EVOLVING SEMANTIC AND PROCEDURAL MEMORY FROM SHARED EXPERIENCE FOR LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

LLM agents need persistent memory that keeps factual knowledge current and develops reusable procedures from experience. Factual revision and procedural learning require distinct evidence. Coordinating these updates requires a shared context that connects factual knowledge with the experience underlying reusable procedures. We propose Self-Evolving Experience Memory (SEEM^2), an architecture that evolves semantic and procedural memory from shared experience scenes that record context, goals, actions, and outcomes while preserving links to source evidence. Semantic updates revise scoped factual states using new observations while retaining history and unresolved conflicts. Procedural updates learn and refine reusable steps from successful traces, while failures inform applicability conditions and reuse eligibility. Retrieved memories and their supporting evidence guide subsequent tasks, whose observations and outcomes drive further updates with model parameters fixed. Compared with the strongest evaluated baselines, SEEM^2 improves accuracy from 67.67% to 79.33% on the MemoryAgentBench variant of LongMemEval and from 75.48% to 92.90% on BFCL V4 Memory. On the τ-bench family, overall task success increases from 90.65% to 93.17% with DeepSeek-V4-Flash and from 74.46% to 80.22% with Qwen3.8-27B. In a matched Qwen3.8-27B comparison, continued memory updates achieve 81.53% success versus 74.32% with frozen memory, supporting the benefit of ongoing memory evolution.

1 INTRODUCTION

As users and environments change, LLM-based agents need to keep their knowledge current and reuse experience across tasks. Persistent memory supports both capabilities. Semantic memory represents facts, states, and behavioral regularities, while procedural memory describes how to perform tasks. The same interaction can inform both, but revising stored knowledge and learning reusable procedures depend on different aspects of the evidence.

Factual updates depend on observed states, whereas procedural learning depends on demonstrated effectiveness. Overall task success does not establish both. Applying a single update criterion can therefore discard useful facts or preserve ineffective steps. Distinct criteria let an agent learn what changed, what worked, and when a procedure should be reused from the same experience.

Temporal memory systems track changing facts (Rasmussen et al., 2025), while procedural memory systems learn reusable guidance from execution traces (Wang et al., 2025; Fang et al., 2026c; Ouyang et al., 2026). Recent work also explores connections among memory types and the use of evidence to guide skill updates (Fang et al., 2026b; Tang et al., 2026). Building on these advances, we study update decisions within connected memories, examining when an observation can supersede a scoped assertion, when an execution supports a reusable procedure, and when failure should restrict its reuse.

We propose Self-Evolving Experience Memory (SEEM^2), an architecture with complementary semantic and procedural evolution paths (Figure 1). Building on SEEM's structured event representation and provenance links (Lu et al., 2026), experience scenes preserve relations among observations, actions, outcomes, and their context. Linking both memories to these scenes and their sources keeps factual context, actions, and outcomes connected for later revision and retrieval. The memories share this evidential context while applying distinct rules to what they retain or revise. Retrieved

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

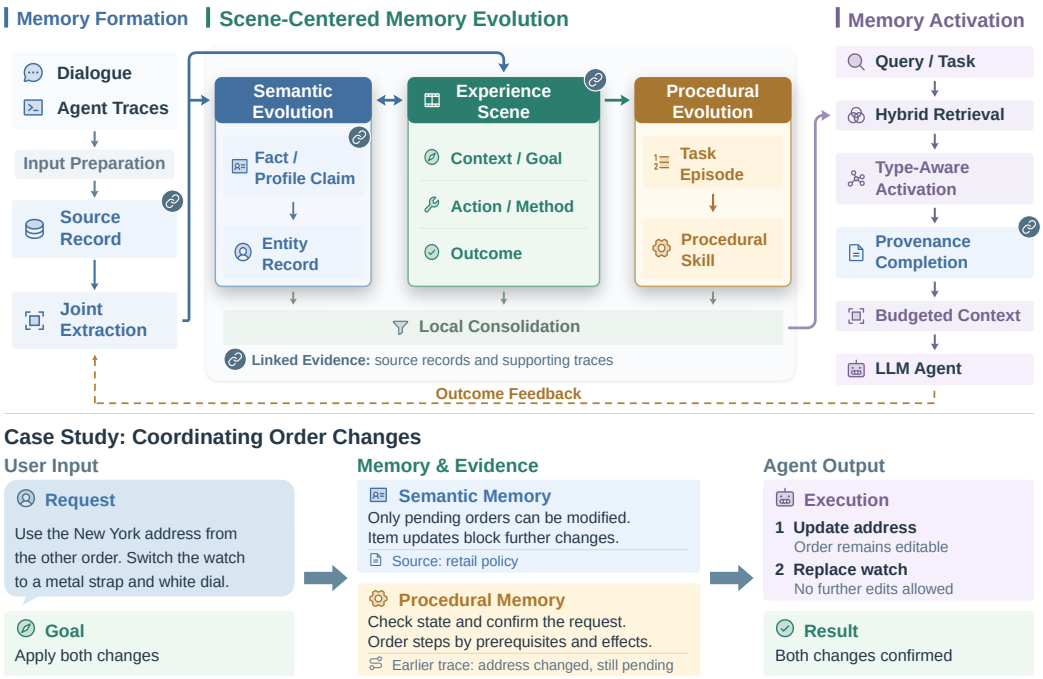


Figure 1: Overview of SEEM^2 and a case study of coordinated order changes using semantic memory, procedural memory, and supporting evidence.

memories guide subsequent tasks, whose observations and outcomes drive further updates. Here, self-evolution denotes automatic changes in external memory content, organization, and reuse eligibility, with model parameters fixed.

We make three contributions.

- **A shared experience architecture.** We organize semantic assertions and reusable procedures around shared experience scenes and source records. This architecture preserves links between factual context, actions, and outcomes, providing a common evidential basis for retrieving and revising both memory types.

- **Distinct evidence criteria for memory evolution.** Semantic updates consider entity identity, scope, and observation time while retaining history and unresolved conflicts. Inferred behavioral profiles require support from multiple distinct events. Successful traces support reusable steps, with recovery steps requiring observed successful repairs, while relevant failures inform applicability and eligibility for reuse. Consolidation proposals are checked against source evidence and these update rules.

- **Empirical evidence across memory and tool-use tasks.** Compared with the strongest evaluated baselines, SEEM^2 improves accuracy from 67.67% to 79.33% on the MemoryAgentBench variant of LongMemEval (S^*) (Wu et al., 2025; Hu et al., 2026b) and from 75.48% to 92.90% on BFCL V4 Memory (Patil et al., 2025). On the τ-bench family (Yao et al., 2024; Barres et al., 2025), overall task success increases from 90.65% to 93.17% with DeepSeek-V4-Flash and from 74.46% to 80.22% with Qwen3.8-27B. Ablations support the value of shared memory organization, semantic state maintenance, and procedural updates from failed tasks. In a matched Qwen3.8-27B comparison, task success is 7.21 percentage points higher with continued updates than with frozen memory (81.53% vs. 74.32% over 222 tasks).

