory agents (Hu et al., 2026). These studies establish forgetting as an explicit component of long term memory management, alongside memory updating and retention. However, forgetting decisions are made before future information needs are known, creating a tension between reducing the influence of currently less useful memories and preserving information that may become relevant again. FOCUS addresses this tension through a reversible memory lifecycle that controls retrieval participation while retaining Dormant memories for later recovery.

6 Conclusion

In this work, we introduced FOCUS, a long term memory framework for managing which memories participate directly in retrieval over extended interactions. FOCUS manages stored memories through a reversible lifecycle, allowing less active memories to leave the main retrieval space while remaining available for later recovery. Experiments on LoCoMo and Memora show that FOCUS improves long term memory performance while maintaining a smaller retrieval space across both backbone models. Future work will explore more efficient use of the Active Set to reduce repeated memory retrieval and context processing across interactions. We also plan to evaluate these improvements in personal assistant agents operating over extended periods in practical settings.

AI USE STATEMENT

In this work, we used generative AI tools to assist with language polishing, improve the clarity and readability of the manuscript, and support parts of the code implementation. All AI-assisted content was carefully reviewed and verified by the authors before being incorporated into the final work. The authors take full responsibility for the content, accuracy, and integrity of this work.

REPRODUCIBILITY STATEMENT

We provide the implementation details necessary to reproduce our experiments in the appendix and supplementary materials. The complete configuration of FOCUS is described in Appendix B.3, including the memory update rules, hyperparameters, and retrieval settings. Detailed experimental settings, datasets, baseline configurations, and evaluation protocols are provided in Appendix B. We also provide anonymous source code and configuration files in the supplementary material.

REFERENCES

Robert A. Bjork and Elizabeth L. Bjork. A new theory of disuse and an old theory of stimulus fluctuation. In Alice F. Healy, Stephen M. Kosslyn, and Richard M. Shiffrin (eds.), *From Learning Processes to Cognitive Processes: Essays in Honor of William K. Estes*, volume 2, pp. 35–67. Lawrence Erlbaum Associates, Hillsdale, NJ, 1992.

Daniil A Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624(7992):570–578, 2023.

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.

Zijie Dai, Shiyuan Deng, Sheng Guan, Yizhou Tian, Xin Yao, Xiao Yan, and James Cheng. RecMem: Recurrence-based memory consolidation for efficient and effective long-running LLM agents. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 32353–32376, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.1619. URL https://aclanthology.org/2026.findings-acl.1619/.

DeepSeek-AI, Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chengyu Hou, et al. Deepseek-v4: Towards highly efficient million-token context intelligence, 2026. URL https://arxiv.org/abs/2606.19348.

Jizhan Fang, Xinle Deng, Haoming Xu, Ziyan Jiang, Yuqi Tang, Ziwen Xu, Shumin Deng, Yunzhi Yao, Mengru Wang, Shuofei Qiao, Huajun Chen, and Ningyu Zhang. Lightmem: Lightweight and efficient memory-augmented generation. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 98706–98729, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/a05b72653ec5b473732129829ae04195-Paper-Conference.pdf.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. Hipporag: Neurobiologically inspired long-term memory for large language models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 59532–59569. Curran Associates, Inc., 2024. doi: 10.52202/079017-1902. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/6ddc001d07ca4f319af96a3024f6dbd1-Paper-Conference.pdf.

Yudai Honda, Yuki Fujita, Keiichi Zempo, and Shogo Fukushima. Human-like remembering and forgetting in llm agents: An act-r-inspired memory architecture. In *Proceedings of the 13th International Conference on Human-Agent Interaction*, HAI '25, pp. 229–237, New York, NY, USA, 2026. Association for Computing Machinery. ISBN 9798400721786. doi: 10.1145/3765766.3765803. URL https://doi.org/10.1145/3765766.3765803.

Yuanzhe Hu, Yu Wang, and Julian McAuley. Evaluating memory in llm agents via incremental multi-turn interactions. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 156259–156291, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/fd1eff9dd295df50a41f2521942fa31d-Paper-Conference.pdf.

Jiazheng Kang, Mingming Ji, Zhe Zhao, and Ting Bai. Memory OS of AI agent. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 25961–25970, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1318. URL https://aclanthology.org/2025.emnlp-main.1318/.

Patrick 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 H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf.

Hao Li, Chenghao Yang, An Zhang, Yang Deng, Xiang Wang, and Tat-Seng Chua. Hello again! LLM-powered personalized agent for long-term dialogue. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 5259–5276, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.272. URL https://aclanthology.org/2025.naacl-long.272/.

Jiaqi Liu, Yaofeng Su, Peng Xia, Yiyang Zhou, Siwei Han, Zeyu Zheng, Cihang Xie, Mingyu Ding, and Huaxiu Yao. Simplemem: Efficient lifelong memory for llm agents. *arXiv preprint arXiv:2601.02553*, 2025. URL https://github.com/aiming-lab/SimpleMem.

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. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Chris Lu, Cong Lu, Robert Tjarko Lange, Yutaro Yamada, Shengran Hu, Jakob Foerster, David Ha, and Jeff Clune. Towards end-to-end automation of ai research. *Nature*, 651(8107):914–919, 2026.

