NeuroMem: A Neuroplastic Memory Framework for Lifelong Agents through Delayed Consolidation

Yingyi Cheng[1,2]* **Xueqiang Han**[1,2]* **Chen Yu**[1,2]†
[1]School of Computer Science and Technology
Huazhong University of Science and Technology
[2]Hubei JJJC BigData Centre
{leocheng_1,hanxq,yuchen}@hust.edu.cn

Abstract

With long-term memory, LLM-based agents have demonstrated remarkable capabilities in handling complex long-horizon tasks. However, existing memory frameworks primarily treat long-term memory as a post-admission management problem, emphasizing how experiences should be organized, updated, or controlled. This overlooks a more fundamental stage in the memory lifecycle, i.e., how transient experiences are progressively transformed into stable and task-transferable knowledge. Biological memory consolidation provides a temporal principle for memory formation, whereby long-term memory emerges through the delayed stabilization and selective transformation of recent experience, rather than through its verbatim preservation. Motivated by this, we propose NeuroMem, a neuroplastic memory framework for lifelong agents through delayed consolidation. NeuroMem formulates the formation of task-transferable knowledge in long-term memory as a delayed-reward reinforcement learning problem. Specifically, newly formed memory candidates are first maintained in a transient buffer, where a consolidation policy learns to promote, merge, retain, or discard them only after their delayed utility rewards become observable through subsequent interactions. Experiments on EHRSQL and LoCoMo show NeuroMem achieves stronger performance with fewer persistent entries than most baselines, improving EHRSQL task success by up to **8.61%** and LoCoMo F1/BLEU-4 by up to **+5.24%/+1.75%** over baselines. Our code is available on https://github.com/leoCHENG100/NeuroMem.

1 Introduction

Large Language Models (LLMs) have recently empowered agents with reasoning, planning, and decision-making capabilities [1, 2, 3]. Long-term memory further extends these capabilities by allowing agents to retain and leverage past experiences across interactions [4, 5, 6]. Therefore, LLM-based agents are increasingly utilized to tackle long-horizon tasks, such as Deep Research [7], Manus [8], and OpenClaw [9]. However, relying heavily on long-term memory also introduces a critical challenge. Recent studies reveal that memory-augmented LLM agents are susceptible to an "experience-following" bias, where LLM agents tend to replicate past actions with retrieved memory [10]. This behavior highlights a fundamental limitation, i.e., replaying flawed or irrelevant experiences causes error propagation and degrades the agent's long-horizon performance [10, 11].

To mitigate these, recent frameworks have shifted external memory from passive storage toward active memory management. One prominent line of work focuses on memory representation construction. Frameworks like Mem0 [12] convert raw interactions into memory records by preserving salient

*Equal contribution.
†Corresponding author.

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

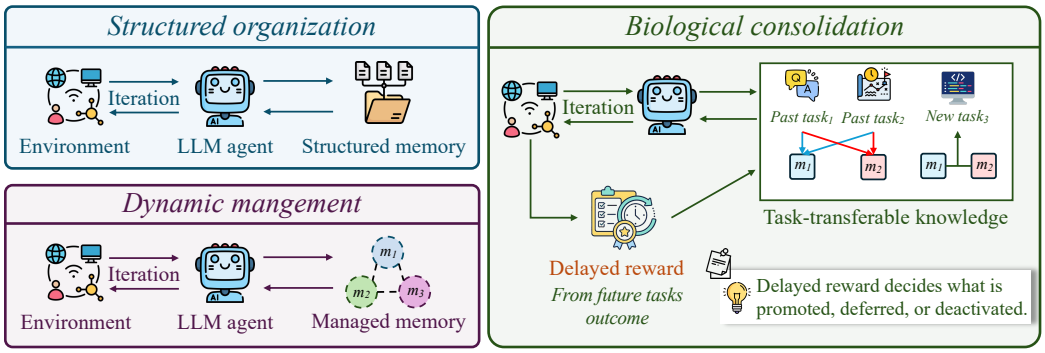


Figure 1: From experience storage to delayed consolidation guided by future task outcomes.

information while compressing redundant details [13, 14, 15]. Beyond how individual memories are formed, another line of work studies relational memory organization, where stored experiences are connected and refined through structural associations. Architectures such as A-MEM [16] achieve this by transforming new memories into structured notes and agentically linking them with related past memories [17, 18, 19]. A more recent line introduces learned memory operation control. Approaches like Memory-R1 [20] replace fixed heuristics with reinforcement learning policies for memory editing and use [21, 22, 23].

Despite these efforts, existing memory frameworks still focus on how experiences should be organized, updated, and controlled. However, long-term memory for a lifelong agent should not merely preserve episodic experiences, but should accumulate knowledge that remains useful beyond the local interaction. Biological memory suggests that recent experience is not directly imprinted into long-term memory, but selectively stabilized and integrated into existing knowledge through delayed consolidation [24]. However, designing such a consolidation mechanism for lifelong agents is non-trivial because the utility of a candidate memory is often sparse, delayed, and only observable through its later contribution across interactions. Existing RL-based memory frameworks mainly optimize operation policies for adding, updating, or deleting memory, leaving trace-level consolidation under delayed utility insufficiently addressed. Therefore, we ask a critical question: *how can a memory framework convert delayed utility into consolidation signals that transform transient experiences into stable and task-transferable long-term memory?*

To this end, we propose NeuroMem, a neuroplastic memory framework for lifelong agents through delayed utility-guided consolidation. Here, "neuroplastic" refers to a functional property of the memory lifecycle. NeuroMem casts the formation of task-transferable knowledge in long-term memory as a trace-level delayed-reward reinforcement learning problem. Newly formed memory candidates are temporarily buffered, evaluated by delayed utility signals from later interactions, and selectively consolidated into persistent memory through a learned consolidation policy. This process enables NeuroMem to distill transient experiences into compact and transferable long-term memory, rather than accumulating raw interactions. Experiments demonstrate the effectiveness of NeuroMem across two widely-used benchmarks, EHRSQL [25] and LoCoMo [26]. On EHRSQL, NeuroMem improves task success over the strongest prior memory baseline by **+6.03%** under DeepSeek-v3.2 and **+8.61%** under Gemini-3-Flash, while reducing the final persistent entry count from 242 to 64 and from 210 to 54. On LoCoMo, NeuroMem achieves the best overall F1/BLEU-4 under both backbones, with gains of **+4.20%/+1.75%** under Gemini-3-Flash and **+5.24%/+1.18%** under DeepSeek-v3.2.

2 Related Work

2.1 Structured Memory Retrieval for Long-Horizon Reasoning

Retrieval serves as the primary interface that allows stored experience to shape the behavior of memory-augmented LLMs. Early approaches often follow the RAG-style formulation, retrieving the top-k most similar entries for downstream generation [27, 28]. This paradigm is effective when the required evidence is localized, but becomes less suitable as the interaction horizon extends. In

long-horizon memory, useful support is often distributed across past experiences: one record may contain a constraint, another may encode a procedure, and another may provide grounding context. Top-k retrieval can expose relevant fragments, but it does not determine whether they jointly form a task-specific support structure. Recent memory frameworks therefore introduce more structured retrieval mechanisms. MemoChat [29] retrieves from self-composed memos of past interactions through memorization–retrieval–response cycles to maintain long-range dialogue consistency. O-Mem [30] performs layered memory access through active user profiling, retrieving persona attributes and topic-related context for personalized response generation. Similarly, H-MEM [31] organizes memory in multiple abstraction levels and routes queries from high-level summaries to more specific episodic memories. These methods show that long-horizon retrieval is not merely matching a query against isolated entries; their common contribution is to make accumulated memory more structured, searchable, and selectively exposed to the model.

However, organizing or filtering memory before generation does not fully address the compositional capability required for long-horizon reasoning. Memories are usually selected according to local relevance before their joint contribution is evaluated, which can miss entries that are weak standalone matches but useful as bridges, constraints, or grounding signals within a composed support path. Search-based reasoning methods such as Tree of Thoughts [32] and MCTS-style exploration [33] show that deliberate search can compare alternative reasoning paths, but their search space is generated intermediate thoughts rather than stored memory structures. This leaves a gap between structured memory access and compositional memory retrieval: long-horizon agents need retrieval mechanisms that can compare alternative combinations of stored knowledge and assemble task-specific support before generation. Such a retrieval view is especially important when a compact set of task-transferable knowledge must be recombined across different future tasks.

2.2 Memory Frameworks and Adaptive Memory Control

Prior work on agent memory has developed a range of mechanisms for organizing long-term memory over extended interactions. MemGPT [14] introduces an operating system-inspired hierarchical memory manager, while MemoryBank [34] models temporal decay through the Ebbinghaus Forgetting Curve. More recent systems such as Mem0 [12] and A-Mem [16] advance memory design through structured storage and dynamic organization. Beyond architectural design, prior work has also explored summarization, compression, pruning, and heuristic write policies to control memory growth and improve retrieval efficiency [12, 35]. More recently, reinforcement learning has been explored to learn memory-related operations, providing a learning-based alternative to fixed write, update, and retention policies [20].

However, many existing approaches maintain memory based primarily on local evidence, such as write-time importance, heuristic scores, or relevance to the current query. While such signals are useful for organizing memory and controlling growth, they do not fully address the delayed credit-assignment problem inherent in memory management. A memory's long-term value is better assessed by whether it contributes to successful reasoning after retrieval and composition. Without this mechanism, noisy or weakly reusable memories may remain active, increasing pressure on both retrieval and memory control. This motivates a lifecycle view of memory management, where memories are first kept provisional and are later reinforced, retained, or weakened through delayed credit-assignment, allowing repeatedly valuable experiences to be consolidated into task-transferable knowledge.