2 Related Work

Structured and Temporal Memory. Agent memory systems organize persistent information through relations, temporal validity, and abstraction. HippoRAG supports associative retrieval through graph structure, and HippoRAG 2 extends this approach to non-parametric continual learning (Gutiérrez et al., 2024; 2025). Zep's Graphiti engine tracks evolving facts and historical relations (Rasmussen et al., 2025), while A-MEM and Mem0 revise and consolidate conversational memories (Xu et al., 2025; Chhikara et al., 2025). MemoryOS organizes storage across memory timescales, and LightMem combines topic-aware consolidation with offline updates (Kang et al., 2025; Fang et al., 2026a). LycheeMemory V2 reduces encoding frequency by consolidating semantic segments into typed memory records, which support plan-guided evidence retrieval (Li et al., 2026). NEMORI distills semantic knowledge from episodic narratives, while EverMemOS consolidates conversational records into thematic scenes (Ma et al., 2026; Hu et al., 2026a). SEEM combines structured event frames with relational memory and reverse provenance expansion for contextual recall (Lu et al., 2026). These approaches inform our use of scenes and provenance. SEEM^2 extends this foundation by connecting evolving semantic assertions and reusable procedures to shared experience, with separate rules for their revision.

Experience and Procedural Memory. Experience can also be abstracted into guidance for future action. ExpeL extracts natural-language insights (Zhao et al., 2024), Agent Workflow Memory induces reusable workflows (Wang et al., 2025), and MemP studies procedural construction, retrieval, and updating at different granularities (Fang et al., 2026c). ReasoningBank distills reasoning strategies from experiences the agent judges successful or unsuccessful (Ouyang et al., 2026). Together, these methods establish several ways to reuse successful and failed experience. Our procedural path uses successful traces to support a skill's procedural core and failures to revise its conditions and support for reuse. Recovery steps require an observed repair within a successful trace.

Evolving Memory and Skills. Voyager maintains an expanding library of executable skills and uses environment feedback and self-verification to refine candidate programs (Wang et al., 2024). FluxMem connects semantic, episodic, and procedural memory in a heterogeneous graph, refines both connections and content through feedback, and evaluates consolidated skills by rerunning their source episodes (Fang et al., 2026b). MemSkill evolves the procedures used to manage memory, including extraction, consolidation, and pruning (Zhang et al., 2026). MSCE links traces, policies, and environmental knowledge through evidence, using positive and counter-evidence to govern skill formation and applicability (Tang et al., 2026). SEEM^2 applies separate update criteria over shared scenes, using entity, scope, and observation time to govern factual succession and execution outcomes to govern procedural revision.

3 Scene-Centered Typed Memory Evolution

3.1 Problem Setup and Overview

We organize SEEM^2 into memory formation, scene-centered memory evolution, and memory activation (Figure 1). Formation records sources and jointly extracts typed memories. Evolution then applies distinct semantic and procedural update rules and local consolidation, and activation assembles relevant memories with supporting evidence.

The agent maintains a typed graph $\mathcal{M}_t = (\mathcal{V}_t, \mathcal{R}_t)$ of memory records and their relations. Given a query or task q_t, it retrieves context $\mathcal{C}_t$, produces a response or execution y_t, and updates memory according to

$$\mathcal{C}_t = \mathrm{Retrieve}(q_t, \mathcal{M}_t, B), \qquad y_t \sim p_\theta(\cdot \mid q_t, \mathcal{C}_t), \qquad \mathcal{M}_{t+1} = \mathcal{T}(\mathcal{M}_t, z_t), \tag{1}$$

where B is the context token budget, model parameters θ remain fixed, and z_t contains new observations, execution records, feedback, and any delayed outcome corrections. The update operator $\mathcal{T}$ incorporates z_t by forming source-linked records, revising semantic and procedural memory, and consolidating the updated graph into $\mathcal{M}_{t+1}$. LLMs interpret source content during extraction, propose profile regularities and reassess their evidence, and propose local consolidation. Rule-based checks validate state succession, procedural acceptance, evidence accounting, and consolidation operations against the stored records and their source links.

3.2 Shared Experience Representation

From sources to memory records. Input preparation (Figure 1) formats incoming dialogue and execution traces with available speaker, time, and session metadata, retaining them as source records. Joint extraction identifies entities, assertions, scenes, and direct profile claims while resolving entity references to persistent identities. An assertion records a claim about an entity or state, while a scene records the surrounding experience. Both link to the original evidence. Statements retain their evidential role, distinguishing observations, requests, proposals, instructions, executed actions, and verified results.

Scenes, episodes, and subgoals. A scene $x = (c_x, g_x, a_x, m_x, o_x, \rho_x)$ follows SEEM's representation (Lu et al., 2026). Its fields record context c_x (participants, time, and locale), goals and constraints g_x, observed actions a_x, methods and tool arguments m_x, and outcomes or state changes o_x. Source references ρ_x link these fields to evidence. Missing information remains unspecified. The same fields describe conversational and execution scenes. An *episode* groups ordered scenes for a task, retaining its goal, constraints, overall outcome, and local failures or corrections. A *subgoal view* selects scenes relevant to one subgoal and keeps links to those scenes and their parent episode. Its outcome comes from the selected scenes, not automatically from the episode's final result.

Sources, entities, assertions, profile claims, scenes, episodes, and skills are stored as canonical records connected by typed relations. These records provide the basis for state histories, current-profile views, and retrieval summaries. These derived views retain source links. Documents can supply assertions directly. Assertions and profile claims feed semantic updates, while episodes and subgoal views provide the source experiences used for procedural updates.

Case study: coordinating order changes. In Figure 1, a user asks to change a pending order's address from A to another order's New York address B and replace a watch. Semantic memory provides policies requiring pending status for modifications and barring further edits after item updates. Procedural memory supplies state checks and sequencing guidance, supported by an earlier trace in which an address update preserved pending status. The agent therefore updates the address before replacing the watch. Current tool results confirm both changes, while source links distinguish the policy, earlier execution evidence, and current outcomes.

3.3 Scope-Aware Semantic and Profile Evolution

Semantic memory updates particular states and infers behavioral regularities from distinct experiences. Table A.1 summarizes the notation.

Identifying the state to update. An assertion f has state key $k(f) = (u_f, p_f, \sigma_f, \mu_f)$, comprising its subject, normalized attribute or relation, scope, and update mode. The changing value is excluded. In the case study, the old and new addresses refer to the same order-scoped state. Scope identifies the state being described, whereas provenance identifies the source of the description. Two conversations can therefore describe the same scoped state.

The update mode determines which changes are allowed. Scalar states and cumulative snapshots can be replaced. Set membership and events are added unless explicitly retracted. For example, a new reported total may replace an earlier total, while a separate purchase remains a distinct event.

Updating state while retaining history. For a stored assertion f and a new assertion f' formed from z_t, semantic evolution permits implicit succession only if

$$k(f') = k(f), \quad \mu_f \in \{\text{scalar}, \text{cumulative}\}, \quad C_{\text{state}}(f, f') = 1, \quad \tau_{\text{obs}}(f') > \tau_{\text{obs}}(f). \tag{2}$$

The rule accepts a newer, compatible assertion about the same replaceable state. The rule-based predicate C_{state} checks identity, scope, value and unit meanings, and source eligibility using extracted attributes and their source links. Observation time τ_{obs} governs succession, while valid time records when the described state holds and supports historical queries.

An accepted successor updates the current view and links to its predecessor, preserving it as history. The fixed state key and strictly increasing observation time keep each succession chain within one

scoped state and prevent cycles. For the order in the case study, an eligible result supporting B can supersede A while retaining the earlier address as history.

The rule is subject to the following boundary conditions.

- **Scope.** Different state keys remain separate. A missing qualifier does not imply global applicability.
- **Time.** Equal or missing observation times do not establish implicit succession. Ingestion order cannot substitute for observation order.
- **Evidence.** Similarity selects candidates for comparison. Explicit updates and retractions require eligible source evidence beyond the extractor's operation label. Competing assertions remain unresolved when evidence does not justify succession.