Wenquan Ma, Jiayan Nan, and WenLong Wu. What deserves memory: Adaptive memory distillation for LLM agents. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 34789–34812, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1607. URL https://aclanthology.org/2026.acl-long.1607/.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of LLM agents. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13851–13870, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.747. URL https://aclanthology.org/2024.acl-long.747/.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. Memgpt: Towards llms as operating systems, 2024. URL https://arxiv.org/abs/2310.08560.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. LLMLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression.

In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 963–981, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.57. URL https://aclanthology.org/2024.findings-acl.57/.

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*, UIST '23, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400701320. doi: 10.1145/3586183.3606763. URL https://doi.org/10.1145/3586183.3606763.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410. URL https://aclanthology.org/D19-1410/.

Burr Settles and Brendan Meeder. A trainable spaced repetition model for language learning. In Katrin Erk and Noah A. Smith (eds.), *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1848–1858, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1174. URL https://aclanthology.org/P16-1174/.

Haoran Sun, Shaoning Zeng, and Bob Zhang. H-MEM: Hierarchical memory for high-efficiency long-term reasoning in LLM agents. In Vera Demberg, Kentaro Inui, and Lluís Marquez (eds.), *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 341–350, Rabat, Morocco, March 2026. Association for Computational Linguistics. ISBN 979-8-89176-380-7. doi: 10.18653/v1/2026.eacl-long.15. URL https://aclanthology.org/2026.eacl-long.15/.

Zhen Tan, Jun Yan, I-Hung Hsu, Rujun Han, Zifeng Wang, Long Le, Yiwen Song, Yanfei Chen, Hamid Palangi, George Lee, Anand Rajan Iyer, Tianlong Chen, Huan Liu, Chen-Yu Lee, and Tomas Pfister. In prospect and retrospect: Reflective memory management for long-term personalized dialogue agents. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8416–8439, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.413. URL https://aclanthology.org/2025.acl-long.413/.

Md Nayem Uddin, Kumar Shubham, Eduardo Blanco, Chitta Baral, and Gengyu Wang. From recall to forgetting: Benchmarking long-term memory for personalized agents. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 26814–26841, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.1337. URL https://aclanthology.org/2026.findings-acl.1337/.

Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. Augmenting language models with long-term memory. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 74530–74543. Curran Associates, Inc., 2023. doi: 10.52202/075280-3259. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/ebd82705f44793b6f9ade5a669d0f0bf-Paper-Conference.pdf.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 5776–5788. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

John T Wixted. The psychology and neuroscience of forgetting. *Annu. Rev. Psychol.*, 55(1):235–269, 2004.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. Longmemeval: Benchmarking chat assistants on long-term interactive memory. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 86809–86836, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/d813d324dbf0598bbdc9c8e79740ed01-Paper-Conference.pdf.

Buqiang Xu, Yijun Chen, Jizhan Fang, Ruobin Zhong, Yunzhi Yao, Yuqi Zhu, Lun Du, and Shumin Deng. StructMem: Structured memory for long-horizon behavior in LLMs. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 122–146, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-391-3. doi: 10.18653/v1/2026.acl-short.12. URL https://aclanthology.org/2026.acl-short.12/.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 17577–17604. Curran Associates, Inc., 2025. doi: 10.52202/085713-0593. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/19909c36f51abc4856b4560aff3d36d6-Paper-Conference.pdf.

Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Jinhe Bi, Kristian Kersting, Jeff Z. Pan, Hinrich Schuetze, Volker Tresp, and Yunpu Ma. Memory-r1: Enhancing large language model agents to manage and utilize memories via reinforcement learning. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12805–12825, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.583. URL https://aclanthology.org/2026.acl-long.583/.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, et al. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

John Yang, Carlos Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 50528–50652. Curran Associates, Inc., 2024. doi: 10.52202/079017-1601. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/5a7c947568c1b1328ccc5230172e1e7c-Paper-Conference.pdf.

Yi Yu, Liuyi Yao, Yuexiang Xie, Qingquan Tan, Jiaqi Feng, Yaliang Li, and Libing Wu. Agentic memory: Learning unified long-term and short-term memory management for large language model agents. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 21457–21483, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.981. URL https://aclanthology.org/2026.acl-long.981/.

Zeyu Zhang, Quanyu Dai, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. A survey on the memory mechanism of large language model-based agents. *ACM Trans. Inf. Syst.*, 43(6), September 2025. ISSN 1046-8188. doi: 10.1145/3748302. URL https://doi.org/10.1145/3748302.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 19724–19731, 2024. URL https://ojs.aaai.org/index.php/AAAI/article/view/29946.

A Method Details

A.1 Memory Construction Pipeline

Following the buffering and topic segmentation procedure of LightMem (Fang et al., 2026), FOCUS processes incoming dialogue turns incrementally in chronological order. When the temporary buffer reaches its preset token capacity, let $s_1, \ldots, s_n$ denote the dialogue turns currently stored in the buffer. Topic segmentation is then applied to these turns as a batch. The trailing portion that has not yet formed a completed topic segment is retained for the next buffer, preventing a topic from being split solely by the buffer boundary.