3 Methodology

3.1 Overview and Problem Formulation

The goal of NeuroMem is to learn a memory evolution mechanism that gradually transforms transient interaction experiences into compact task-transferable knowledge. Given a sequence of tasks $\mathcal{Q} = \{q_1, \ldots, q_T\}$, the agent maintains two memory repositories at each step t: a transient buffer $\mathcal{B}_t$ and a persistent memory $\mathcal{M}_t$. The transient buffer stores newly extracted memory candidates whose long-term utility has not yet been established, whereas the persistent memory stores consolidated knowledge that is expected to transfer across future tasks.

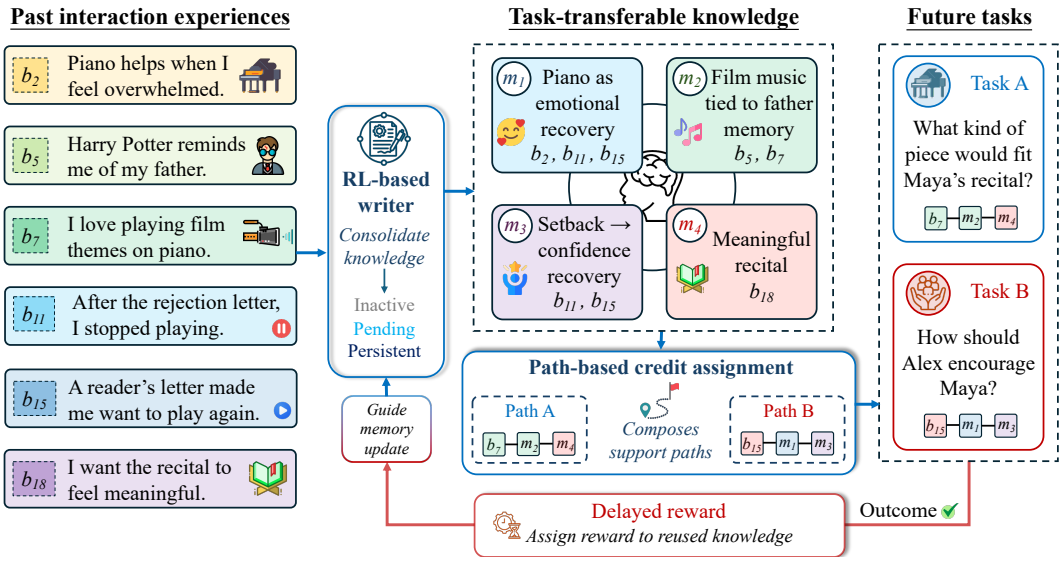


Figure 2: Overview of NeuroMem.

At each step t, retrieval is performed over the active memory set $\mathcal{B}_t \cup \mathcal{M}_t$, allowing provisional memories to support downstream reasoning before they are committed to long-term storage. Given a task query q_t, the agent retrieves a compositional support path $\mathcal{P}_t$ from $\mathcal{B}_t \cup \mathcal{M}_t$, generates a response y_t, observes an outcome o_t, and extracts a new candidate memory c_t from the interaction. Instead of immediately inserting c_t into $\mathcal{M}_t$, NeuroMem places it into $\mathcal{B}_t$, where it remains plastic until future interactions provide evidence of its downstream utility.

We formulate long-horizon memory management as a delayed optimization problem over memory evolution policies. The objective is to maximize cumulative task-level reward while controlling the growth and redundancy of persistent memory:

$$\max_{\theta} \; J(\theta) = \mathbb{E}_{\xi \sim p_\theta}\left[\sum_{t=1}^{T} (\rho_t - \lambda|\mathcal{M}_t| - \beta \mathrm{Red}(\mathcal{M}_t))\right], \tag{1}$$

where p_θ denotes the trajectory distribution induced by the consolidation policy, ρ_t is the task reward from the benchmark evaluator (binary execution success for EHRSQL; per-question token-level F1 normalized to $[0, 1]$ for LoCoMo), $|\mathcal{M}_t|$ penalizes persistent memory growth, and $\mathrm{Red}(\mathcal{M}_t)$ penalizes redundancy among consolidated memories. Rewards have different semantics and are not compared or aggregated across benchmarks. BLEU-4 and final persistent entry count are evaluation metrics only (Appendix A.3); no LLM judge or additional reward shaping is used. This formulation does not assume that the value of a memory candidate is known at the time of its formation. Instead, a candidate's utility is revealed only through subsequent retrieval, composition, and task outcomes.

3.2 Delayed Memory Consolidation

To optimize the delayed memory objective in Eq. 1, NeuroMem instantiates the memory evolution mechanism as a consolidation policy. The policy operates on each buffered candidate memory block and decides whether to promote, merge, retain for further observation, or discard it.

For each buffered candidate $b_i \in \mathcal{B}_t$, we construct a consolidation state

$$z_t(b_i) = \left[b_i,\ \mathrm{age}_t(b_i),\ \mathrm{rel}_t(b_i, \mathcal{M}_t)\right], \tag{2}$$

where $\mathrm{age}_t(b_i)$ is the elapsed time since the candidate was formed. Let f_{BGE} be the frozen `BAAI/bge-m3` encoder [36] and $r(x)$ the serialized text of trace x. The relation score is the maximum cosine similarity to existing persistent memories:

$$e(x) = \frac{f_{\mathrm{BGE}}(r(x))}{\|f_{\mathrm{BGE}}(r(x))\|_2}, \qquad \mathrm{rel}_t(b_i, \mathcal{M}_t) = \begin{cases} 0, & \mathcal{M}_t = \varnothing, \\ \max_{m_j \in \mathcal{M}_t} e(b_i)^\top e(m_j), & \text{otherwise.} \end{cases} \tag{3}$$

The consolidation policy is defined as $a_t(b_i) \sim \pi_\theta(\cdot \mid z_t(b_i))$, with the action space

$$a_t(b_i) \in \{\textsc{Promote}, \textsc{Merge}, \textsc{Retain}, \textsc{Discard}\}. \tag{4}$$

Under sparse delayed feedback, we limit low-evidence retrieval and promotion using utility gates inspired by RL action masking and shielding [37, 38]. The pending retrieval minimum utility controls provisional-trace retrieval eligibility; the promotion score threshold and promotion minimum candidate utility control PROMOTE admissibility; review delay and maximum transient-buffer age control review timing (Table 5). At each review, π_θ selects among admissible actions.

Each action induces a different transition of the transient buffer and persistent memory. PROMOTE moves b_i from $\mathcal{B}_t$ to $\mathcal{M}_t$ as a new persistent memory. MERGE integrates b_i into an existing related memory in $\mathcal{M}_t$ when it provides overlapping but useful evidence. RETAIN keeps b_i in $\mathcal{B}_t$ for further delayed observation. DISCARD removes b_i when it is insufficiently useful, unreliable, or redundant. The resulting memory transition is written as

$$(\mathcal{B}_{t+1}, \mathcal{M}_{t+1}) = \mathcal{T}_{\pi_\theta}(\mathcal{B}_t, \mathcal{M}_t, b_i, a_t(b_i)). \tag{5}$$

The consolidation policy is trained to favor actions that yield high delayed utility while discouraging unnecessary memory growth and redundancy. For a candidate memory b_i and action $a_t(b_i)$, we define the regularized delayed return as

$$\widetilde{G}_t(b_i, a_t) = G_t(b_i, a_t) - \lambda\Delta|\mathcal{M}_t| - \beta\Delta\mathrm{Red}(\mathcal{M}_t), \tag{6}$$

where $G_t(b_i, a_t)$ denotes the unregularized action-conditioned delayed utility accumulated from future task rewards (Eq. 15), $\Delta|\mathcal{M}_t|$ is the increase in persistent entry count induced by the action, and $\Delta\mathrm{Red}(\mathcal{M}_t)$ is the resulting increase in redundancy. Equations 8–14 introduce the support-path attribution and successor-trace quantities required to compute G_t.

In practice, we use a direct score-function (REINFORCE-style) policy-gradient estimator for the four-action consolidation policy, focusing on action-conditioned delayed credit from future memory reuse:

$$\nabla_\theta J(\theta) = \mathbb{E}\left[\nabla_\theta \log \pi_\theta(a_t(b_i) \mid z_t(b_i)) \left(\widetilde{G}_t(b_i, a_t) - \mu_t\right)\right], \tag{7}$$

where μ_t is a non-learned baseline used for variance reduction. Let R_n denote the n-th finalized regularized delayed return in an independent benchmark–backbone–seed run. At that update, $\mu_t = 0$ for $n = 1$ and $\mu_t = \frac{1}{n-1}\sum_{j=1}^{n-1} R_j$ otherwise. We update the policy with online Adam (learning rate 0.03) using $R_n - \mu_t$, then incorporate R_n into the mean. On LoCoMo, this mean carries over with the policy when memory is reset between samples. This formulation applies reinforcement learning specifically to the consolidation boundary: NeuroMem does not learn generic memory-tool execution, but learns whether each provisional trace has accumulated sufficient delayed evidence to become part of persistent task-transferable knowledge.

3.3 Path-Based Credit Assignment

The consolidation policy requires an action-conditioned delayed return for each buffered memory candidate. However, task feedback is observed at the interaction level, whereas consolidation decisions are made at the level of individual memory traces. We therefore introduce path-based credit assignment, which converts interaction-level outcomes into trace-level delayed rewards.