Inferring profiles across events. A direct profile claim records an explicit statement, while an inferred claim generalizes a routine or choice tendency from observed events. Within an observation window, an LLM proposes a scoped pattern. A second LLM pass reassesses the pattern against the original sources without receiving the proposer's evidence labels. Both passes assess supporting and opposing events, including contrary declarations and temporary constraints in the source context.

Acceptance requires agreement on the evidence labels, supported attribution and scope, and no unresolved gaps in relevant evidence. Rule-based checks then enforce minimum support across distinct events, days, temporal span, and source groups. Choice tendencies additionally require observed voluntary choices among explicit alternatives and sufficient support relative to the recorded choice opportunities. Each underlying event contributes once, regardless of how many assertions or scenes describe it.

Accepted claims retain their scope, observation window, and inferred status. New relevant evidence can suspend reuse pending review. Conflicting direct statements or insufficient remaining support invalidate a claim. Claims can also expire after a validity period measured in source time. Direct statements take precedence. The induction procedure excludes inferred claims as source events. Repeated voluntary choices of coffee over tea on business trips, for example, may support a travel-scoped tendency.

3.4 Outcome-Conditioned Procedural Evolution

Representing a reusable skill. A source experience ϵ is an episode or subgoal view from Section 3.2. A skill s stores a procedural core π_s (how to act), applicability conditions α_s (when to use it), a lifecycle state ℓ_s governing reuse, and a source-indexed evidence ledger $\mathcal{L}_s$. Candidates preserve action order, methods, parameter dependencies, expected effects, and executor roles. Instance identifiers become parameters, and required inputs remain preconditions.

Applicability covers the task domain, required inputs, and tool assumptions. The case study's address assertion concerns one order, whereas its procedure applies to other orders with compatible states and requested changes. Experience matching compares goals, domains, tool operations, and state changes.

Maintaining the evidence ledger. For a matched experience ϵ, let d_ϵ contain its execution evidence, observed outcome, and provenance. We update the source-indexed ledger as

$$\mathcal{L}_s^+(i) = \begin{cases} d_\epsilon, & i = \mathrm{id}(\epsilon), \\ \mathcal{L}_s(i), & \text{otherwise} \end{cases}. \tag{3}$$

Here $+$ denotes a local skill update within the transition $\mathcal{M}_t \to \mathcal{M}_{t+1}$, and $\mathrm{id}(\epsilon)$ identifies the source experience. Reprocessing or delayed feedback revises the same entry, leaving other evidence unchanged. Outcome statistics are recomputed from these source-indexed records, so corrected feedback can withdraw earlier support. One explicitly successful experience with reusable execution content may initialize a provisional skill.

Deciding whether to replace the core. An eligible success ϵ proposes π_ϵ, and the core update is

$$\pi_s^+ = \begin{cases} \pi_\epsilon, & C_{\mathrm{proc}}(\epsilon, s) = 1 \ \wedge \ Q(\epsilon) > Q_s + \delta_Q, \\ \pi_s, & \text{otherwise} \end{cases}. \tag{4}$$

The same ϵ is recorded in Equation 3. C_{proc} checks compatibility of the execution structure and applicability conditions. $Q(\epsilon)$ and Q_s score the evidence supporting the candidate and stored cores, respectively, on $[0, 1]$ using the fixed criteria in Appendix B.1. The positive margin δ_Q sets a threshold for the score increase over Q_s. When the core is replaced, Q_s becomes $Q(\epsilon)$. Outcome frequencies remain separate in the ledger. Compatible successes can add support and compact variants without replacing the core. Mutually exclusive branches remain separate variants.

Learning from failures and controlling reuse. Matched failures, partial outcomes, and timeouts contribute evidence through Equation 3 and can revise α_s and ℓ_s. The core update in Equation 4 draws on eligible successful traces. A recovery step requires an observed error followed by a successful repair supported by an eligible trace.

The updated ledger $\mathcal{L}_s^+$ supports reassessment of applicability and reuse eligibility. A provisional skill can be retrieved before repeated support accumulates. Contradictory evidence can quarantine a weakly supported skill. Support statistics and reuse eligibility can change while the core stays intact. The system records *retrieval exposure* when a skill is included in a task's context, separately from execution evidence and task outcomes.

3.5 Consolidation and Activation

Validating local consolidation. Formation and the updates in Equations 2–4, together with profile and reuse revisions, yield $\widetilde{\mathcal{M}}_{t+1}$. For local groups with redundancy, conflicts, or procedural evidence requiring review, an LLM proposes operations $\widehat{\Delta}_t$ to mark duplicates, organize supported updates, record conflicts, add restricted annotations, or retire skills. Validation and application yield

$$\begin{aligned} \Delta_t^{\text{valid}} &= \text{Validate}(\widehat{\Delta}_t, \widetilde{\mathcal{M}}_{t+1}), \\ \mathcal{M}_{t+1} &= \text{Apply}(\widetilde{\mathcal{M}}_{t+1}, \Delta_t^{\text{valid}}). \end{aligned} \tag{5}$$

Validate deterministically checks proposals against source evidence and the preceding update rules. Apply executes accepted operations while preserving memory text and source evidence, yielding $\mathcal{M}_{t+1}$ and completing $\mathcal{T}$ in Equation 1.

Activating memory with its evidence. The retrieval operator in Equation 1 constructs $\mathcal{C}_{t+1}$ from the consolidated graph and is defined by

$$\text{Retrieve}(q_{t+1}, \mathcal{M}_{t+1}, B) = \text{Pack}(\text{ProvExpand}(\text{Select}(q_{t+1}, \mathcal{M}_{t+1})), B). \tag{6}$$

Select combines dense, lexical, entity, and associative retrieval with bounded expansion along typed relations, then applies type-aware activation. Current-state queries favor compatible current assertions, including successors accepted by Equation 2. Historical queries can retrieve earlier versions. Profiles retain their scope and inferred status. Skills are selected by relevance, applicability α_s, support derived from $\mathcal{L}_s$, and lifecycle state ℓ_s.

ProvExpand completes provenance by following links to supporting scenes, episodes, and source passages (Lu et al., 2026). In the case study, policy passages ground the semantic conditions, while earlier tool results support the procedural context. Retrieved sources guide sequencing, and current tool results verify execution. Pack fits memories and supplementary evidence within the shared context budget. Question answering mainly uses semantic records, scenes, and sources, while tool-using tasks also use episodes and skills. New observations and feedback enter z_{t+1} for the next update in Equation 1, closing the dashed loop in Figure 1.

4 Experiments and Results

We evaluate SEEM^2 on long-term conversational memory and conversational tool use. The experiments compare overall performance and examine shared experience organization, memory updates, and long-term recall.

4.1 Experimental Setup

Benchmarks and models. We evaluate long-term conversational memory on LongMemEval (S^*), the MemoryAgentBench variant with 300 questions over five histories (Wu et al., 2025; Hu et al.,

2026b). We use BFCL V4 Memory's 155 questions across five scenarios to evaluate tool-mediated persistent memory (Patil et al., 2025). Conversational tool use is evaluated on the airline, retail, and telecom domains of the τ-bench family (Yao et al., 2024; Barres et al., 2025). We use the same official DeepSeek-V4-Flash-0731 open-weight checkpoint (denoted DeepSeek-V4-Flash throughout) as the agent backbone for LongMemEval (S^*), BFCL V4 Memory, and τ-bench (DeepSeek-AI, 2026), and additionally evaluate Qwen3.8-27B on τ-bench (Qwen Team, 2026). For memory retrieval on LongMemEval and τ-bench, SEEM^2 uses Qwen3-Embedding-8B (Zhang et al., 2025). The full SEEM^2 configuration enables cross-event profile induction.