To identify candidate topic boundaries, FOCUS constructs a turn-level attention matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$, where $M_{i,j}$ denotes the attention score between turns s_i and s_j. We focus on the attention between each turn and its immediately preceding turn, given by the sequence of entries $M_{k,k-1}$. Following LightMem, a position is treated as an attention-based candidate boundary when this score forms a local peak relative to its neighboring positions:

$$B_1 = \{k \,|\, M_{k,k-1} > M_{k-1,k-2},\ M_{k,k-1} > M_{k+1,k},\ 1 < k < n\}\,.$$

Each $k \in B_1$ therefore marks a possible transition immediately before turn s_k.

Semantic similarity is then used to verify these candidate transitions. Let $\mathrm{sim}(s_{k-1}, s_k)$ denote the semantic similarity between two adjacent turns, and let τ_s be the similarity threshold used for topic segmentation. Positions at which adjacent turns are sufficiently dissimilar form

$$B_2 = \{k \,|\, \mathrm{sim}(s_{k-1}, s_k) < \tau_s,\ 2 \le k \le n\}\,.$$

The final topic boundaries are given by $B = B_1 \cap B_2$, so a boundary is retained only when it is supported by both the attention pattern and a semantic change between adjacent turns. These boundaries partition the buffered dialogue into topic segments T_i, which are appended in order to the construction buffer $\mathcal{B} = \{T_1, \ldots, T_N\}$. Memory extraction is triggered once $\mathcal{B}$ reaches its preset capacity.

At each extraction, the writing operation $\mathcal{W}_F$ maps the buffered topic segments to the set of atomic memory entries $\mathcal{N} = \mathcal{W}_F(\mathcal{B})$. The entries in $\mathcal{N}$ are then processed individually using the incorporation procedure defined in Section 3.1. For each entry, $\mathcal{R}_F$ retrieves related memories from the Active and Dormant sets, after which the LLM selects one of `ADD`, `UPDATE`, `REWRITE`, or `IGNORE`. The selected operation is applied through $\mathcal{M}_F$, and the process continues until all newly extracted entries have been incorporated into the memory store.

A.2 Efficient Lifecycle Maintenance

Algorithm 1 specifies the exact update order for the lifecycle maintenance mechanism, including retrieval feedback, state transitions, and scheduled checks of Active memories.

B Experimental Details

B.1 Datasets and Evaluation Metrics

LoCoMo. LoCoMo (Maharana et al., 2024) evaluates long term conversational memory over extended multi session dialogues. The released benchmark contains ten conversations, each with an average of 27.2 sessions, 588 dialogue turns, and 16.6K tokens. The conversations follow persistent personas and events that unfold across sessions, requiring information introduced earlier in the history to be recalled or combined with later context. We evaluate the question answering task using four question types: Single Hop, Multi Hop, Temporal, and Open Domain, comprising 841, 282, 321, and 96 questions, respectively. Single Hop questions rely on a single relevant part of the dialogue, while Multi Hop questions require information from multiple parts of the history. Temporal questions test reasoning over dates and temporal relations, and Open Domain questions combine conversational evidence with general knowledge. Following the standard evaluation setting, we report Accuracy (ACC), computed as

$$\mathrm{ACC} = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \mathbb{I}[J(\hat{y}_q, y_q) = 1]\,,$$

Algorithm 1 Efficient Lifecycle Maintenance

Require: Current management step t, retrieved memories $\mathcal{R}_t$, used memories $\mathcal{U}_t$, priority queue Q, schedule version table V

for all $m \in \mathcal{R}_t$ **do**
  if $m \in \mathcal{U}_t$ **then**
    $A(m,t) \leftarrow 1$
    $a(m) \leftarrow t$
    $k_m(t) \leftarrow 0$
  else
    $k_m(t) \leftarrow k_m(t-1) + 1$
    $A(m,t) \leftarrow \max\{A_d(m,t), A_r(m,t)\}$
  end if
  if m is Active **then**
    if $A(m,t) \leq \tau_F$ **then**
      $V[m] \leftarrow V[m] + 1$
      MoveToDormant(m)
    else
      $d \leftarrow \min\{t' > t \mid A_d(m,t') \leq \tau_F\}$
      $V[m] \leftarrow V[m] + 1$
      Push$(Q, (d, m, V[m]))$
    end if
  else if m is Dormant and $A(m,t) \geq \tau_R$ **then**
    MoveToActive(m)
    $d \leftarrow \min\{t' > t \mid A_d(m,t') \leq \tau_F\}$
    $V[m] \leftarrow V[m] + 1$
    Push$(Q, (d, m, V[m]))$
  end if
end for
while $Q \neq \emptyset$ and $Q.\min.d \leq t$ **do**
  $(d, m, v) \leftarrow$ PopMin(Q)
  if $v \neq V[m]$ **then**
    continue
  end if
  if m is not the current memory version or m is not Active **then**
    continue
  end if
  $A(m,t) \leftarrow A_d(m,t)$
  if $A(m,t) \leq \tau_F$ **then**
    $V[m] \leftarrow V[m] + 1$
    MoveToDormant(m)
  else
    $d' \leftarrow \min\{t' > t \mid A_d(m,t') \leq \tau_F\}$
    $V[m] \leftarrow V[m] + 1$
    Push$(Q, (d', m, V[m]))$
  end if
end while

where $\mathcal{Q}$ is the set of evaluation questions, $\hat{y}_q$ and y_q denote the generated and reference answers, and J is the binary judgement of the LLM evaluator.