At step t, retrieval is performed over the active memory set $\mathcal{X}_t = \mathcal{B}_t \cup \mathcal{M}_t$. Given query q_t, eligible traces are ranked by query-memory cosine similarity using the frozen `BAAI/bge-m3` encoder [36]. NeuroMem performs tree search by recursively extending ordered, non-repeating support paths over the candidate pool $\mathcal{C}_t \subseteq \mathcal{X}_t$. Each path candidate is a memory sequence $\mathcal{P} = (x_{i_1}, x_{i_2}, \ldots, x_{i_\ell}), \quad x_{i_j} \in \mathcal{C}_t$. Each generated and scored path extension consumes one unit of b, a global expansion budget that allows different path structures while bounding search cost. Search stops after b extensions or when no valid extension remains. The explored paths form $\mathcal{S}_t(q_t, \mathcal{X}_t)$, and the selected support path is obtained by

$$\mathcal{P}_t^* = \arg\max_{\mathcal{P} \in \mathcal{S}_t(q_t, \mathcal{X}_t)} \widehat{V}_t(\mathcal{P} \mid q_t). \tag{8}$$

Here, $\widehat{V}_t(\mathcal{P} \mid q_t)$ is a deterministic, embedding-based path score:

$$\widehat{V}_t(\mathcal{P} \mid q_t) = \frac{1}{\ell}\left[s(q_t, x_{i_1}) + \sum_{j=2}^{\ell} s(x_{i_{j-1}}, x_{i_j})\right], \tag{9}$$

where $s(x, y) = e(x)^\top e(y)$ uses the normalized embeddings in Eq. 3, with $r(x)$ denoting serialized query or trace text. Thus, $\widehat{V}_t(\mathcal{P} \mid q_t)$ favors paths whose first memory is relevant to the query and whose subsequent memories form a coherent semantic chain.

After the agent generates a response y_t and observes an outcome o_t, the outcome is converted into a scalar task reward ρ_t. To assign this reward to individual memories, we define the attribution weight of each memory $x \in \mathcal{X}_t$ as

$$A_t(x) = \begin{cases} \alpha_t(x, \mathcal{P}_t^*), & \text{if } x \in \mathcal{P}_t^*, \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

For a memory x_{i_j} on the selected path, we compute its attribution by normalized marginal path-value gain using this fixed score:

$$\delta_t(x_{i_j}, \mathcal{P}_t^*) = \max\left(0,\, \widehat{V}_t(\mathcal{P}_{t,1:j}^* \mid q_t) - \widehat{V}_t(\mathcal{P}_{t,1:j-1}^* \mid q_t)\right), \tag{11}$$

$$\alpha_t(x_{i_j}, \mathcal{P}_t^*) = \frac{\delta_t(x_{i_j}, \mathcal{P}_t^*)}{\sum_{k=1}^{\ell} \delta_t(x_{i_k}, \mathcal{P}_t^*) + \epsilon}. \tag{12}$$

Here, $\mathcal{P}_{t,1:j}^*$ denotes the prefix of the selected path up to position j, and we define $\widehat{V}_t(\mathcal{P}_{t,1:0}^* \mid q_t) = 0$. The constant ϵ prevents numerical instability when all marginal gains are zero.

To make the delayed return action-conditioned, we track how the information in a buffered candidate b_i persists after a lifecycle action. Let $s_t(b_i, a_t)$ denote the successor trace induced by action a_t:

$$s_t(b_i, a_t) = \begin{cases} b_i^{\text{prom}}, & a_t = \textsc{Promote}, \\ m_j^{\text{merge}}, & a_t = \textsc{Merge}, \\ b_i, & a_t = \textsc{Retain}, \\ \varnothing, & a_t = \textsc{Discard}, \end{cases} \tag{13}$$

where b_i^{prom} is the promoted persistent memory, m_j^{merge} is the persistent memory after merging b_i into a related memory $m_j \in \mathcal{M}_t$, and $\varnothing$ denotes that the candidate is removed. The action-conditioned attribution of b_i at a future step τ is then defined as

$$A_\tau^{(a_t)}(b_i) = \begin{cases} A_\tau(s_t(b_i, a_t)), & \text{if } s_t(b_i, a_t) \in \mathcal{X}_\tau, \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

For a lifecycle action a_t applied to candidate b_i at review step t, its delayed return is accumulated over a credit horizon H:

$$G_t(b_i, a_t) = \sum_{\tau=t+1}^{\min(T,\, t+H)} \gamma^{\tau-t} A_\tau^{(a_t)}(b_i) \rho_\tau, \tag{15}$$

where γ is a discount factor. This return increases when the information carried by b_i repeatedly contributes to successful future support paths after the action is applied, and remains low when it is unused, redundant, or removed.

In the analogy to biological consolidation, G_t captures delayed evidence of utility, and the transient buffer $\mathcal{B}_t$ holds provisional traces. The four lifecycle actions correspond to selective stabilization (PROMOTE), integration (MERGE), continued observation (RETAIN), and removal (DISCARD). This biological analogy is functional rather than mechanistic.

4 Experiments

We evaluate NeuroMem in two distinct long-horizon settings: EHRAgent [2] built on EHRSQL [25], and the long-term conversational memory benchmark LoCoMo [26]. These benchmarks differ in task structure and interaction history: EHRSQL requires agents to use past experiences for later EHR tasks, while LoCoMo requires agents to use long multi-session dialogue experiences for later questions over distant conversational evidence. This contrast allows us to examine whether delayed utility observed through later interactions can guide provisional experiences toward compact task-transferable knowledge in these two settings. Beyond main results, we conduct ablations and sensitivity analyses to analyze how this consolidation process affects performance and entry-level compactness.

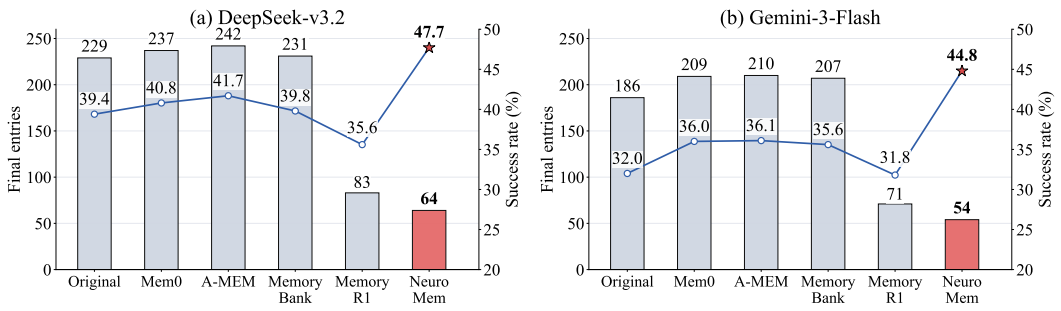


Figure 3: EHRSQL results on MIMIC-III. Bars denote final persistent entry count after all tasks, and lines denote task success rate. NeuroMem achieves the highest success rate under both backbones while retaining fewer persistent entries than prior memory baselines.

4.1 Experimental Setup

Datasets and Benchmarks. Our benchmark suite consists of EHRAgent [2] built on EHRSQL [25] and LoCoMo [26]. EHRAgent executes EHRSQL tasks on MIMIC-III (Medical Information Mart for Intensive Care III) [39], a de-identified critical-care electronic health record database. These benchmarks have been used in recent studies of long-term memory and memory management for LLM agents [10, 12, 16, 20], providing a common basis for comparison. EHRSQL evaluates delayed consolidation in structured, executable EHR tasks, while LoCoMo evaluates long-range conversational reasoning over distant evidence. Together, they test whether NeuroMem can form compact task-transferable knowledge in these two structurally distinct long-horizon settings. Detailed dataset descriptions are provided in Appendix A.1.

LLM Backbones and Baselines. We use DeepSeek-v3.2 [40] and Gemini-3-Flash [41] as backbone models. We compare against the original benchmark baselines, as well as representative memory-augmented baselines including Mem0 [12], A-MEM [16], MemoryBank [34], and Memory-R1 [20]. These baselines cover structured memory construction, agentic memory organization, and learning-based memory control. More details on the baseline methods are provided in Appendix A.2.

Metrics. For EHRSQL, we measure task success rate and final persistent entry count after all tasks, and report average turn time for the budget–efficiency analysis. For LoCoMo, we use F1, BLEU-4, and mean final persistent entry count across samples. Counts exclude transient-buffer traces. Merged entries may be longer, so fewer entries do not necessarily imply lower token, storage, or context costs. Detailed ablations and hyperparameter analysis are conducted on EHRSQL, whose structured task outcomes provide a controlled signal for examining when buffered experiences become useful in later tasks. This allows us to analyze how merge, retain, and discard decisions affect task performance and final entry count. We provide metric definitions in Appendix A.3.

Implementation Details. For the main results, all methods are evaluated under the same task order, benchmark split, and backbone configuration. NeuroMem uses hyperparameters selected on an EHRSQL development split, frozen before evaluation, and reused unchanged on LoCoMo (Appendix A). Additional task-order permutation results for NeuroMem with Gemini-3-Flash are reported in Appendix B.2.

4.2 Main Results

EHRSQL. Figure 3 presents the main EHRSQL results on the MIMIC-III evaluation set, with the Original agent included as the benchmark baseline. NeuroMem achieves the best task success rate with substantially fewer persistent entries. Compared with the strongest prior baseline, A-MEM, NeuroMem improves success from 41.65% to 47.68% under DeepSeek-v3.2 and from 36.14% to 44.75% under Gemini-3-Flash, while reducing final entry count from 242 to 64 and from 210 to 54, respectively.