Baselines. For LongMemEval, we compare with Full Context, two dense retrieval-augmented generation (Dense RAG) baselines, Graphiti, the open-source temporal graph engine powering Zep's memory service (Rasmussen et al., 2025), LightMem (Fang et al., 2026a), Mem0 (Chhikara et al., 2025), and A-MEM (Xu et al., 2025). Dense RAG uses Qwen3-Embedding-8B or KaLM-Embedding-V2.5 (Zhao et al., 2026) (Qwen3 and KaLM in Table 1). BFCL V4 Memory compares against Vector Store (unstructured vector retrieval), Graphiti, LightMem, and Mem0. For τ-bench, the baselines are Agent Workflow Memory (AWM), ReasoningBank, and an agent without persistent memory. AWM induces reusable workflows from past task trajectories (Wang et al., 2025), while ReasoningBank distills reasoning strategies from successful and failed trajectories (Ouyang et al., 2026).

Evaluation protocol. On LongMemEval, we report accuracy judged by DeepSeek-V4-Pro (DeepSeek-AI, 2026) and normalized token-overlap F1. We report official answer-matching accuracy on BFCL V4 Memory and task success rates on τ-bench. Overall scores pool questions or tasks. The main τ-bench comparison covers 278 tasks per backbone, with memory accumulated across tasks. Appendix B gives initialization and execution details.

4.2 Main Results

Long-term conversational memory. SEEM^2 achieves 79.33% accuracy and 51.86 F1, compared with 64.33% and 46.42 for Graphiti and 62.67% and 42.82 for LightMem, the two memory baselines with the highest accuracy (Table 1). It outperforms LightMem across all six question types (Figure 2(a)), while Full Context remains stronger on user-information questions. The gains on questions spanning conversations or involving changing facts suggest value in linking experiences and keeping information current. Selected examples are analyzed in Appendix C.2.

Tool-mediated persistent memory. SEEM^2 achieves 92.90% accuracy, compared with 75.48% for Mem0 and 74.84% for Graphiti, the two strongest baselines by overall accuracy (Table 2). It ranks first or joint first in all five scenarios, with the largest margins over the respective strongest baselines in the student and customer scenarios (Figure 2(b)).

Conversational tool use. With DeepSeek-V4-Flash, SEEM^2 achieves 93.17% overall success, compared with 86.69% without memory, 90.65% for ReasoningBank, and 89.93% for AWM. With Qwen3.8-27B, the corresponding scores are 80.22%, 69.06%, 70.50%, and 74.46%, making AWM the strongest baseline overall for this backbone. SEEM^2 ranks first or joint first in every domain with both backbones, tying AWM at 84.21% on Qwen3.8-27B retail (Table 3). Both backbones benefit, with larger gains for Qwen3.8-27B, although a substantial performance gap remains between them.

4.3 Ablation Study

Semantic evolution and long-term recall. On LongMemEval (S^*), all three ablations reduce accuracy and F1 (Table 4(a)). Removing state and profile maintenance has the largest observed effect on accuracy, from 79.33% to 74.67%, even though dated, scoped assertions remain available. Removing scene associations or supplementary source retrieval also lowers performance. Together, these results support keeping knowledge current and retaining access to its context and evidence.

Shared experience and procedural updates. On τ-bench with Qwen3.8-27B, maintaining separate memory structures, removing semantic memory, or disabling failure-driven procedural updates lowers success (Table 4(b)). The organization ablation retains both memory types and their updates

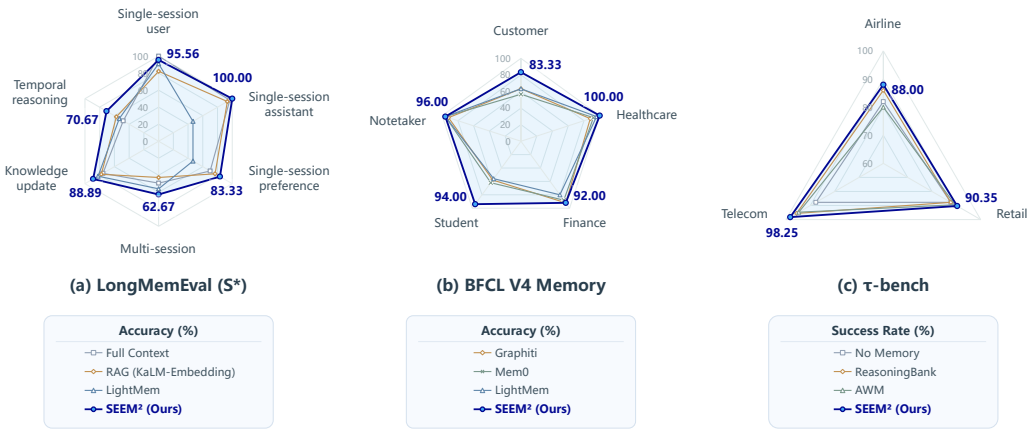


Figure 2: Category-level performance with selected baselines on (a) LongMemEval (S^*), (b) BFCL V4 Memory, and (c) the τ-bench family, using DeepSeek-V4-Flash.

Table 1: Accuracy (%) and F1 on LongMemEval (S^*). In the main comparison tables, **bold** and underlining indicate the best and second-best scores, respectively.

Method	Single-session			Multi-session (75)	Knowledge update (45)	Temporal reasoning (75)	Overall (300)	
	User (45)	Assistant (30)	Preference (30)				Acc.	F1
Full Context	**100.00**	96.67	70.00	49.33	75.56	48.00	67.33	39.15
Dense RAG (Qwen3)	86.67	96.67	63.33	53.33	75.56	56.00	67.67	45.19
Dense RAG (KaLM)	82.22	93.33	76.67	42.67	77.78	57.33	66.00	42.92
Graphiti	86.67	**100.00**	33.33	40.00	75.56	66.67	64.33	46.42
LightMem	91.11	46.67	46.67	56.00	82.22	53.33	62.67	42.82
Mem0	93.33	13.33	70.00	58.67	66.67	36.00	56.00	35.66
A-MEM	77.78	96.67	36.67	30.67	68.89	41.33	53.33	37.91
SEEM^2	95.56	**100.00**	**83.33**	**62.67**	**88.89**	**70.67**	**79.33**	**51.86**

Table 2: Accuracy (%) across five persistent-memory scenarios in BFCL V4 Memory.

Method	Customer (30)	Healthcare (25)	Finance (25)	Student (50)	Notetaker (25)	Overall (155)
Vector Store	36.67	52.00	76.00	52.00	76.00	56.77
Graphiti	63.33	88.00	**92.00**	58.00	92.00	74.84
LightMem	63.33	96.00	80.00	56.00	**96.00**	74.19
Mem0	56.67	92.00	88.00	62.00	**96.00**	75.48
SEEM^2	**83.33**	**100.00**	**92.00**	**94.00**	**96.00**	**92.90**

within the same context budget. The failure ablation retains failure trajectories and successful-task updates. Together, the results suggest that effective reuse benefits from connecting experiences and using their outcomes to guide later updates. Appendix D.1 details these controls.

From the same memory built on approximately 20% of tasks, we freeze subsequent updates in Frozen and retain them in Full. On the remaining 222 tasks in matched order, Full achieves 81.53% success versus 74.32% for Frozen, consistent with a benefit from continued updates (Appendix D.2).

Table 3: Task success rates (%) on the τ-bench family with DeepSeek-V4-Flash and Qwen3.8-27B.