Memora. Memora (Uddin et al., 2026) benchmarks long term memory for personalized agents through interaction histories that develop over weeks or months. The benchmark contains ten professional personas and tracks three forms of user memory. Preference memory records changing likes and dislikes, activity memory captures personal and professional activities, and goal memory represents objectives that persist across interactions. These memories evolve through ADD, UPDATE, and DELETE operations, allowing the benchmark to record how user information is introduced, revised, and invalidated over time. Memora provides Weekly, Monthly, and Quarterly

settings with an average of 155, 615, and 1,991 sessions per persona, respectively, and roughly 16 dialogue turns per session. The three settings contain an average of 103.2, 374.3, and 1,171.4 memory operations, while their evaluation sets contain 150, 150, and 300 questions together with 749, 1,421, and 4,884 evaluation criteria. The questions cover Remembering, Reasoning, and Recommending. Remembering tests the use of relevant stored information, Reasoning requires information to be combined across the interaction history, and Recommending evaluates personalized responses against the user's current memory state.

Memora evaluates each response with atomic criteria derived from the memory state associated with the question. Memory presence criteria specify information that should be reflected in the response, whereas forgetting absence criteria identify information that has become invalid and should no longer be used. Their satisfaction rates are measured by Memory Presence Accuracy (MPA) and Forgetting Absence Accuracy (FAA). For a question with N_{presence} presence criteria and N_{forget} forgetting criteria, the forgetting weight is defined as $\lambda = N_{\text{forget}}/(N_{\text{presence}} + N_{\text{forget}})$. Memora then combines the two accuracies using Forgetting-Aware Memory Accuracy (FAMA),

$$\text{FAMA} = \max\left(0,\ \text{MPA} - \lambda(1 - \text{FAA})\right).$$

MPA measures how well the response preserves information that remains valid, while $1 - \text{FAA}$ captures violations caused by information that should have been forgotten. The weight λ scales this penalty according to the proportion of forgetting criteria associated with the question. FAMA is computed at the question level and then aggregated within each task and temporal setting, with the reported scores scaled to 0–100.

B.2 Baseline Methods

We compare FOCUS with five representative baselines that cover complementary designs for long term memory, including direct retrieval, operation based memory updating, associative memory organization, hierarchical memory management, and efficiency oriented memory construction. For LoCoMo, we follow the baseline implementations released with LightMem (Fang et al., 2026)[1], while for Memora we use its official benchmark implementation (Uddin et al., 2026)[2]. The number of retrieved entries for answer generation are same among all methods.

NativeRAG. NativeRAG follows the standard retrieval augmented generation paradigm (Lewis et al., 2020) and serves as a simple retrieval baseline without explicit memory construction or update. Dialogue history is indexed directly as retrievable text units, and each query retrieves the most semantically relevant units through embedding similarity. The retrieved content is then added to the context used to generate the response.

LightMem. LightMem (Fang et al., 2026) organizes memory construction as a multistage pipeline designed to reduce the cost of processing long interaction histories. It first compresses incoming dialogue and identifies topic boundaries using attention patterns together with semantic similarity, after which related turns are accumulated by topic and processed in batches to construct memory entries. Memory maintenance is further separated from online interaction through a sleep time update stage that consolidates and revises stored memories offline.

Mem0. Mem0 (Chhikara et al., 2025) represents conversational memory as concise factual entries extracted from new dialogue. For each extracted fact, the system retrieves related memories from the existing store and asks an LLM to select among `ADD`, `UPDATE`, `DELETE`, and `NOOP` according to the relation between the new and existing information. The resulting facts are maintained in a vector memory store and retrieved by semantic similarity for subsequent responses.

A-MEM. A-MEM (Xu et al., 2025) represents interactions as structured memory notes containing the memory content together with contextual descriptions, keywords, tags, and other attributes. When a new note is added, related historical notes are retrieved and linked according to their semantic relationships. New information can also trigger the evolution of existing notes, allowing both their attributes and their connections to change as the memory store develops.

[1]Repository: `https://github.com/zjunlp/LightMem`

[2]Repository: `https://github.com/geniesinc/Memora`

Table 3: FAMA scores on Memora for each task and temporal setting. W, M, and Q denote the Weekly, Monthly, and Quarterly settings. Scores are reported on a 0–100 scale. Bold and underlined values indicate the highest and second-highest distinct scores under each backbone, including ties.