EHRSQL stresses structured EHR task decisions, where later queries may reuse experiences related to clinical grounding, temporal constraints, field selection, and SQL operations. The simultaneous

Table 1: LoCoMo results under two backbones. Each cell reports F1/BLEU-4 (B4, %); Final Entries denotes mean final persistent entry count across samples. Bold and underline indicate the best and second-best results within each backbone.

Model	Method	Category												Final Entries
		Multi-Hop		Temporal		Open-Domain		Single-Hop		Adversarial		Overall		
		F1	B4	F1	B4	F1	B4	F1	B4	F1	B4	F1	B4	
DeepSeek-v3.2	LoCoMo	5.45	0.82	24.12	7.55	15.60	1.25	8.75	3.10	1.85	0.65	9.55	2.86	220.4
	A-MEM	11.38	1.52	35.82	12.50	19.85	2.48	16.74	6.12	5.06	**1.84**	16.59	5.36	207.5
	Mem0	7.29	1.48	36.55	11.18	**28.43**	2.25	12.76	4.57	2.85	0.92	14.36	4.27	198.4
	MemoryBank	10.22	1.35	33.25	11.46	24.30	2.07	15.04	5.65	4.32	1.51	15.34	4.88	205.2
	Memory-R1	8.42	1.15	31.40	10.25	21.30	1.85	11.85	4.10	3.12	1.05	13.02	3.88	**134.1**
	NeuroMem (Ours)	**14.86**	**2.31**	**37.15**	**13.05**	27.50	**2.65**	**24.80**	**8.45**	**8.40**	1.76	**21.83**	**6.54**	135.4
Gemini-3-Flash	LoCoMo	8.15	1.12	25.40	8.65	9.85	0.45	10.60	3.85	3.52	1.05	11.02	3.45	215.8
	A-MEM	14.57	1.80	48.26	**17.34**	12.45	1.45	38.47	17.42	4.97	1.43	27.88	10.83	203.2
	Mem0	10.20	1.49	40.36	11.23	14.24	0.56	17.36	5.16	11.40	4.58	18.57	5.27	196.6
	MemoryBank	12.84	2.21	33.78	11.45	**22.61**	0.79	15.24	5.71	9.48	3.34	16.96	5.37	208.4
	Memory-R1	11.25	1.65	32.15	10.85	13.50	0.65	14.80	5.05	8.45	3.10	15.61	4.85	132.8
	NeuroMem (Ours)	**17.50**	**2.24**	**49.35**	16.02	15.22	**1.99**	**42.15**	**19.86**	**13.50**	**5.20**	**32.08**	**12.58**	**123.3**

improvement in success and reduction in entry count suggests that NeuroMem does not gain by accumulating more task-specific records. Instead, delayed utility-guided consolidation filters provisional experiences after their value becomes observable in later interactions, preserving those that repeatedly support structured task decisions as compact task-transferable knowledge.

LoCoMo. Table 1 reports the full LoCoMo results, including the original LoCoMo memory setting as the benchmark baseline. NeuroMem achieves the best overall F1/BLEU-4 under both backbone models, improving the strongest prior overall score from 16.59/5.36 to 21.83/6.54 with DeepSeek-v3.2 and from 27.88/10.83 to 32.08/12.58 with Gemini-3-Flash. It also retains fewer persistent entries than LoCoMo, Mem0, A-MEM, and MemoryBank, while remaining close to Memory-R1 under DeepSeek-v3.2.

LoCoMo extends the evaluation to long multi-session conversations, where later questions may depend on distant conversational evidence such as personal facts, temporal anchors, relational cues, and preference constraints. These results suggest that NeuroMem does not rely on preserving more dialogue fragments. Instead, delayed utility-guided consolidation filters conversational noise and consolidates provisional experiences that later prove useful into compact task-transferable knowledge. This result complements EHRSQL, providing evidence for delayed consolidation across the two evaluated settings.

Cross-Benchmark Analysis. The two benchmarks reveal different backbone strengths: DeepSeek-v3.2 performs better on the structured EHRSQL agent task, whereas Gemini-3-Flash is stronger on LoCoMo, where later questions depend on long conversational histories. Under one unchanged shared configuration, NeuroMem improves overall performance on the two evaluated benchmarks with both backbones.

The results also show that entry-level compactness alone is not sufficient. Under DeepSeek-v3.2 on LoCoMo, Memory-R1 reaches the smallest mean final entry count of 134.1 but obtains 13.02 overall F1, while NeuroMem keeps a nearly identical count of 135.4 and improves F1 to 21.83. This indicates that the benefit comes not from reducing entry count by itself, but from consolidating provisional experiences into compact task-transferable knowledge after their delayed utility becomes observable. The main exception is Open-Domain F1, where NeuroMem trails the strongest baseline by 0.93 and 7.39 points under DeepSeek-v3.2 and Gemini-3-Flash, respectively. Rare details may receive insufficient delayed credit before review or lose specificity through merging, suggesting a coverage–selectivity trade-off.

4.3 Ablation study

Table 2 summarizes the ablation results on the EHRSQL agent setting. We examine whether the improvement comes from learned delayed consolidation, path-based credit assignment, or merely

Table 2: Ablation analysis on EHRSQL. We isolate the effects of learned delayed consolidation, path-based credit assignment, and adaptive consolidation. The gain denotes the improvement of full NeuroMem over each ablated variant.

Analysis	Variant	Credit	Policy	Success ↑	Gain
Delayed Consolidation	w/o RL Consolidation	Path-based	None	30.98%	+13.77
Credit Assignment	w/o Path-Based Credit	Similarity	Learnable	32.53%	+12.22
Policy Adaptivity	NeuroMem (static)	Path-based	Static	28.23%	+16.52
Full Method	NeuroMem	Path-based	Learnable	**44.75%**	–

fixed consolidation rules, while keeping the same path scorer and search procedure. The *w/o RL Consolidation* variant removes the learnable consolidation policy and instead uses a fixed high-threshold rule to manage the persistent memory, while still keeping path-based credit signals available. Its success rate drops to 30.98%, indicating that simply imposing a strict admission rule cannot replace learned delayed consolidation. Although a high threshold can prevent some noisy traces from entering persistent memory, it cannot adaptively determine when a provisional trace should be promoted, merged, retained, or discarded based on delayed utility observed in later interactions. The *w/o Path-Based Credit* variant keeps a learnable consolidation policy but replaces path-based credit assignment with similarity-based credit. It reaches 32.53%, showing that local similarity is a weak proxy for long-term memory value: a trace may be only weakly similar to a future query, yet still contribute as a bridge, constraint, or grounding signal within a successful compositional support path. In contrast, full NeuroMem achieves 44.75%, suggesting that delayed consolidation is most effective when future task outcomes are attributed through the actual memory paths used for reasoning.

To rule out the possibility that the gain comes from fixed consolidation heuristics, NeuroMem (static) keeps the same memory actions and hyperparameter settings as full NeuroMem, but removes the adaptive policy. The preset consolidation thresholds and timing rules (Appendix A.4) determine each action directly throughout evaluation, without policy-gradient updates. The static variant achieves a success rate of 28.23%, 16.52 points below full NeuroMem. This result suggests that the consolidation boundary cannot be reliably determined by static thresholds alone, because the utility of a provisional trace is revealed only through later interactions and may change as the memory state evolves. Both non-learned variants also fall below Original, possibly because incorrect merges or discards remove information useful in later tasks. These variants remove or replace the learned policy, so the results do not directly measure under-training. A shuffled-feedback sanity check in Appendix B.1 further shows that randomly reassigning feedback to memory updates hurts performance, supporting that NeuroMem benefits from task-grounded delayed utility rather than online updates alone. Removing either the memory-growth or redundancy penalty also lowers success and increases final persistent entry count (Appendix B.3). Overall, the ablation shows that NeuroMem improves long-horizon memory management by coupling provisional storage with state-dependent policy learning and aligned delayed credit, enabling provisional traces to become compact task-transferable knowledge only after their future utility is observed.

4.4 Hyperparameter Sensitivity and Budget–Efficiency Analysis

We first conduct a hyperparameter sensitivity analysis on EHRSQL, with full results reported in Appendix A.4. The analysis covers consolidation thresholds, path-based credit parameters, support-path search budgets, and transient-buffer access thresholds. Across these groups, the default NeuroMem configuration consistently achieves the strongest success rate, suggesting that the gain is not driven by a single favorable threshold or isolated tuning choice. Both shorter ($H = 16$) and longer ($H = 64$) horizons reduce EHRSQL success relative to $H = 32$ (Table 6), suggesting that temporal credit requires calibration. This does not establish that increasing H improves LoCoMo Open-Domain F1.

Table 3 further analyzes the trade-off between support-path search budget and execution efficiency on EHRSQL at two operating points rather than providing a complete scaling curve. The fixed-threshold results come from a separate experiment from the ablation in Table 2. For the fixed-threshold consolidation variant, increasing the search budget from b=512 to b=3072 increases success only from 30.29% to 30.64%, while average turn time increases from 67.13s to 86.20s. This suggests that

Table 3: Budget–efficiency trade-off on EHRSQL. For methods with compositional support-path retrieval, b counts generated and scored path extensions. Fixed-threshold Consolidation removes the learnable consolidation policy and manages memory with a fixed high-threshold rule.

Method	Budget b	Success Rate (%) ↑	Avg Turn Time (s) ↓
Original	–	32.01	70.50
A-MEM	–	36.14	135.20
Fixed-threshold Consolidation	512	30.29	67.13
Fixed-threshold Consolidation	3072	30.64	86.20
NeuroMem	512	38.73	92.30
NeuroMem	3072	**44.75**	126.15

larger search alone is insufficient when memory evolution is governed by non-adaptive consolidation rules. In contrast, NeuroMem improves from 38.73% to 44.75% under the same budget increase, showing that additional search becomes more effective when delayed utility-guided consolidation maintains compact task-transferable knowledge. Thus, b=512 provides a more efficient configuration, whereas b=3072 offers higher accuracy when additional computation is acceptable.