Method	Airline (50)	Retail (114)	Telecom (114)	Overall (278)
DeepSeek-V4-Flash				
No memory	82.00	87.72	87.72	86.69
ReasoningBank	86.00	87.72	95.61	90.65
AWM	80.00	89.47	94.74	89.93
SEEM^2	**88.00**	**90.35**	**98.25**	**93.17**
Qwen3.8-27B				
No memory	68.00	82.46	56.14	69.06
ReasoningBank	64.00	83.33	60.53	70.50
AWM	70.00	**84.21**	66.67	74.46
SEEM^2	**82.00**	**84.21**	**75.44**	**80.22**

Table 4: Component ablations of SEEM^2 evaluated by (a) accuracy and F1 on LongMemEval (S^*) and (b) task success rate on the τ-bench family.

(a) LongMemEval (S^*) DeepSeek-V4-Flash			(b) τ-bench Qwen3.8-27B	
Variant	Acc. (%)	F1	Variant	Success (%)
SEEM^2	**79.33**	**51.86**	SEEM^2	**80.22**
w/o Semantic State Maintenance	74.67	49.75	w/o Shared Memory Organization	74.82
w/o Scene Associations	76.67	50.22	w/o Semantic Memory	73.74
w/o Source Completion	78.67	50.16	w/o Failure Updates	76.98

5 Conclusion

We presented SEEM^2, a memory architecture that updates semantic knowledge and reusable procedures from shared experience. Shared scenes connect source evidence to facts, behavioral profiles, and procedures. Each memory type follows its own update rules, and derived records retain links to their sources for later review and retrieval.

SEEM^2 achieves higher overall accuracy than the evaluated baselines on LongMemEval (S^*) and BFCL V4 Memory, and higher task success on the τ-bench family. The corresponding ablations report lower aggregate scores when semantic state maintenance, scene associations, or failure updates are removed.

The architecture connects what an agent knows with how it acts through shared experience, while retaining the evidence used to form each memory. Further work can examine how these memories evolve over longer interactions.

Reproducibility Statement

Section 3 describes the architecture, memory update rules, and retrieval procedure. Section 4.1 specifies the benchmark variants, model backbones, baselines, and evaluation metrics. Appendix B provides implementation details, including hardware, memory initialization and construction, retrieval settings, and operational update criteria. Appendix D documents the component ablation configurations and the protocol for comparing frozen and continuously updated memory. Appendix E reports LLM call counts and token consumption for memory construction and question answering on the LongMemEval (S^*) variant in MemoryAgentBench.

AI USE STATEMENT

We used generative AI tools to assist with language editing and improve the clarity and readability of the manuscript. The authors take full responsibility for the final manuscript, including the accuracy of its scientific claims, experimental results, and conclusions.

ETHICS STATEMENT

SEEM^2 is evaluated on existing conversational-memory benchmarks and simulated tool-use tasks. Persistent storage of interaction histories and inference of behavioral profiles raise privacy concerns, while inaccurate or biased memories may influence subsequent responses and actions. The architecture links derived memories to source evidence, preserves unresolved factual conflicts, and applies evidence-based criteria to memory updates and procedural reuse, supporting inspection of stored information and its revision. Deployment with real users should provide clear disclosure and appropriate consent for memory collection and profiling, limit data retention and access, and enable users to inspect, correct, and delete both source records and derived memories. Reused procedures should remain subject to current user authorization and domain policies. The effectiveness of these deployment safeguards requires evaluation beyond the benchmark experiments reported here.

REFERENCES

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment. *arXiv preprint arXiv:2506.07982*, 2025. URL https://arxiv.org/abs/2506.07982.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready AI agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025. URL https://arxiv.org/abs/2504.19413.

Gordon V. Cormack, Charles L. A. Clarke, and Stefan Buettcher. Reciprocal rank fusion outperforms Condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 758–759. Association for Computing Machinery, 2009. doi: 10.1145/1571941.1572114. URL https://doi.org/10.1145/1571941.1572114.

DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026. URL https://arxiv.org/abs/2606.19348.

Jizhan Fang, Xinle Deng, Haoming Xu, Ziyan Jiang, Yuqi Tang, Ziwen Xu, Shumin Deng, Yunzhi Yao, Mengru Wang, Shuofei Qiao, Huajun Chen, and Ningyu Zhang. LightMem: Lightweight and efficient memory-augmented generation. In *International Conference on Learning Representations*, volume 2026, pp. 98706–98729, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/a05b72653ec5b473732129829ae04195-Abstract-Conference.html.

Jizhan Fang, Buqiang Xu, Zhixian Wang, Haoliang Cao, Xinle Deng, Baohua Dong, Hangcheng Zhu, Ruohui Huang, Gang Yu, Ying Wei, Guozhou Zheng, Feiyu Xiong, Haofen Wang, Huajun Chen, and Ningyu Zhang. Rethinking memory as continuously evolving connectivity. *arXiv preprint arXiv:2605.28773*, 2026b. URL https://arxiv.org/abs/2605.28773.

Runnan Fang, Yuan Liang, Xiaobin Wang, Jialong Wu, Shuofei Qiao, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. Memp: Exploring agent procedural memory. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 17490–17502. Association for Computational Linguistics, 2026c. doi: 10.18653/v1/2026.findings-acl.866. URL https://aclanthology.org/2026.findings-acl.866/.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. HippoRAG: Neurobiologically inspired long-term memory for large language models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 59532–59569. Curran Associates, Inc., 2024. doi: 10.52202/079017-1902. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/6ddc001d07ca4f319af96a3024f6dbd1-Abstract.html.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. From RAG to memory: Non-parametric continual learning for large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 21497–21515. PMLR, 2025. URL https://proceedings.mlr.press/v267/gutierrez25a.html.

Chuanrui Hu, Xingze Gao, Zuyi Zhou, Dannong Xu, Yi Bai, Xintong Li, Hui Zhang, Tong Li, Chong Zhang, Lidong Bing, and Yafeng Deng. EverMemOS: A self-organizing memory operating system for structured long-horizon reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 45836–45853, San Diego, California, United States, July 2026a. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.2125. URL https://aclanthology.org/2026.acl-long.2125/.

Yuanzhe Hu, Yu Wang, and Julian McAuley. Evaluating memory in LLM agents via incremental multi-turn interactions. In *International Conference on Learning Representations*, volume 2026, pp. 156259–156291, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/fd1eff9dd295df50a41f2521942fa31d-Abstract-Conference.html.

Jiazheng Kang, Mingming Ji, Zhe Zhao, and Ting Bai. Memory OS of AI agent. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 25961–25970. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.1318. URL https://aclanthology.org/2025.emnlp-main.1318/.

Dongfang Li, Zixuan Liu, Junmai Wang, Jiahe Huang, Fuhao Li, Bonian Jia, Baotian Hu, and Min Zhang. LycheeMemory V2: Efficient long-term memory for LLM agents via semantic segment-level consolidation. *arXiv preprint arXiv:2608.12990*, 2026. URL https://arxiv.org/abs/2608.12990.

Zhengxuan Lu, Dongfang Li, Yukun Shi, Beilun Wang, Longyue Wang, and Baotian Hu. Structured episodic event memory. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6125–6141, San Diego, California, United States, July 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.277. URL https://aclanthology.org/2026.acl-long.277/.

Wenquan Ma, Jiayan Nan, and WenLong Wu. What deserves memory: Adaptive memory distillation for LLM agents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 34789–34812, San Diego, California, United States, July 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.1607. URL https://aclanthology.org/2026.acl-long.1607/.

Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. ReasoningBank: Scaling agent self-evolving with reasoning memory. In *International Conference on Learning Representations*, volume 2026, pp. 94327–94354, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/980ea04d23d1f6908964eba2a74afe45-Abstract-Conference.html.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. LLMLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 963–981. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.57. URL https://aclanthology.org/2024.findings-acl.57/.

Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The Berkeley Function Calling Leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 48371–48392. PMLR, 2025. URL https://proceedings.mlr.press/v267/patil25a.html.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork. Official Qwen release blog, August 2026. URL https://qwen.ai/blog?id=qwen3.8.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. Zep: A temporal knowledge graph architecture for agent memory. *arXiv preprint arXiv:2501.13956*, 2025. doi: 10.48550/arXiv.2501.13956. URL https://arxiv.org/abs/2501.13956.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009. doi: 10.1561/1500000019. URL https://doi.org/10.1561/1500000019.

Bo Tang, Yang Zhang, Guomian Zhuang, Wenqiang Wei, Gaoyang Zheng, Lindong Xie, Yanchao Tan, Feiyu Xiong, Qingyu Yang, Edward Chung, and Zhiyu Li. From memory to skills: Evidence-grounded co-evolution governance for long-horizon LLM agents. *arXiv preprint arXiv:2607.16621*, 2026. doi: 10.48550/arXiv.2607.16621. URL https://arxiv.org/abs/2607.16621.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=ehfRiF0R3a.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 63897–63911. PMLR, 2025. URL https://proceedings.mlr.press/v267/wang25bx.html.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. LongMemEval: Benchmarking chat assistants on long-term interactive memory. In *International Conference on Learning Representations*, volume 2025, pp. 86809–86836, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/d813d324dbf0598bbdc9c8e79740ed01-Abstract-Conference.html.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-Mem: Agentic memory for LLM agents. In *Advances in Neural Information Processing Systems*, volume 38, pp. 17577–17604. Curran Associates, Inc., 2025. doi: 10.52202/085713-0593. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/19909c36f51abc4856b4560aff3d36d6-Abstract-Conference.html.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024. URL https://arxiv.org/abs/2406.12045.

Haozhen Zhang, Quanyu Long, Jianzhu Bao, Tao Feng, Weizhi Zhang, Haodong Yue, and Wenya Wang. MemSkill: Learning and evolving memory skills for self-evolving agents. *arXiv preprint arXiv:2602.02474*, 2026. doi: 10.48550/arXiv.2602.02474. URL https://arxiv.org/abs/2602.02474.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 Embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025. URL https://arxiv.org/abs/2506.05176.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17):19632–19642, March 2024. doi: 10.1609/aaai.v38i17.29936. URL https://ojs.aaai.org/index.php/AAAI/article/view/29936.

Xinping Zhao, Xinshuo Hu, Zifei Shan, Shouzheng Huang, Yao Zhou, Xin Zhang, Zetian Sun, Zhenyu Liu, Dongfang Li, Xinyuan Wei, Youcheng Pan, Yang Xiang, Meishan Zhang, Haofen Wang, Jun Yu, Baotian Hu, and Min Zhang. KaLM-Embedding-V2: Superior training techniques and data inspire a versatile embedding model. In

International Conference on Learning Representations, volume 2026, pp. 58958–58992, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/6082359ffba504842a41da0d6a3cffe2-Abstract-Conference.html.

A Notation Reference

Table A.1 collects the notation used in Sections 3.3–3.5. Here f denotes an assertion, s a skill, and ϵ a source experience (an episode or subgoal view). The superscript $+$ denotes an updated value.

Table A.1: Notation for semantic and procedural evolution, consolidation, and activation.

Symbol	Meaning
State updates	
$k(f)$	State key $(u_f, p_f, \sigma_f, \mu_f)$ formed from the subject, normalized attribute or relation, scope, and update mode.
$\tau_{\text{obs}}(f)$	Observation time of f, used to order implicit succession.
$C_{\text{state}}(f, f')$	Binary check of identity, scope, value and unit compatibility, and source eligibility.
Skills and evidence	
$\pi_s,\ \pi_\epsilon$	Stored procedural core of skill s and candidate core from experience ϵ, respectively.
α_s	Applicability conditions specifying when skill s can be used.
ℓ_s	Lifecycle state controlling the skill's eligibility for reuse.
$\mathcal{L}_s$	Evidence ledger indexed by source experience.
d_ϵ	Record of execution evidence, observed outcome, and provenance for experience ϵ.
$C_{\text{proc}}(\epsilon, s)$	Binary compatibility check for execution structure and applicability conditions.
$Q(\epsilon),\ Q_s$	Evidence-quality scores in $[0, 1]$ for the candidate and stored core, respectively.
δ_Q	Positive quality margin required to replace the stored core.
Consolidation and activation	
$\widetilde{\mathcal{M}}_{t+1},\ \mathcal{M}_{t+1}$	Memory after formation and updates, and after validated consolidation, respectively.
$\widehat{\Delta}_t,\ \Delta_t^{\text{valid}}$	Proposed consolidation operations and the validated operations applied to memory, respectively.
$q_{t+1},\ \mathcal{C}_{t+1}$	Next query or task, and the memory context assembled for it, respectively.
B	Shared token budget for retrieved memories and supplementary source evidence.

B Implementation Details

All experiments were conducted on Ascend 910C NPUs. For DeepSeek-based agents, we use the official DeepSeek-V4-Flash-0731 open-weight checkpoint (DeepSeek-AI, 2026).

LongMemEval (S^*). We construct a separate memory bank for each of the five histories and reuse it across the corresponding evaluation questions. Histories are ingested in units of at most 4,096 tokens while preserving session boundaries, timestamps, and speaker roles. Memory extraction uses conversation-oriented prompts with a 4,096-token input budget per batch. Dense representations use Qwen3-Embedding-8B with 4,096 dimensions, except for the KaLM Dense RAG baseline, which uses 896-dimensional embeddings. We use the official DeepSeek-V4-Pro-0813 weights (denoted DeepSeek-V4-Pro in the main text) to judge answer correctness (DeepSeek-AI, 2026). Table B.1 summarizes the memory and retrieval settings.

Table B.1: Memory and retrieval settings on LongMemEval (S^*). Chunk sizes are expressed in tokens. Retrieval counts refer to the indicated stage and memory type.

Method	Ingestion chunk size	Retrieval configuration
SEEM^2	4,096	Up to 8 entities, 3 profile claims, 30 facts, 18 scenes, and 20 source chunks
Full Context	Entire history	Complete source history
Dense RAG (Qwen3)	1,024	Top 10 chunks by cosine similarity
Dense RAG (KaLM)	1,024	Top 10 chunks by cosine similarity
Graphiti	4,096	Hybrid retrieval with reciprocal rank fusion and up to 100 candidates per entity, fact, and episode category
LightMem	4,096	Hybrid retrieval with 50 final memory entries
Mem0	4,096	Top 500 memories
A-MEM	4,096	10 retrieval seeds followed by one-hop expansion

For SEEM^2, candidates from dense retrieval and BM25 (Robertson & Zaragoza, 2009) are combined using a weighted variant of reciprocal rank fusion (Cormack et al., 2009), followed by bounded expansion along typed graph relations. Context selection adapts to the query, and provenance expansion adds supporting source passages within the same memory budget.

Our Graphiti baseline uses the open-source engine with episodic context. Its architecture is described by Rasmussen et al. (2025). LightMem applies LLMLingua-2 compression (Pan et al., 2024) and offline memory updates.

τ-bench. The main comparison evaluates 50 airline, 114 retail, and 114 telecom tasks per backbone. Memory starts empty and is maintained separately for each domain. In the online protocol, a completed task's visible trajectory and evaluation feedback update memory for subsequent tasks. The current SEEM^2 implementation retrieves up to six procedural skills within an injected memory-context budget of 8,192 tokens. The no-memory baseline retains the current dialogue, domain policy, and available tools while disabling persistent memory retrieval and writing. We adapt the authors' open-source implementations of AWM and ReasoningBank to τ-bench. For Qwen3.8-27B, task definitions are verified to match across methods, and success follows the official binary reward.