	Remembering			Recommending			Reasoning		
Method	W	M	Q	W	M	Q	W	M	Q
Qwen3-30B-A3B-Instruct									
NativeRAG	63.88	41.11	41.90	39.13	39.94	37.25	4.00	7.00	4.00
A-MEM	<u>64.14</u>	47.26	**54.86**	46.01	38.48	37.85	8.67	8.00	10.50
LightMem	50.25	33.99	36.11	45.39	41.83	37.70	<u>30.00</u>	13.00	**14.50**
Mem0	63.80	47.94	47.39	<u>56.18</u>	<u>44.97</u>	40.89	17.33	<u>14.00</u>	<u>13.50</u>
MemoryOS	64.06	<u>48.54</u>	50.59	49.21	42.42	<u>43.07</u>	15.33	11.00	<u>13.50</u>
FOCUS	**65.37**	**50.53**	<u>54.47</u>	**58.08**	**56.27**	**46.51**	**32.00**	**15.00**	**14.50**
DeepSeek-V4-Flash									
NativeRAG	65.09	48.93	<u>59.79</u>	37.50	33.63	35.82	3.33	8.00	5.50
A-MEM	<u>74.50</u>	<u>57.67</u>	59.70	41.35	34.41	41.66	8.67	10.00	11.00
LightMem	55.10	30.55	35.71	55.40	47.91	39.29	20.67	13.00	9.00
Mem0	66.03	45.90	43.68	**67.93**	<u>60.74</u>	<u>43.48</u>	<u>35.33</u>	<u>27.00</u>	<u>13.50</u>
MemoryOS	60.12	40.91	38.41	45.71	50.04	36.30	14.00	18.00	**14.50**
FOCUS	**75.54**	**59.39**	**60.35**	<u>67.41</u>	**63.10**	**44.05**	**43.00**	**35.00**	**14.50**

MemoryOS. MemoryOS (Kang et al., 2025) organizes conversational memory into Short Term Memory, Mid Term Memory, and Long Term Personal Memory. Recent interactions first enter Short Term Memory and are subsequently transferred and organized into longer term layers according to recency, topic relationships, and memory usage. During response generation, relevant information is retrieved across these memory levels to provide both recent conversational context and longer term user information.

B.3 Implementation Details

We use Qwen3-30B-A3B-Instruct-2507 and DeepSeek-V4-Flash as the language models for memory construction and question answering. Qwen3-30B-A3B-Instruct-2507 is deployed locally with vLLM 0.8.5 on two NVIDIA A100 GPUs, while DeepSeek-V4-Flash is accessed through its official API. For both backbone models, we set the generation temperature to 0.3. GPT-4o-mini is used as the LLM judge for response evaluation.

Following LightMem (Fang et al., 2026), we set the sensory buffer capacity to 512 tokens and the construction buffer capacity to 768 tokens. Topic segmentation uses LLMLingua-2 with the `bert-base-multilingual-cased-meetingbank` model and a compression rate of 0.6. We use `all-MiniLM-L6-v2` to produce representations with 384 dimensions for memory retrieval and grouping. For the FOCUS memory lifecycle, we set $A_0 = 0.8$, $H_{\text{idle}} = 30$, $H_{\text{miss}} = 5$, $\tau_F = 0.35$, and $\tau_R = 0.45$.

For Dormant memory organization, we set $K_G = 10$ and use a group join threshold of 0.7 when assigning a memory to an existing group. During retrieval, we retrieve $K_A = 40$ memories from the Active Set and select the top $K_R = 10$ Dormant groups according to their summary similarity. From each selected Dormant group, we retrieve up to $K_D = 10$ memories. The Active and Dormant candidates are then merged and reranked by cosine similarity, with at most 60 memories retained for subsequent processing.

C Additional Results

C.1 Detailed Benchmark Results

Table 3 provides a finer breakdown of Memora performance across Remembering, Recommending, and Reasoning under the Weekly, Monthly, and Quarterly settings. With Qwen3-30B-A3B-Instruct,

Table 4: Parameter sensitivity of FOCUS with Qwen3-30B-A3B-Instruct. Only H_{idle}, K_G, or K_R is varied in each row; all other settings follow Appendix B.3. LoCoMo reports overall accuracy, and Memora reports FAMA. The default configuration is shown in bold. All scores are on a 0–100 scale.

Parameter Configuration			LoCoMo	Memora		
H_{idle}	K_G	K_R	Accuracy	Weekly	Monthly	Quarterly
30	**10**	**10**	75.97	51.82	40.60	38.50
15	10	10	74.87	51.48	39.87	38.23
60	10	10	76.10	51.16	40.35	37.84
30	5	10	74.61	50.83	40.08	37.92
30	20	10	72.47	50.39	39.73	38.46
30	10	5	75.78	51.74	40.37	37.63
30	10	20	74.35	50.36	39.52	38.08

Table 5: Performance variability on LoCoMo and Memora under both backbone models. Values are reported as mean ± sample standard deviation over three runs.

	Qwen3-30B-A3B-Instruct				DeepSeek-V4-Flash			
	LoCoMo Accuracy (%)	Memora FAMA (%)			LoCoMo Accuracy (%)	Memora FAMA (%)		
Method	ACC	Week	Month	Quarter	ACC	Week	Month	Quarter
NativeRAG	62.53±0.51	35.67±0.60	29.35±0.34	27.72±0.32	60.78±0.32	35.31±0.42	30.19±0.43	33.70±0.40
A-MEM	54.35±0.29	39.61±0.37	31.25±0.53	34.40±0.32	51.95±0.42	41.51±0.40	34.03±0.51	37.45±0.34
LightMem	70.13±0.41	41.88±0.42	29.61±0.59	29.44±0.71	75.58±0.50	43.72±0.45	30.49±0.34	28.00±0.42
Mem0	47.08±0.80	45.77±0.17	35.64±0.43	33.93±0.15	52.66±0.59	56.43±0.80	44.55±0.40	33.55±0.27
MemoryOS	59.87±0.35	42.87±0.52	33.99±0.32	35.72±0.79	62.08±0.59	39.94±0.24	36.32±0.57	29.74±0.21
FOCUS	75.97±0.12	51.82±0.34	40.60±0.40	38.50±0.31	77.86±0.30	61.98±0.41	52.50±0.23	39.63±0.62