5 Limitations

Our evaluation covers only two task types, one dataset per type, and two backbones. The three additional NeuroMem runs yield means close to the original results, but they primarily characterize task-order variation under one backbone. With main results based on single runs and no matched baseline reruns, we do not claim statistical significance. Establishing wider generality requires broader datasets, matched repeated baseline runs, and benchmark-specific horizon analyses. Compute reporting is also limited to client-observed average turn time, and provider-side computation is not observable through the hosted APIs. Beyond these evaluation limitations, memory quality in NeuroMem depends on both the underlying language model and delayed credit assignment. Different backbones can generate different memory abstractions, contextual descriptions, or connections among experiences, which may affect memory evolution and support-path retrieval. Delayed credit may also be weak or misaligned when trace contributions are indirect or distributed across a support path. These limitations motivate future work on uncertainty-aware retention for low-frequency traces, more robust credit assignment, and adaptive delayed-utility horizons and search budgets.

6 Conclusion

In this work, we propose NeuroMem, a neuroplastic memory framework for delayed consolidation in lifelong LLM agents. NeuroMem keeps new experiences provisional until later interactions reveal their delayed utility, and learns to merge, retain, or discard them into compact task-transferable knowledge. Experiments on EHRSQL and LoCoMo show that NeuroMem improves long-horizon performance with fewer persistent entries than most baselines, highlighting the value of transforming transient experiences into task-transferable long-term memory rather than merely managing stored records.

Acknowledgments and Disclosure of Funding

We thank our families, especially our parents, and our teachers for their support.

This work received no external funding. The authors declare no competing interests.

References

[1] Z. Shi, Y. Chen, H. Li, W. Sun, S. Ni, Y. Lyu, R.-Z. Fan, B. Jin, Y. Weng, M. Zhu, Q. Xie, X. Guo, Q. Yang, J. Wu, J. Zhao, X. Tang, X. Ma, C. Wang, J. Mao, Q. Ai, J.-T. Huang,

W. Wang, Y. Zhang, Y. Yang, Z. Tu, and Z. Ren, "Deep research: A systematic survey," 2025. [Online]. Available: https://arxiv.org/abs/2512.02038

[2] W. Shi, R. Xu, Y. Zhuang, Y. Yu, J. Zhang, H. Wu, Y. Zhu, J. C. Ho, C. Yang, and M. D. Wang, "EHRAgent: Code empowers large language models for few-shot complex tabular reasoning on electronic health records," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, Y. Al-Onaizan, M. Bansal, and Y.-N. Chen, Eds. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 22 315–22 339. [Online]. Available: https://aclanthology.org/2024.emnlp-main.1245/

[3] J. Mao, J. Ye, Y. Qian, M. Pavone, and Y. Wang, "A language agent for autonomous driving," in *First Conference on Language Modeling*, 2024. [Online]. Available: https://openreview.net/forum?id=UPE6WYE8vg

[4] J. Luo, W. Zhang, Y. Yuan, Y. Zhao, J. Yang, Y. Gu, B. Wu, B. Chen, Z. Qiao, Q. Long, R. Tu, X. Luo, W. Ju, Z. Xiao, Y. Wang, M. Xiao, C. Liu, J. Yuan, S. Zhang, Y. Jin, F. Zhang, X. Wu, H. Zhao, D. Tao, P. S. Yu, and M. Zhang, "Large language model agent: A survey on methodology, applications and challenges," 2025. [Online]. Available: https://arxiv.org/abs/2503.21460

[5] C. Zhou, H. Chai, W. Chen, Z. Guo, R. Shan, Y. Song, T. Xu, Y. Yang, A. Yu, W. Zhang, C. Zheng, J. Zhu, Z. Zheng, Z. Zhang, X. Lou, C. Zhang, Z. Fu, J. Wang, W. Liu, J. Lin, and W. Zhang, "Externalization in llm agents: A unified review of memory, skills, protocols and harness engineering," 2026. [Online]. Available: https://arxiv.org/abs/2604.08224

[6] X. Jiang, F. Li, H. Zhao, J. Qiu, J. Wang, J. Shao, S. Xu, S. Zhang, W. Chen, X. Tang, Y. Chen, M. Wu, W. Ma, M. Wang, and T. Chen, "Long term memory: The foundation of ai self-evolution," 2025. [Online]. Available: https://arxiv.org/abs/2410.15665

[7] OpenAI, "Introducing deep research," https://openai.com/index/introducing-deep-research/, 2025, accessed: 2026-04-23.

[8] Manus, "Welcome — manus documentation," https://manus.im/docs/introduction/welcome, 2026, accessed: 2026-04-23.

[9] OpenClaw, "Openclaw — personal ai assistant," https://openclaw.ai/, 2026, accessed: 2026-04-23.

[10] Z. Xiong, Y. Lin, W. Xie, P. He, Z. Liu, J. Tang, H. Lakkaraju, and Z. Xiang, "How memory management impacts llm agents: An empirical study of experience-following behavior," 2025. [Online]. Available: https://arxiv.org/abs/2505.16067

[11] Y. Wu, S. Liang, C. Zhang, Y. Wang, Y. Zhang, H. Guo, R. Tang, and Y. Liu, "From human memory to ai memory: A survey on memory mechanisms in the era of llms," 2025. [Online]. Available: https://arxiv.org/abs/2504.15965

[12] P. Chhikara, D. Khant, S. Aryan, T. Singh, and D. Yadav, "Mem0: Building production-ready ai agents with scalable long-term memory," 2025. [Online]. Available: https://arxiv.org/abs/2504.19413

[13] J. S. Park, J. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein, "Generative agents: Interactive simulacra of human behavior," in *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '23. New York, NY, USA: Association for Computing Machinery, 2023. [Online]. Available: https://doi.org/10.1145/3586183.3606763

[14] C. Packer, S. Wooders, K. Lin, V. Fang, S. G. Patil, I. Stoica, and J. E. Gonzalez, "Memgpt: Towards llms as operating systems," 2024. [Online]. Available: https://arxiv.org/abs/2310.08560

[15] N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao, "Reflexion: language agents with verbal reinforcement learning," in *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., vol. 36. Curran Associates, Inc., 2023, pp. 8634–8652. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2023/file/1b44b878bb782e6954cd888628510e90-Paper-Conference.pdf

[16] W. Xu, Z. Liang, K. Mei, H. Gao, J. Tan, and Y. Zhang, "A-mem: Agentic memory for LLM agents," in *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. [Online]. Available: https://openreview.net/forum?id=FiM0M8gcct

[17] B. J. Gutiérrez, Y. Shu, Y. Gu, M. Yasunaga, and Y. Su, "Hipporag: Neurobiologically inspired long-term memory for large language models," in *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, Eds., vol. 37. Curran Associates, Inc., 2024, pp. 59 532–59 569. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2024/file/6ddc001d07ca4f319af96a3024f6dbd1-Paper-Conference.pdf

[18] S. Xia, Z. Xu, J. Chai, W. Fan, Y. Song, X. Wang, G. Yin, W. Lin, H. Zhang, and J. Wang, "From experience to strategy: Empowering llm agents with trainable graph memory," 2025. [Online]. Available: https://arxiv.org/abs/2511.07800

[19] Z. Wu, H. Zhang, F. Lin, W. Xu, X. Xu, Y. Chen, H. P. Zou, S. Chen, W. Zhang, X. Liu, P. S. Yu, and H. Wang, "Gam: Hierarchical graph-based agentic memory for llm agents," 2026. [Online]. Available: https://arxiv.org/abs/2604.12285

[20] S. Yan, X. Yang, Z. Huang, E. Nie, Z. Ding, Z. Li, X. Ma, J. Bi, K. Kersting, J. Z. Pan, H. Schütze, V. Tresp, and Y. Ma, "Memory-r1: Enhancing large language model agents to manage and utilize memories via reinforcement learning," 2026. [Online]. Available: https://arxiv.org/abs/2508.19828

[21] Y. Yu, L. Yao, Y. Xie, Q. Tan, J. Feng, Y. Li, and L. Wu, "Agentic memory: Learning unified long-term and short-term memory management for large language model agents," 2026. [Online]. Available: https://arxiv.org/abs/2601.01885

[22] Y. Wang, R. Takanobu, Z. Liang, Y. Mao, Y. Hu, J. McAuley, and X. Wu, "Mem-α: Learning memory construction via reinforcement learning," 2025. [Online]. Available: https://arxiv.org/abs/2509.25911

[23] Z. Zhou, A. Qu, Z. Wu, S. Kim, A. Prakash, D. Rus, B. K. H. Low, and P. P. Liang, "MEM1: Learning to synergize memory and reasoning for efficient long-horizon agents," in *The Fourteenth International Conference on Learning Representations*, 2026. [Online]. Available: https://openreview.net/forum?id=XY8AaxDSLb

[24] Y. Dudai, "The Neurobiology of Consolidations, or How Stable Is the Engram?" *Annu. Rev. Psychol.*, vol. 55, no. 1, pp. 51–86, 2004.