For AWM, we use the online setting (Wang et al., 2025), with a separate, initially empty workflow library per domain. Before each task, the latest library compatible with the domain, policy, and tool schemas is appended to the system prompt and remains fixed throughout the task. Complete workflows are included in library order within a 16,384-token estimated context budget (one token per four characters), without query-dependent retrieval, embeddings, top-k selection, or reranking. The library is re-induced from successful trajectories after each task with an official binary reward of 1, using a 32,768-token estimated induction input budget.

BFCL V4 Memory. We evaluate the benchmark's `memory_vector` setting, with 37 prerequisite conversations for memory construction followed by 155 evaluation questions across five scenarios. We retain the official memory-tool interface and answer-matching checker. Core Memory

supports up to 7 entries of 300 characters each, while Archival Memory supports up to 50 entries of 2,000 characters each. Core Memory is included in the agent's context, and archival information is retrieved as needed.

SEEM^2 and the baselines use their respective memory-construction and retrieval procedures within these capacity constraints. The principal retrieval settings are summarized in Table B.2.

Table B.2: Retrieval configurations on BFCL V4 Memory.

Method	Retrieval configuration
SEEM^2	Dense and lexical retrieval informed by structured-memory context
Vector Store	Cosine similarity over raw memory entries
Graphiti	Hybrid retrieval with reciprocal rank fusion and a retrieval upper bound of 100
LightMem	Hybrid retrieval with offline memory updates and a retrieval upper bound of 20
Mem0	Vector-memory retrieval with an upper bound of 100

Evaluation follows Section 4.1.

B.1 Instantiation of Memory Update Criteria

We specify update criteria and reference defaults for Section 3. Benchmark configurations appear above.

Compatibility and evidence quality. The compatibility predicates are binary checks on extracted attributes and their source links. C_{state} checks identity, scope, value and unit compatibility, and source eligibility. C_{proc} checks task requirements and compatibility of observed operations and state changes. Observation order and successful execution remain separate requirements in Eqs. 2 and 4.

The fixed quality score $Q \in [0, 1]$ summarizes how well a candidate is grounded in recorded actions, terminal-step evidence, executor attribution, and observed outcomes. The replacement margin δ_Q and other defaults are listed in Table B.3.

An isolated successful read, narrative content, or absence of an error is insufficient to initialize a provisional skill. Consolidation checks record existence and preserves evidential roles. Retired skills retain their supporting episodes. State and profile maintenance operates independently of model-assisted consolidation.

Table B.3: Reference defaults for memory-update decisions. Profile thresholds govern cross-event induction, which is enabled in the full system.

Component	Criterion or default
Core replacement	An eligible, compatible candidate replaces the core only if its quality score exceeds the stored score by more than 0.05.
Profile support	Within a 90-day observation window, support must include at least 3 events on 3 distinct days, spanning at least 14 days and drawn from at least 2 source groups. After the proposal and verification passes agree on the evidence, acceptance requires all of these conditions.
Choice tendencies	At least 4 voluntary choice opportunities with explicit alternatives are required. Supporting choices must account for at least 75% of the supporting and opposing opportunities.
Skill reuse	One eligible success permits provisional reuse, subject to relevance and applicability. With only one successful source, a matched failure or timeout triggers quarantine. Quarantined and retired skills are excluded from reuse.

Context budget and scope. For SEEM^2, retrieved memory records and supplementary source evidence share a context budget of $B = 8{,}192$ tokens. The full-system experiments retain direct

profile claims and enable cross-event profile induction, including on LongMemEval (S^*). Inferred claims follow the evidence checks in Section 3.3, while direct statements retain precedence.

C CASE STUDIES

C.1 FROM INTERACTIONS TO EXPERIENCE SCENES

We illustrate the experience-scene representation with a retail task from the τ-bench family. A customer asks to reinstate a cancelled order and insists on keeping the original order. The assistant explains that restoration is unavailable. The customer declines a new order and requests a supervisor. Table C.1 shows how task-local memory records this interaction in a dialogue scene describing the blocked request and a tool scene recording the subsequent human handoff.

Table C.1: Two experience scenes from an order-restoration interaction. Selected fields are condensed from the saved objects. "Not specified" denotes an unfilled field in the original record.

Scene attribute	**Scene 1: Restoration request**	**Scene 2: Human handoff**
Type	Dialogue	Tool
Participants	Customer and assistant	Assistant
Goal	Restore the original cancelled order.	Not specified.
Actions	The assistant explains the system limitation. The customer declines a new order and requests a supervisor.	The assistant invokes the human-handoff tool.
Execution	Not specified.	*Tool:* `transfer_to_human_agents`. *Summary argument:* The customer wants the original order restored, has declined a new order, and requests a supervisor.
Local outcome	*Failure.* Restoration remains blocked, and the customer requests escalation.	*Success.* The transfer is completed.
Source reference	Summary attached to the handoff call.	Summary attached to the handoff call.

Preserving local outcomes within an episode. The failure label applies to the unmet restoration goal, while success applies to the handoff operation. The representation therefore preserves the unresolved customer request alongside the completed escalation. The two scenes are ordered within one episode and linked to a provisional local handoff procedure. This procedure keeps a summary argument that can be populated from a current request, with references to the originating scenes and episode. Both scenes cite the handoff-call summary. The successful transfer is also corroborated by the subsequent tool response in the execution trace.

C.2 EVIDENCE INTEGRATION FOR LONG-TERM RECALL

We analyze three LongMemEval (S^*) examples to illustrate event participation, temporal grounding, and aggregation across conversations. Table C.2 summarizes responses from SEEM^2, Mem0, and Graphiti. The analysis relates these responses to the evidence supplied during answering.

Table C.2: Responses for three illustrative LongMemEval (S^*) cases. Responses are paraphrased. The reference accepts both elapsed and inclusive durations for the camping case.

Question	Reference	Mem0	Graphiti	SEEM^2
Weddings attended this year	3	Abstains	2	**3**
Duration of the Yosemite camping trip	2 or 3 days	Abstains	Abstains	**3 days**
Total gaming time	140 hours	Abstains	130 hours	**140 hours**

Preserving participation and event status. The wedding question requires identifying three attended events while excluding the user's own wedding, which is still being planned. The user had

attended Jen's wedding and was separately considering inviting her to participate in the user's future ceremony. SEEM^2 records the completed attendance and planned invitation as distinct scenes and supplies the supporting user statements. The baseline contexts mention Jen but lack the complete evidence establishing the user's attendance at her wedding. Graphiti counts the two other weddings, while Mem0 abstains. This case highlights the importance of preserving who participated in an event and whether it had occurred.

Grounding relative dates across conversations. The camping question links a departure statement on May 15 with a return statement on May 17, each referring to *today*. Interpreting these expressions relative to their own conversation dates gives two elapsed days, or three days including both endpoints. SEEM^2 presents a departure fact and a return-event scene together with their source statements and date cues. Mem0's context contains other travel memories but lacks the target trip's dated endpoints. Graphiti receives both dated statements yet abstains, stating that the duration is unspecified. The case illustrates both the complementary information retained in facts and scenes and the need to combine evidence across sessions after retrieval.

Aggregating distinct experiences. For the gaming question, the five recorded durations sum to $70+25+30+5+10 = 140$ hours. These include separate normal- and hard-difficulty playthroughs of *The Last of Us Part II*, with repeated mentions of the hard-difficulty playthrough counted once. All three methods' inputs contain the five durations. Graphiti's 10-hour *Celeste* record appears in an entity description, but its answer omits that contribution. Mem0 abstains. SEEM^2 supplies experience descriptions and original user statements alongside a calculation record linking each operand to its source. Its answer reports 140 hours for the five documented durations. This example highlights the distinction between making individual numbers available and organizing them into a sum over distinct experiences.