FOCUS achieves the highest or tied highest score in eight of the nine task and temporal combinations, with the remaining result ranking second on Remembering Quarterly. A similar pattern appears with DeepSeek-V4-Flash, where FOCUS again ranks highest or tied highest in eight combinations and ranks second on Recommending Weekly. The gains are distributed across all three task types and temporal settings, with particularly clear improvements on Recommending Monthly under Qwen and Reasoning Weekly and Monthly under DeepSeek. FOCUS also remains competitive on Remembering as the interaction history extends to the Quarterly setting. These results show that the overall Memora gains are consistently reflected across different memory tasks and interaction lengths under both backbone models.

C.2 Parameter Analysis

We examine the sensitivity of FOCUS to three key parameters by varying each independently while keeping the remaining settings fixed. As shown in Table 4, performance remains relatively stable across most configurations. Changing the decay half-life H_{idle} from 30 to 15 or 60 results in a maximum difference of 1.10 points on LoCoMo, while Memora FAMA changes by no more than 0.73 points across all three settings. These results suggest that FOCUS is relatively insensitive to the natural decay rate within the tested range.

Variations in K_G and K_R also have a limited effect on Memora, with FAMA decreasing by at most 1.46 points across all three settings. LoCoMo shows greater sensitivity to K_G, particularly when it increases to 20, resulting in a 3.50 point decline in accuracy. All other tested configurations differ from the default by at most 1.62 points on LoCoMo. The default configuration achieves the highest FAMA across all Memora settings and remains close to the highest LoCoMo accuracy, indicating generally stable performance within the tested parameter ranges.

C.3 Reliability of Experimental Results

We further examine the reliability of the main results through independent run variability and paired bootstrap analysis. Table 5 reports the mean and standard deviation over three independent runs, while Table 6 presents 95% bootstrap confidence intervals for overall performance and the paired

Table 6: Overall performance and paired differences between FOCUS and each baseline on LoCoMo and Memora. Brackets indicate 95% percentile confidence intervals from 10,000 cluster bootstrap resamples, using the mean over three runs. Underlining marks the highest-scoring baseline within each backbone and benchmark.

Method	LoCoMo Accuracy (%)		Memora FAMA (%)	
	Overall score [95% CI]	FOCUS − method [95% CI]	Overall score [95% CI]	FOCUS − method [95% CI]
Qwen3-30B-A3B-Instruct				
NativeRAG	62.53 [57.39, 67.56]	+13.44 [7.69, 19.70]	30.91 [27.75, 32.44]	+12.73 [8.97, 14.30]
A-MEM	54.35 [50.71, 58.62]	+21.62 [17.27, 25.53]	35.09 [31.89, 37.58]	+8.55 [4.99, 10.08]
LightMem	<u>70.13</u> [67.81, 72.41]	+5.84 [2.77, 9.28]	33.64 [31.65, 35.08]	+10.00 [6.74, 11.06]
Mem0	47.08 [43.27, 50.26]	+28.90 [24.92, 33.17]	<u>38.45</u> [34.27, 38.73]	+5.19 [3.14, 8.57]
MemoryOS	59.87 [56.59, 63.73]	+16.10 [11.70, 19.60]	37.53 [35.52, 38.84]	+6.11 [2.46, 8.16]
FOCUS	**75.97** [74.03, 78.27]	–	**43.64** [40.51, 44.75]	–
DeepSeek-V4-Flash				
NativeRAG	60.78 [55.02, 66.14]	+17.08 [10.67, 23.71]	33.07 [31.69, 36.33]	+18.30 [11.89, 18.86]
A-MEM	51.95 [47.99, 56.78]	+25.91 [19.47, 31.41]	37.66 [35.90, 39.44]	+13.71 [9.10, 14.69]
LightMem	<u>75.58</u> [72.85, 78.75]	+2.28 [0.82, 5.29]	34.07 [29.79, 34.46]	+17.30 [14.68, 20.18]
Mem0	52.66 [48.70, 56.35]	+25.19 [19.91, 30.85]	<u>44.84</u> [42.26, 47.97]	+6.53 [4.17, 10.48]
MemoryOS	62.08 [60.01, 64.66]	+15.78 [11.45, 20.38]	35.33 [32.38, 36.90]	+16.04 [15.20, 20.64]
FOCUS	**77.86** [74.14, 81.53]	–	**51.37** [47.05, 51.68]	–

differences between FOCUS and each baseline. These analyses examine the consistency of the reported scores and the statistical support for the observed performance gains.

As shown in Table 5, FOCUS exhibits limited variation across independent runs, with standard deviations ranging from 0.12 to 0.62 points across all settings. On LoCoMo, the standard deviations are 0.12 points with Qwen and 0.30 points with DeepSeek. The results on Memora show similarly small fluctuations across the Weekly, Monthly, and Quarterly settings under both backbones. These results support the consistency of FOCUS across independent runs and interaction histories of different lengths.