[25] G. Lee, H. Hwang, S. Bae, Y. Kwon, W. Shin, S. Yang, M. Seo, J.-Y. Kim, and E. Choi, "Ehrsql: A practical text-to-sql benchmark for electronic health records," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35. Curran Associates, Inc., 2022, pp. 15 589–15 601. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/643e347250cf9289e5a2a6c1ed5ee42e-Paper-Datasets_and_Benchmarks.pdf

[26] A. Maharana, D.-H. Lee, S. Tulyakov, M. Bansal, F. Barbieri, and Y. Fang, "Evaluating very long-term conversational memory of LLM agents," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, L.-W. Ku, A. Martins, and V. Srikumar, Eds. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 13 851–13 870. [Online]. Available: https://aclanthology.org/2024.acl-long.747/

[27] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive nlp tasks," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 9459–9474. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf

[28] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, and H. Wang, "Retrieval-augmented generation for large language models: A survey," 2024. [Online]. Available: https://arxiv.org/abs/2312.10997

[29] J. Lu, S. An, M. Lin, G. Pergola, Y. He, D. Yin, X. Sun, and Y. Wu, "Memochat: Tuning llms to use memos for consistent long-range open-domain conversation," 2023. [Online]. Available: https://arxiv.org/abs/2308.08239

[30] P. Wang, M. Tian, J. Li, Y. Liang, Y. Wang, Q. Chen, T. Wang, Z. Lu, J. Ma, Y. E. Jiang, and W. Zhou, "O-mem: Omni memory system for personalized, long horizon, self-evolving agents," 2025. [Online]. Available: https://arxiv.org/abs/2511.13593

[31] H. Sun, S. Zeng, and B. Zhang, "H-MEM: Hierarchical memory for high-efficiency long-term reasoning in LLM agents," in *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, V. Demberg, K. Inui, and L. Marquez, Eds. Rabat, Morocco: Association for Computational Linguistics, Mar. 2026, pp. 341–350. [Online]. Available: https://aclanthology.org/2026.eacl-long.15/

[32] S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving with large language models," in *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., vol. 36. Curran Associates, Inc., 2023, pp. 11 809–11 822. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2023/file/271db9922b8d1f4dd7aaef84ed5ac703-Paper-Conference.pdf

[33] D. Zhang, X. Huang, D. Zhou, Y. Li, and W. Ouyang, "Accessing gpt-4 level mathematical olympiad solutions via monte carlo tree self-refine with llama-3 8b," 2024. [Online]. Available: https://arxiv.org/abs/2406.07394

[34] W. Zhong, L. Guo, Q. Gao, H. Ye, and Y. Wang, "Memorybank: Enhancing large language models with long-term memory," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 17, pp. 19 724–19 731, Mar. 2024. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/29946

[35] A. Zhao, D. Huang, Q. Xu, M. Lin, Y.-J. Liu, and G. Huang, "Expel: Llm agents are experiential learners," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 17, pp. 19 632–19 642, Mar. 2024. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/29936

[36] J. Chen, S. Xiao, P. Zhang, K. Luo, D. Lian, and Z. Liu, "Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation," 2024.

[37] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe reinforcement learning via shielding," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/11797

[38] B. Estermann, L. Lanzendörfer, Y. Niedermayr, and R. Wattenhofer, "Puzzles: A benchmark for neural algorithmic reasoning," *Advances in Neural Information Processing Systems*, vol. 37, pp. 127 059–127 098, 2024.

[39] P. Rogers, D. Wang, and Z. Lu, "Medical information mart for intensive care: A foundation for the fusion of artificial intelligence and real-world data," *Frontiers in Artificial Intelligence*, vol. Volume 4 - 2021, 2021. [Online]. Available: https://www.frontiersin.org/journals/artificial-intelligence/articles/10.3389/frai.2021.691626

[40] DeepSeek-AI, A. Liu, A. Mei, B. Lin, B. Xue, B. Wang, B. Xu, B. Wu, B. Zhang, C. Lin, C. Dong, C. Lu, C. Zhao, C. Deng, C. Xu, C. Ruan, D. Dai, D. Guo, D. Yang, D. Chen, E. Li, F. Zhou, F. Lin, F. Dai, G. Hao, G. Chen, G. Li, H. Zhang, H. Xu, H. Li, H. Liang, H. Wei, H. Zhang, H. Luo, H. Ji, H. Ding, H. Tang, H. Cao, H. Gao, H. Qu, H. Zeng, J. Huang, J. Li, J. Xu, J. Hu, J. Chen, J. Xiang, J. Yuan, J. Cheng, J. Zhu, J. Ran, J. Jiang, J. Qiu, J. Li, J. Song, K. Dong, K. Gao, K. Guan, K. Huang, K. Zhou, K. Huang, K. Yu, L. Wang, L. Zhang, L. Wang, L. Zhao, L. Yin, L. Guo, L. Luo, L. Ma, L. Wang, L. Zhang, M. S. Di, M. Y. Xu, M. Zhang, M. Zhang, M. Tang, M. Zhou, P. Huang, P. Cong, P. Wang, Q. Wang, Q. Zhu, Q. Li, Q. Chen, Q. Du, R. Xu, R. Ge, R. Zhang, R. Pan, R. Wang, R. Yin, R. Xu, R. Shen, R. Zhang, S. H. Liu, S. Lu, S. Zhou, S. Chen, S. Cai, S. Chen, S. Hu, S. Liu, S. Hu, S. Ma, S. Wang, S. Yu, S. Zhou, S. Pan, S. Zhou, T. Ni, T. Yun, T. Pei, T. Ye, T. Yue, W. Zeng, W. Liu, W. Liang, W. Pang, W. Luo, W. Gao, W. Zhang, X. Gao, X. Wang, X. Bi, X. Liu, X. Wang, X. Chen, X. Zhang, X. Nie, X. Cheng, X. Liu, X. Xie, X. Liu, X. Yu, X. Li, X. Yang, X. Li, X. Chen, X. Su, X. Pan, X. Lin, X. Fu, Y. Q. Wang, Y. Zhang, Y. Xu, Y. Ma, Y. Li, Y. Li, Y. Zhao, Y. Sun, Y. Wang, Y. Qian, Y. Yu, Y. Zhang, Y. Ding, Y. Shi, Y. Xiong, Y. He, Y. Zhou, Y. Zhong, Y. Piao, Y. Wang, Y. Chen, Y. Tan, Y. Wei, Y. Ma, Y. Liu, Y. Yang, Y. Guo, Y. Wu, Y. Wu, Y. Cheng, Y. Ou, Y. Xu, Y. Wang, Y. Gong, Y. Wu, Y. Zou, Y. Li, Y. Xiong, Y. Luo, Y. You, Y. Liu, Y. Zhou, Z. F. Wu, Z. Z. Ren, Z. Zhao, Z. Ren, Z. Sha, Z. Fu, Z. Xu, Z. Xie, Z. Zhang, Z. Hao, Z. Gou, Z. Ma, Z. Yan, Z. Shao, Z. Huang, Z. Wu, Z. Li, Z. Zhang, Z. Xu, Z. Wang, Z. Gu, Z. Zhu, Z. Li, Z. Zhang, Z. Xie, Z. Gao, Z. Pan, Z. Yao, B. Feng, H. Li, J. L. Cai, J. Ni, L. Xu, M. Li, N. Tian, R. J. Chen, R. L. Jin, S. S. Li, S. Zhou, T. Sun, X. Q. Li, X. Jin, X. Shen,

X. Chen, X. Song, X. Zhou, Y. X. Zhu, Y. Huang, Y. Li, Y. Zheng, Y. Zhu, Y. Ma, Z. Huang, Z. Xu, Z. Zhang, D. Ji, J. Liang, J. Guo, J. Chen, L. Xia, M. Wang, M. Li, P. Zhang, R. Chen, S. Sun, S. Wu, S. Ye, T. Wang, W. L. Xiao, W. An, X. Wang, X. Sun, X. Wang, Y. Tang, Y. Zha, Z. Zhang, Z. Ju, Z. Zhang, and Z. Qu, "Deepseek-v3.2: Pushing the frontier of open large language models," 2025. [Online]. Available: https://arxiv.org/abs/2512.02556

[41] Google, "Gemini 3 developer guide," https://ai.google.dev/gemini-api/docs/gemini-3, 2026, accessed: 2026-05-06.

[42] Y. Liao, S. Jiang, Y. Wang, and Y. Wang, "ReflecTool: Towards reflection-aware tool-augmented clinical agents," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, W. Che, J. Nabende, E. Shutova, and M. T. Pilehvar, Eds. Vienna, Austria: Association for Computational Linguistics, Jul. 2025, pp. 13 507–13 531. [Online]. Available: https://aclanthology.org/2025.acl-long.663/

[43] S. Dong, S. Xu, P. He, Y. Li, J. Tang, T. Liu, H. Liu, and Z. Xiang, "Memory injection attacks on LLM agents via query-only interaction," in *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. [Online]. Available: https://openreview.net/forum?id=QINnsnppv8

A Experimental Details

A.1 Dataset Descriptions

EHRSQL/EHRAgent. EHRSQL [25] is a practical text-to-SQL benchmark for electronic health records. Its questions are collected from hospital staff, including physicians, nurses, insurance reviewers, and health-record teams, and are grounded in structured EHR databases such as MIMIC-III and eICU. The benchmark covers realistic clinical information needs, including patient-level retrieval, aggregation, temporal reasoning, and answerability judgment, making it more than a simple factual lookup task. We use the EHRAgent [2] environment built on EHRSQL, where an LLM agent solves EHR questions through planning, code or SQL-related execution, feedback observation, and iterative correction. EHRAgent/EHRSQL has also served as a representative testbed in recent studies of memory-augmented and tool-augmented LLM agents [10, 42, 43]. This setting provides structured and verifiable task outcomes, allowing us to evaluate whether past task-solving experience can be retained as task-transferable knowledge and reused for new executable queries. In our experiments, useful memories are expected to capture reusable task-solving patterns, such as clinical variable grounding, temporal condition handling, field selection, and SQL operation construction, rather than storing only task-specific examples.