These cases compare complete answering pipelines, including their reader settings and auxiliary computation. The ablations in Section 4.3 evaluate individual memory components.

C.3 Grounding Tool Use in Rules and Experience

Case study of order changes. Figure 1 illustrates how semantic constraints and prior procedural experience guide the order of actions, while current tool results verify their outcomes. This example comes from a supplementary retail run with DeepSeek-V4-Flash, separate from the main benchmark evaluation.

Three airline examples from τ-bench illustrate operation applicability, cancellation eligibility, and adaptation to changing requests. Table C.3 contrasts the retrieved memory and its application in SEEM^2 and ReasoningBank.

Both methods receive the same base policy and obtain current task facts through interaction. In these examples, SEEM^2 refreshes its semantic and procedural context as the request evolves, while ReasoningBank retains lessons retrieved for the initial request.

Checking applicability before acting. The destination-change case shows how checking an operation's applicability can redirect execution toward a valid alternative. A tool's acceptance of an update alone does not establish compliance with the task policy.

Binding rules to current facts. The cancellation case separates retrieving a rule from applying it to current facts. Both agents have the relevant guidance and reservation details, yet their different eligibility judgments lead to different actions.

Maintaining relevance as the request evolves. The fare-selection case illustrates the value of keeping retrieved guidance aligned with the user's current goal. SEEM^2 refreshes its context for booking, while ReasoningBank retains guidance from the earlier request.

Table C.3: Memory content and its application on three illustrative airline tasks. Entries summarize the retrieved contexts available before each key decision and the subsequent execution.

Aspect	SEEM^2	ReasoningBank
Destination change. The user requests a different destination for an existing booking. Cancellation is permitted, and both agents find the same replacement flights.		
Retrieved memory	*Semantic rule:* Modifications must preserve the route endpoints. *Procedure:* Check applicability before updating and verify the result.	General guidance for reservation changes and flight selection.
Application and outcome	Uses cancellation and rebooking to change the destination.	The tool accepts a modification that leaves the booking's destination inconsistent with its flight segments.
Cancellation eligibility. The user asks to cancel an economy booking after the cancellation window has expired, with no travel insurance. Both agents observe the same reservation details.		
Retrieved memory	*Semantic rule:* Cancellation is subject to a time window and policy exceptions. *Procedure:* Check reservation details before cancelling.	Guidance to verify cancellation eligibility against current reservation details.
Application and outcome	Recognizes that the booking is ineligible and leaves it unchanged.	Treats the booking as eligible and cancels it.
Fare selection. The user's goal shifts from a refund request to booking the second-cheapest itinerary. Both agents receive the same search results.		
Retrieved memory	*Semantic rule:* Use the current request and tool results. *Procedure:* Compare current fares and confirm the selection.	Guidance for the earlier refund request remains after the goal changes.
Application and outcome	Books the requested second-cheapest itinerary after clarification.	Misquotes the fare and, after failed attempts and a revised user choice, books the cheapest itinerary.

D Additional Ablation Details

D.1 Component Ablation Configurations

For the τ-bench ablations with Qwen3.8-27B, *w/o Shared Memory Organization* retains semantic and procedural memory and their update mechanisms, but maintains separate scene structures, associations, and source references. Associations do not propagate between the two memories. Retrieval still uses both within the same total context budget. *w/o Semantic Memory* removes semantic memory while retaining procedural memory. *w/o Failure Updates* retains failure trajectories but prevents failed tasks from updating shared procedural memory, while successful-task updates remain enabled. These controls correspond to Table 4(b).

D.2 Frozen vs. Full Memory

Protocol. We compare frozen and continuously updated memory using Qwen3.8-27B. For each domain, approximately 20% of tasks build an initial memory (10 airline, 23 retail, and 23 telecom tasks). The remaining 40, 91, and 91 tasks, respectively, form disjoint evaluation sets. Memory starts empty for the build stage and is maintained separately for each domain. After this stage, the same memory snapshot initializes two isolated branches. *Frozen* permits retrieval but disables persistent updates, including changes to metadata that affect subsequent retrieval. *Full* retains the system's normal memory updates. The branches execute the same evaluation tasks in the same order, using matched model, retrieval, and task-execution settings. Each task resets its environment and dialogue state. Current-task context remains available to both branches. Full can use completed tasks' trajectories and feedback to update memory for later tasks, after the completed task has been scored.

The experiment uses one task split, with each branch evaluated once. The scores include only the 222 evaluation tasks and exclude the 56 build tasks. Overall success pools task counts across domains, and Δ denotes Full minus Frozen in percentage points (pp). The main comparison in Table 3 scores all 278 tasks.

Table D.1: Frozen vs. Full memory with Qwen3.8-27B. Cells report successful/evaluated tasks (success rate in %). Δ is reported in pp. Build tasks are excluded.

Domain	Frozen	Full	Δ (pp)
Airline	31/40 (77.50)	33/40 (82.50)	+5.00
Retail	71/91 (78.02)	75/91 (82.42)	+4.40
Telecom	63/91 (69.23)	73/91 (80.22)	+10.99
Overall	165/222 (74.32)	181/222 (81.53)	+7.21

Results. Full scores higher in all three domains (Table D.1), with the largest gain on telecom (+10.99 pp). Across domains, continued updates yield a net increase of 16 successful tasks, from 165/222 to 181/222. The comparison supports a benefit from continued memory evolution beyond reuse of the initial snapshot.

E COST ANALYSIS

Table E.1 compares the LLM usage of SEEM^2 with three memory-system baselines on LongMemEval (S^*), separating memory construction from question answering. We report call counts and token consumption. Implementation settings are given in Appendix B.

SEEM^2 requires 1,086 LLM calls in total, the fewest among the compared memory systems. This represents reductions of 39.0%, 27.1%, and 75.9% relative to A-MEM, Mem0, and LightMem, respectively. The construction stage contributes most to this advantage, as SEEM^2 builds its memory with 618 calls, approximately half the number required by A-MEM and Mem0. Compared with LightMem, it reduces construction calls by 85.3% while also using 6.3% fewer construction tokens.

Relative to A-MEM, SEEM^2 also achieves substantial token savings across both stages. Total token consumption decreases from 23.29M to 7.13M, a reduction of 69.4%. Memory construction uses 66.7% fewer tokens, and question answering uses 73.2% fewer tokens. During question answering, SEEM^2 additionally reduces the number of calls from 601 to 468, a reduction of 22.1%.

Table E.1: LLM usage on LongMemEval (S^*), split into memory construction and question answering. Totals sum the two stages. Token counts are reported without rounding.

	Memory construction		Question answering		Total	
Method	Calls	Tokens	Calls	Tokens	Calls	Tokens
SEEM^2	618	4,554,733	468	2,576,411	1,086	7,131,144
A-MEM	1,179	13,675,767	601	9,609,856	1,780	23,285,623
Mem0	1,189	3,981,419	301	2,211,442	1,490	6,192,861
LightMem	4,197	4,862,175	300	564,527	4,497	5,426,702

F LIMITATIONS

Our experiments evaluate SEEM^2 on long-term conversational memory and conversational tool use. Its generalization to more diverse domains and sustained deployment under changing task and tool conditions remains to be explored. Memory evolution also depends on the quality of extracted experience and the availability of execution feedback, which may be incomplete or noisy in open-ended environments. Future work can examine robust memory updates under these conditions and efficient memory maintenance as interaction histories grow.