To examine the statistical support for the performance gains, we perform 10,000 paired cluster bootstrap resamples using the same three run aggregation as in the main results. Conversations serve as clusters for LoCoMo and personas for Memora. As shown in Table 6, the 95% confidence intervals for the gains over the strongest baselines are [2.77, 9.28] with Qwen and [0.82, 5.29] with DeepSeek on LoCoMo. On Memora, the corresponding intervals are [3.14, 8.57] and [4.17, 10.48]. All paired confidence intervals remain above zero, providing statistical support for the performance gains of FOCUS across both benchmarks and backbone models.

D Prompts

D.1 Memory Construction Prompts

Long-Term Memory Extraction Prompt

You are a precise **long-term memory extractor** for conversations between a user and an assistant. Process every supplied message in ascending `source_id` order and extract all meaningful, explicitly supported information that may be useful later.

Extraction Rules.

1. Read both user and assistant messages, but distinguish the speaker who stated something from the person or entity the statement is about. An assistant's suggestion is not a user preference, decision, experience, or plan unless the user explicitly confirms it.

2. Preserve exact details. Never generalize away or silently omit full names, roles, project titles, descriptions, organizations, locations, dates, times, durations, milestones, quantities, amounts, currencies, units, or named items.
3. Preserve complete lists. Emit each independently useful list member, todo, preference, recommendation, requirement, project field, or scheduled item as an atomic standalone fact, while retaining the parent person, project, list, or activity needed to understand it.
4. Preserve status and chronology exactly, including planned, active, pending, completed, deleted, cancelled, rejected, replaced, corrected, and no-longer-preferred states. Do not silently overwrite an earlier state during extraction; extract the later update as its own explicit fact.
5. Keep distinct facts separate. Do not combine unrelated people, projects, attributes, events, times, or list members into a vague summary. Do not create duplicate paraphrases of the same fact.
6. Perform only light contextual completion needed to make each fact independently understandable. Resolve pronouns only when the referent is clear from the supplied messages. Do not invent missing facts, user preferences, causal links, or completion status.
7. Each item must use the `source_id` of the message that directly supports it. Skip only greetings, acknowledgements, filler, and content with no factual or memory value.
8. Return JSON only, with no markdown, commentary, or extra keys.
Output Schema.

```
{
  "data": [
    {
      "source_id": <source_id>,
      "fact": "<atomic standalone fact preserving all specific
details>"
    }
  ]
}
```

Input Fields.
`[CONVERSATION]`
`{conversation}`

Memory Incorporation Prompt

You are performing conservative **FOCUS memory incorporation**. Process each item independently and compare its `new_memory` only with candidates in the same item. Each `candidate_index` is local and 1-based.
The `speaker_id` identifies who supplied a fact, not necessarily the person or entity described by it. Infer entities, relation direction, time, and conditions from the supplied facts and context. Do not assume that facts from the same speaker describe the same entity, or that facts from different speakers describe different entities.
Action Rules.

- **IGNORE:** discard the new memory because it is redundant with an existing candidate, too vague, non-durable, or contains no additional fact worth storing. Do not select a candidate and do not preserve the new memory.
- **ADD:** the new fact is an independent fact or event that should coexist with all candidates. Distinct dated observations, quantities, transactions, activities, meetings, plans, preferences, goals, and list items normally coexist. Do not select a candidate.
- **UPDATE:** select one candidate only when both facts describe the same real-world fact or event and the new fact adds compatible detail. new rewritten text must be one standalone fact preserving every supported person and entity name, date, time, quantity, amount, unit, list member, relation direction, condition, event status, and other concrete detail from both facts. The selected old version is removed from retrieval and replaced by this merged canonical version. If a complete lossless merge is not possible, choose ADD instead.
- **REWRITE:** select one candidate only when the new fact explicitly corrects, retracts, deletes, completes, cancels, replaces, or states that the candidate is no longer current. old rewritten text must rewrite the old memory as an accurate historical or superseded state without presenting it as current. new rewritten text must state the new valid state unambiguously. Both texts must remain independently understandable and grounded in the supplied evidence.

Decision Policy. Prefer ADD when two facts can coexist, when they refer to different observations or events, or when merging them would blur chronology or lose detail. Shared topics, projects, activities, or preference domains do not make two facts the same. Use UPDATE only for compatible details of the same fact or event, and REWRITE only when an explicit change invalidates or replaces the earlier state. Select at most one candidate. Do not invent facts, identifiers, conditions, or explanations.
Return exactly one result for every `item_index`, preserving ascending order. Copy only the supplied `item_index` and selected local `candidate_index`. Output JSON only, with no markdown, explanations, extra keys, omitted objects, or reordered objects.

Items.

```
[{"item_index": 1,
  "new_memory": "<NEW_MEMORY>",
  "new_context": "<NEW_MEMORY_CONTEXT>",
  "source_timestamp": "<NEW_SOURCE_TIMESTAMP>",
  "speaker_id": "<NEW_SPEAKER_ID>",
  "candidates": [{
    "candidate_index": 1,
    "old_memory": "<OLD_MEMORY>",
    "old_context": "<OLD_MEMORY_CONTEXT>",
    "source_timestamp": "<OLD_SOURCE_TIMESTAMP>",
    "speaker_id": "<OLD_SPEAKER_ID>"
  }]
}]
```

Output.

```
{
  "results": [
    {
      "item_index": 1,
      "action": null,
      "candidate_index": null,
      "rewritten_text": null
    }
  ]
}
```

For **IGNORE** and **ADD**, `candidate_index` and `rewritten_text` must be null. For **UPDATE** and **REWRITE**, `candidate_index` must identify one supplied candidate and `rewritten_text` must be a non-empty standalone fact grounded only in the supplied facts and context.