LoCoMo. LoCoMo [26] is a benchmark for very long-term conversational memory, where each conversation spans many turns across multiple sessions and contains long-range personal, temporal, and event-level dependencies. Unlike structured EHR tasks, LoCoMo stresses whether an agent can preserve and recombine information over extended interaction histories. Its evaluation includes question answering over different reasoning types, such as single-hop recall, multi-hop reasoning, temporal reasoning, open-domain knowledge, and adversarial questions. LoCoMo has also become a common evaluation benchmark for recent long-term memory systems and memory-management agents, including structured memory construction, agentic memory organization, and RL-based memory operation control [12, 16, 20]. We use LoCoMo to examine the compositional side of task-transferable knowledge: relevant evidence may be scattered across distant sessions, and answering a future question often requires combining stable user facts, temporal anchors, relational cues, preferences, and constraints. This makes LoCoMo suitable for testing whether a compact memory state can support long-horizon memory recomposition instead of relying on the accumulation of more conversational fragments.

A.2 Baseline Details

We compare NeuroMem with representative baselines that cover different long-term memory paradigms. Table 4 summarizes their implementation sources and configuration protocols.

Original EHRAgent [2]follows the memory strategy used in EHRAgent, where successful task-solving cases are stored and later retrieved as demonstrations for related queries. This baseline tests whether accumulating correct past experiences is sufficient, or whether those experiences need to be refined into compact task-transferable knowledge.

LoCoMo [26] is the original LoCoMo memory setting as a benchmark-specific baseline. It represents the standard long-term conversational memory setup associated with the dataset and provides a direct reference point for evaluating memory preservation over extended multi-session histories.

Mem0 [12] represents structured memory construction. It extracts, consolidates, and retrieves salient information from prior interactions, providing a strong baseline for compact memory formation.

MemoryBank [34] represents long-term conversational memory with continuous memory updating, recall, and forgetting-inspired reinforcement. It is designed to maintain user-related information across long interactions.

A-MEM [16] represents agentic memory organization. It converts new memories into structured notes and dynamically links them with related historical memories, enabling relational organization over accumulated experience.

Memory-R1 [20] represents learning-based memory control. It optimizes memory operations such as adding, updating, deleting, and retaining entries, providing a baseline for learned memory-operation policies.

Table 4: Baseline implementation sources and configuration protocols.

Baseline	Implementation and configuration protocol
Original EHRAgent LoCoMo	Rerun with benchmark-author settings.
Mem0 A-MEM MemoryBank	Released implementations with only the adapters needed to map benchmark interactions to each memory interface; released or author-recommended settings where applicable.
Memory-R1 (reproduced)	Reproduced from the reported architecture, memory operations, reward formulation, optimization setup, and hyperparameters. For unspecified details, we conducted multiple paper-consistent reproduction and debugging runs and report the strongest valid run.

Together, these baselines cover success-case memory, benchmark-specific conversational memory, structured memory construction, relational memory organization, and learned memory control. This comparison allows us to test whether NeuroMem improves long-horizon performance by retaining compact task-transferable knowledge rather than simply storing more memory records.

All baseline results are from controlled reruns; no baseline was tuned separately for individual instances or LoCoMo categories. Memory-R1 had no official implementation available at the time of our experiments, so its results are a best-case paper-based reproduction, not an official checkpoint result or an optimizer-only comparison. Under sparse binary feedback, Update or Delete may remove entries whose utility appears only in later tasks. This may explain its lower EHRSQL success than Original, but is not a causal proof.

A.3 Evaluation Metrics

EHRSQL Metrics. For EHRSQL, we report task success rate and final persistent entry count. Given an evaluation sequence of N tasks, task success rate is computed as

$$\mathrm{SuccessRate} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{1}[s_i = 1]. \tag{16}$$

where $s_i = 1$ indicates that the agent successfully completes the i-th task according to the execution-based evaluator, and $s_i = 0$ otherwise. Final persistent entry count measures the number of independently addressable persistent entries after the full evaluation sequence:

$$\mathrm{FinalPersistentEntryCount} = |\mathcal{M}_N|, \tag{17}$$

where $\mathcal{M}_N$ denotes the retained memory set after completing all N tasks. This metric is based on memory entries rather than tokens or storage size, and measures entry-level compactness.

LoCoMo Metrics. For LoCoMo, we report F1 and BLEU-4 for question answering, following the benchmark evaluation protocol. F1 measures token-level overlap between the predicted answer $\hat{a}$ and the reference answer a, while BLEU-4 measures n-gram overlap up to 4-grams. Overall F1 and BLEU-4 use category question counts as weights (282, 321, 96, 841, and 446 in the order shown in Table 1). We additionally report mean final persistent entry count across LoCoMo samples:

$$\mathrm{MeanFinalPersistentEntryCount} = \frac{1}{K}\sum_{j=1}^{K}|\mathcal{M}_{\mathrm{end}}^{(j)}|, \tag{18}$$

where K is the number of evaluated LoCoMo samples and $\mathcal{M}_{\mathrm{end}}^{(j)}$ is the retained memory set at the end of the j-th sample.

A.4 Implementation Details and Hyperparameter Sensitivity

Global NeuroMem hyperparameters were selected on a fixed development subset from the MIMIC-III portion of the official EHRSQL training split, disjoint from the 581 final evaluation tasks. No final-evaluation question, outcome, or reward trajectory was used for selection. The shared configuration

Table 5: Key NeuroMem hyperparameters shared across EHRSQL and LoCoMo. The retrieval budget b counts generated and scored path extensions. The main experiments use $b = 3072$; the EHRSQL budget–efficiency analysis additionally evaluates $b = 512$.

Hyperparameter	**Shared value**
Retrieval budget b	3072
Candidate pool size	32
Delayed-utility horizon H	32
Pending retrieval minimum utility	0.60
Promotion score threshold	0.50
Promotion minimum candidate utility	0.55
Review delay	12
Maximum transient-buffer age	96
Memory-growth coefficient λ	0.05
Redundancy coefficient β	0.10

was frozen before evaluation and reused unchanged on LoCoMo, without using LoCoMo validation or evaluation data for tuning.

Each evaluation run starts from a fresh policy and empty memory, with no policy parameters, running statistics, or memory states transferred from development. The policy adapts online only from previously observed, finalized outcomes while global hyperparameters remain fixed; no separate offline policy-training set or pretrained policy checkpoint is used. In the main evaluation, an EHRSQL run contains one stream of 581 task interactions, while a LoCoMo run comprises 10 sample-level episodes and 1,986 QA interactions. Memory is reset between LoCoMo samples to prevent direct cross-conversation trace reuse, while the policy is retained and continues to adapt within the run.

Table 5 reports the key retrieval and consolidation hyperparameters used by NeuroMem. The memory-growth and redundancy coefficients (λ, β) are shared across EHRSQL and LoCoMo, whose task rewards both lie in $[0, 1]$. For a new task, we would normalize rewards to this range using the evaluator's natural bounds and reuse these coefficients if the normalized redundancy definition is unchanged. If rewards are instead scaled by $c > 0$, scaling both coefficients by c preserves their relative weights when credit and regularization definitions are fixed. We would freeze the coefficients before evaluation, recalibrating them on a held-out development stream only if delayed-return dynamics differ substantially.

For support-path retrieval, we use $b = 3072$ in the main experiments and also evaluate $b = 512$ in the budget–efficiency analysis, keeping the remaining hyperparameters fixed. This isolates the effect of retrieval budget rather than introducing additional tuning factors. The retrieval budget limits generated and scored path extensions, the candidate pool size limits the number of available traces, and the consolidation hyperparameters regulate how provisional traces are accessed, reviewed, and promoted into persistent memory. Keeping these settings fixed in the main comparison ensures that performance differences primarily reflect how each method forms and reuses memory. Varying b therefore evaluates whether compact task-transferable knowledge can support efficient low-budget retrieval and benefit from deeper support-path exploration when more computation is available.

To illustrate the state of a candidate after the default review delay, consider the EHRSQL training question "What is the ingesting method of methimazole?" [25]. Suppose a successful interaction yields a candidate b_i that records how to retrieve distinct `prescriptions.route` values for a medication by filtering `prescriptions.drug`. After the default review delay of 12 interactions, an illustrative maximum cosine similarity of 0.82 to persistent memory gives $z_t(b_i) = [b_i, 12, 0.82]$. This similarity makes MERGE plausible, but π_θ selects among admissible actions. If the policy selects MERGE, delayed credit depends on whether the merged trace contributes to later task rewards.

Table 6: One-at-a-time hyperparameter sensitivity analysis on EHRSQL. Each sweep changes one hyperparameter while keeping all others fixed to the main NeuroMem setting. Gray rows indicate the default setting used in the main experiments.

Family	**Setting**	**Success Rate (%) ↑**
Promotion	score threshold = 0.45	41.31
Promotion	**score threshold = 0.50 (Ours)**	**44.75**
Promotion	score threshold = 0.55	42.51
Promotion	min candidate utility = 0.50	41.65
Promotion	**min candidate utility = 0.55 (Ours)**	**44.75**
Promotion	min candidate utility = 0.60	43.37
Consolidation	review delay = 8	40.79
Consolidation	**review delay = 12 (Ours)**	**44.75**
Consolidation	review delay = 24	42.86
Credit	delayed-utility horizon $H = 16$	39.93
Credit	**delayed-utility horizon $H = 32$ (Ours)**	**44.75**
Credit	delayed-utility horizon $H = 64$	42.00
Transient Buffer	retrieval min utility = 0.50	41.31
Transient Buffer	**retrieval min utility = 0.60 (Ours)**	**44.75**
Transient Buffer	retrieval min utility = 0.70	43.37

We further conduct a post-selection one-at-a-time hyperparameter sensitivity analysis on EHRSQL, reported in Table 6. The analysis covers promotion thresholds, review delay, delayed-utility horizon, and transient-buffer access thresholds. Across these groups, the default NeuroMem configuration consistently achieves the strongest success rate, suggesting that the gain is not driven by a single favorable threshold or an isolated tuning choice. Instead, the results point to a balanced delayed-consolidation process: promotion thresholds prevent noisy or local traces from entering persistent memory too early, review delay and delayed-utility horizon allow later task outcomes to reveal whether a trace is reusable, and transient-buffer access enables provisional traces to support future tasks before permanent consolidation. Overall, these results support that compact task-transferable knowledge emerges when provisional traces are neither immediately fixed nor prematurely discarded, but are consolidated after their future utility becomes observable.

A.5 Prompt Template

To support reproducibility, we provide a reference prompt template for constructing provisional task-transferable traces, specifying its inputs and structured output. The prompt neither selects lifecycle actions nor computes $\widehat{V}_t$. Lifecycle actions are selected by the learned four-way neural policy π_θ under utility-gated action masking.

To avoid prompt-level confounding, we keep the task-solving prompts, benchmark instructions, output formats, and evaluation-related prompts identical across methods whenever applicable.

The prompt template for Task-Transferable Knowledge Construction: P_{ttk}

You construct provisional memory candidates for a long-horizon LLM agent. Your goal is to distill the current task-solving experience into compact **task-transferable knowledge** that can support future unseen tasks.

Given the completed task trajectory, retrieved memories, and observed feedback, generate memory knowledge that may support later tasks.

Focus on identifying:

1. reusable reasoning patterns, constraints, or decision rules;
2. task conditions under which this knowledge should be applied;
3. failure modes, caveats, or misleading patterns that should be avoided;
4. how this knowledge can be combined with other memories to support future tasks.

Extract only information supported by the provided interaction. Preserve factual and temporal details when they are needed for later questions.

Return a provisional memory candidate as a JSON object:

```
{
  "knowledge": "A concise statement of the task-transferable knowledge.",
  "applicability": "The task conditions or query patterns where this knowledge
    is useful.",
  "reasoning_pattern": "The reusable reasoning or action strategy abstracted
    from the experience.",
  "composition_keys": [
    "keywords or concepts that help this knowledge combine with other memories"
  ],
  "evidence": "Observed evidence from the current interaction supporting this
    knowledge.",
  "cautions": "Known limitations, failure cases, or conditions where this
    knowledge may not apply.",
  "confidence": "Confidence in the supporting evidence: low / medium / high"
}
```

Task trajectory: `{task_trajectory}`

Retrieved memories: `{retrieved_memories}`

Observed feedback: `{feedback}`

B Additional Ablations and Analysis

B.1 Feedback Sanity Check

Table 7: Feedback sanity check on a 120-turn EHRSQL subset. Shuffling feedback breaks the correspondence between task outcomes and memory updates.

Feedback	Success Rate ↑
Real feedback	38.33%
Shuffled feedback	27.50%

Table 7 reports a shuffled-feedback sanity check on a 120-turn EHRSQL subset. The goal is to test whether NeuroMem benefits from task-aligned delayed utility rather than from online memory updates alone. Under real feedback, later task outcomes remain linked to the provisional traces and support paths involved in task solving; in the shuffled setting, this correspondence is randomly broken while the same consolidation mechanism is kept unchanged.

Shuffling feedback reduces the success rate from 38.33% to 27.50%, showing that delayed consolidation depends on meaningful feedback alignment. This suggests that NeuroMem does not merely accumulate or update memories online, but uses later task outcomes to decide which provisional traces should be promoted, merged, retained, or discarded. The sanity check therefore supports our claim that compact task-transferable knowledge emerges through delayed utility-guided consolidation.

B.2 Task-Order Robustness

We ran NeuroMem three additional times with Gemini-3-Flash using task-order permutations indexed by seeds 0, 1, and 2, keeping evaluation instances, splits, decoding settings, evaluator, and hyperparameters fixed. Each run started from a fresh policy and empty memory. We permuted the same 581 EHRSQL tasks. For LoCoMo, we permuted QA order within each sample while preserving dialogue and session chronology; memory was reset between samples, while the policy was retained within each run.

B.3 Reward-Component Ablation

We ablate the memory-growth and redundancy penalties while fixing task order, delayed-credit construction, policy architecture, optimizer, and all other settings.

Removing either penalty lowers success and increases final persistent entry count; removing both causes the largest degradation. These ablations assess the regularized delayed-return formulation while keeping the optimizer fixed. Memory-R1 also differs in state, operations, reward formulation, and credit assignment, so it is not an optimizer-only comparison. We have not tested whether PPO or GRPO improves downstream performance; a matched optimizer-only comparison remains future work.

C Impact Statement

NeuroMem introduces a lifecycle-based memory framework for long-horizon LLM agents, enabling agents to selectively consolidate, retrieve, and compose past experiences as compact task-transferable knowledge. By modeling the delayed utility of memories across future tasks and adapting external memory at test time, NeuroMem may improve the efficiency and generalization ability of LLM agents in settings that require long-term reasoning. More broadly, this work contributes to the development of scalable and adaptive memory-augmented AI systems that can reuse prior experience across related and previously unseen tasks while maintaining a compact memory state.

However, as with other memory-augmented LLM systems, the broader impact of this framework depends on the reliability of the underlying language model and the quality of stored information. If the model is compromised, adversarially manipulated, or exposed to incorrect or biased inputs, memory mechanisms may preserve or recall misleading information in future interactions. We therefore encourage responsible deployment with appropriate safeguards, including privacy-preserving

Table 8: Task-order permutations with Gemini-3-Flash. Mean $\pm$ SD denotes the mean and sample standard deviation across the three additional runs, excluding the original run.

Benchmark	**Metric**	**Seed 0**	**Seed 1**	**Seed 2**	**Mean $\pm$ SD**	**Original**
EHRSQL	Success (%)	44.23	43.55	45.61	44.46 ± 1.05	44.75
	Final entries	54	53	55	54.0 ± 1.0	54
LoCoMo	Overall F1	29.08	31.00	32.38	30.82 ± 1.66	32.08
	Overall BLEU-4	11.98	12.56	12.63	12.39 ± 0.36	12.58
	Mean final entries	125.9	121.7	121.1	122.9 ± 2.6	123.3

Table 9: Reward-component ablation on EHRSQL with Gemini-3-Flash.

Reward configuration	**Success (%)**	**Final entries**
w/o memory-growth penalty ($\lambda = 0$)	39.76	82
w/o redundancy penalty ($\beta = 0$)	40.28	61
Unregularized delayed utility ($\lambda = \beta = 0$)	36.83	114
Full regularized return	**44.75**	**54**

memory storage, human oversight, continual validation, adversarial robustness checks, and alignment with human values.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The main claims in the abstract and introduction are limited to the proposed memory framework, delayed utility based consolidation, and empirical improvements on EHRSQL and LoCoMo. These claims are supported by the methodology and experimental results reported in Sections 3 and 4.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss limitations in Section 5.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [N/A]

Justification: This paper does not include theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide the relevant information for reproducing the experimental results; see Section 4 and Appendix A.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: We provide the relevant code with clear instructions through the public GitHub repository linked in the abstract. All datasets used in this paper are publicly available and clearly cited.

Guidelines:

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

6. **Experimental setting/details**

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

Answer: [Yes]

Justification: We describe the dataset setups, evaluation metrics, backbone models, baselines, and memory-management settings in Section 4 and Appendix A. The key hyperparameters are reported in Appendix A.4. We follow the official benchmark settings for dataset splits and evaluation.

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [No]

Justification: We do not report error bars or statistical significance tests for the main comparisons, which use single runs because repeated long-horizon evaluations through hosted LLM APIs are costly. Appendix B.2 reports means and sample standard deviations across three additional NeuroMem runs with Gemini-3-Flash under different task orders. Since the baselines were not rerun under those same orders, we do not claim that the differences between methods are statistically significant.

Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [No]

   Justification: Table 3 reports client-observed average turn time for the EHRSQL budget comparison. The hosted APIs do not expose provider-side hardware, GPU hours, or energy use, so we cannot report total compute for each experiment.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and believe the research conforms to it.

   Guidelines:

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

10. **Broader impacts**

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

Answer: [Yes]

Justification: We provide a broader impact statement in Appendix C, discussing both the potential benefits of NeuroMem for long-horizon memory-augmented agents and the risks associated with unreliable underlying models or stored information.

Guidelines:

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

11. **Safeguards**

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

Answer: [N/A]

Justification: The paper poses no such risks.

Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: We cite the original sources for all datasets, benchmarks, baselines, and model backbones used in the experiments; see Section 4 and Appendix A. We use existing assets according to their stated licenses and terms of use where available.

Guidelines:

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

13. **New assets**

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

    Answer: [Yes]

    Justification: We provide the code, prompts, and configuration files with documentation and reproduction instructions in the public GitHub repository. The paper does not introduce a new dataset or pretrained model; all used datasets and model backbones are publicly available or accessed through their official APIs and are properly cited.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

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

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

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

Answer: [N/A]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [Yes]

Justification: LLMs are used as the backbone models of the evaluated agents, while the proposed contribution is the external memory framework. We describe the LLM backbones in Section 4 and Appendix A.

Guidelines:

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