Input Fields.

```
[ITEMS]
{items}
```

D.2 Memory Management Prompts

Used-Memory Attribution Prompt

You are performing **FOCUS used-memory attribution**.

Given a generated response and the retrieved candidate memories, identify which candidate memories actually contributed information to the response. A memory is considered used only when information from that memory is reflected in or clearly supports specific content in the generated response.

Rules.

- Select only integer indices listed in the candidate memories.
- Select a memory when its factual content is explicitly reflected in the response or clearly supports a specific statement in the response.
- Do not select a memory merely because it is topically related, semantically similar, consistent with the response, or useful as background information.
- A retrieved memory may be relevant but still unused; leave it unselected in that case.
- Do not infer hidden reasoning or unsupported dependencies.
- Output JSON only, with no markdown, explanation, or extra keys.

Input Format.

```
[GENERATED RESPONSE]
<MODEL_RESPONSE>


[RETRIEVED MEMORIES]
[{"index": 1, "text": "<RETRIEVED_MEMORY_1>"},
 {"index": 2, "text": "<RETRIEVED_MEMORY_2>"}]
```

Output Format.

```
{"used_indices": [1]}
```

Input Fields.

```
[GENERATED RESPONSE]
{model_response}


[RETRIEVED MEMORIES]
{retrieved_memories}
```

Memory Group Summary Prompt

Summarize the following related memories into one compact paragraph.

The summary serves as a retrieval index for deciding whether the full memory group should be searched. It should capture the main people, entities, events, relationships, attributes, preferences, projects, and temporal information that distinguish this group and may help match future queries.

Rules.

- Write one natural-language paragraph of at most three sentences and 100 words.
- Preserve exact person and entity names and the concrete details needed to retrieve this group, including relevant events, relationships, projects, attributes, preferences, and time information.
- Merge only genuinely repeated information. Keep distinct events, states, or time periods distinguishable when they matter for retrieval.
- Use `speaker_name` only as context for interpreting who supplied each memory; do not treat it as the person or entity described by the memory unless the memory text clearly supports that interpretation.
- Do not enumerate every memory or produce a keyword list.
- Do not add unsupported facts, explanations, labels, headings, or commentary.
- Use the same language as the memories.
- Output only the summary text. Do not output JSON or Markdown.

Group Members.

```
[{"text": "<GROUP_MEMORY_1>", "speaker_name": "<SPEAKER_NAME>"},
 {"text": "<GROUP_MEMORY_2>", "speaker_name": "<SPEAKER_NAME>"}]
```

Input Fields.

```
[GROUP MEMBERS]
{group_members}
```

D.3 Evaluation Prompts

LoCoMo LLM-as-a-Judge Prompt

Your task is to label an answer to a question as 'CORRECT' or 'WRONG'. You will be given the following data:

(1) a question (posed by one user to another user),
(2) a 'gold' (ground truth) answer,
(3) a generated answer

which you will score as CORRECT/WRONG.

The point of the question is to ask about something one user should know about the other user based on their prior conversations.

The gold answer will usually be a concise and short answer that includes the referenced topic, for example:

```
Question: Do you remember what I got the last time I went to Hawaii?
Gold answer: A shell necklace
```

The generated answer might be much longer, but you should be generous with your grading—as long as it touches on the same topic as the gold answer, it should be counted as CORRECT.

For time related questions, the gold answer will be a specific date, month, year, etc. The generated answer might be much longer or use relative time references (like "last Tuesday" or "next month"), but you should be generous with your grading—as long as it refers to the same date or time period as the gold answer, it should be counted as CORRECT. Even if the format differs (e.g., "May 7th" vs "7 May"), consider it CORRECT if it's the same date.

Now it's time for the real question:

```
Question: {question}
Gold answer: {gold_answer}
Generated answer: {generated_answer}
```

First, provide a short (one sentence) explanation of your reasoning, then finish with CORRECT or WRONG.

Do NOT include both CORRECT and WRONG in your response, or it will break the evaluation script.

Just return the label CORRECT or WRONG in a json format with the key as "label".

Memora LLM-as-a-Judge Prompt

[System]

You are an expert evaluator assessing AI assistant responses. Your task is to answer a YES/NO evaluation question about a given response.

You must provide your answer in the following JSON format:

```
{
  "answer": "yes" or "no",
```

```
  "confidence":  0.0 to 1.0,
  "explanation":  "Brief explanation of your reasoning"
}
```

Be objective and thorough in your evaluation.

[User]

Please evaluate the following AI response against the evaluation question.

Input Format.

```
AI RESPONSE TO EVALUATE:
<ANSWER>

EVALUATION QUESTION:
<EVALUATION_QUESTION>
```

Provide your evaluation in JSON format with answer (yes/no), confidence (0.0–1.0), and explanation.

Output Format.

```
{
  "answer":  "yes" or "no",
  "confidence":  0.0 to 1.0,
  "explanation":  "Brief explanation of your reasoning"
}
```