

Propose, Verify, Commit: Evidence-Grounded Memory for Long-Horizon Multi-Actor Conversations

Zihao Lu^{1*}, Zhihang Yuan^{1*}, Lei Shi^{1†}

¹Alibaba Cloud Computing, Hangzhou, China
{qinian.lzh,yuanzhihang.yzh,juetian.sl}@alibaba-inc.com

Abstract

Long-horizon conversational memory is especially challenging in multi-actor settings, where relevant evidence is distributed across participants and contexts and previously established information may later be revised. We introduce EGMEMORY, which formulates long-horizon multi-actor memory as a *searchable state machine* that separates persistent message-level evidence from an explicit active state. At write time, adaptive state resolution and an evidence-grounded *propose-verify-commit* protocol govern how this state evolves. At read time, adaptive evidence navigation iteratively resolves the state and supporting evidence required for a query, using conversational structure to narrow the search space and lexical-semantic relevance to rank candidates. The system operates through prompting and tool use without memory-specific policy training. EGMEMORY achieves **68.2%** on GroupMemBench and **77.9%** on EverMemBench, outperforming the strongest evaluated baselines by **22.7** and **21.4** percentage points, respectively. It further reaches **73.6%** on the dyadic LoCoMo benchmark, demonstrating generalization beyond multi-actor conversations. We will release the codebase upon formal publication.

1 Introduction

Large language model agents increasingly operate over interactions that extend far beyond a single context window. Persistent memory has therefore become a central component of long-horizon agents, enabling them to retain past interactions, retrieve relevant experiences, and maintain information across sessions (Maharana et al., 2024; Wu et al., 2025). Recent memory systems have substantially improved how such histories are constructed, updated, organized, and retrieved (Chhikara et al., 2025; Xu et al., 2025; Li et al., 2025). Much prior

work and evaluation, however, has centered on dyadic or relatively localized interactions, where memories are typically associated with a single conversational context.

Long-horizon *multi-actor* conversations introduce a different memory regime. Information is distributed across participants, threads, and evolving contexts; similar statements may refer to different discussions; and previously established decisions may later be revised. Recent benchmarks expose these challenges directly. GroupMemBench (Yang et al., 2026) emphasizes group dynamics, speaker-grounded information, and audience-dependent language, while EverMemBench (Hu et al., 2026) further evaluates multi-party and multi-group collaboration with cross-context reasoning and temporally evolving information. In such settings, successful memory access requires more than finding semantically similar text: the system must determine *whose* evidence is relevant, *which* conversational state it refers to, and *whether* that state remains valid.

Existing work addresses these challenges from complementary directions: dynamic and structured memory approaches preserve evolving histories (Banerjee et al., 2026; Lu et al., 2026), while agentic retrieval supports iterative memory access (Trivedi et al., 2023; Packer et al., 2024; Yu et al., 2026; Yan et al., 2026). Multi-actor conversations couple these requirements into a single state-management problem: the system must decide when new evidence changes the active state and later recover the relevant state and supporting evidence when the query alone is insufficient.

We address this problem with EGMEMORY, which formulates conversational memory as a *searchable state machine*. Persistent evidence preserves interaction history, an evidence-grounded propose-verify-commit protocol governs active-state evolution, and adaptive evidence navigation resolves the state and supporting evidence required at read time.

*Equal contribution.

†Corresponding author.

Rather than overwriting earlier records, revisions change which evidence supports the active state while preserving the history they supersede; the same versioned memory is then available to both the writer and reader.

We evaluate EGMEMORY on two long-horizon multi-actor benchmarks. EGMEMORY achieves **68.2%** accuracy on GroupMemBench and **77.9%** on EverMemBench, outperforming the strongest evaluated baselines by **22.7** and **21.4** percentage points, respectively. We further evaluate EGMEMORY on the dyadic LoCoMo benchmark (Maharana et al., 2024) as a complementary test of generalization. EGMEMORY achieves **73.6%**, outperforming the strongest baseline by **4.3 points** and demonstrating that its benefits extend beyond multi-actor conversations.

Our main contributions are:

- We formulate long-horizon multi-actor memory as a **searchable state machine** that separates persistent conversational evidence from the active state it supports, allowing state evolution without erasing provenance or history.
- We realize this formulation through two complementary processes: **evidence-grounded state evolution**, where adaptive state resolution and propose–verify–commit govern memory updates, and **adaptive evidence navigation**, where the reader progressively resolves the state and supporting evidence required for a query.
- We establish leading performance on both long-horizon multi-actor benchmarks, outperforming the strongest evaluated baselines by **22.7** points on GroupMemBench and **21.4** points on EverMemBench, while demonstrating strong generalization to the dyadic LoCoMo setting.

2 Related Work

From persistent memory to evolving memory. Long-term memory systems extend language models beyond their immediate context by transforming interaction histories into persistent, retrievable representations. MemoryBank summarizes past interactions while selectively forgetting and reinforcing memories (Zhong et al., 2024), while SeCom shows that memory construction granularity and topic-coherent segmentation substantially affect retrieval quality (Pan et al., 2025). More recent

systems make memory itself increasingly dynamic. Mem0 reconciles newly extracted information with existing memories through addition, update, deletion, or no-op operations (Chhikara et al., 2025), and A-MEM evolves the attributes and links of interconnected memory notes (Xu et al., 2025). MemOS further generalizes memory management into a lifecycle spanning plaintext, activation-based, and parameter-level memories (Li et al., 2025). Together, these works show that memory construction is increasingly dynamic. As memories are updated or reorganized, preserving their provenance under change becomes a separate design problem.

Structured memory under change. Temporal and structured memory systems address this problem by preserving relations and histories among memories rather than treating them as independent records. Zep connects episodic sources with temporally evolving semantic facts and invalidates conflicting relations as new information arrives (Rasmussen et al., 2025). APEX-MEM preserves interaction history in an append-only temporal property graph and resolves evolving information through a multi-tool retrieval agent at query time (Banerjee et al., 2026). SEEM combines relational facts with provenance-linked episodic event frames and reconstructs coherent contexts through associative fusion and reverse provenance expansion (Lu et al., 2026). These approaches demonstrate the value of retaining provenance and temporal structure as memory evolves. However, preserving historical alternatives does not by itself specify when new conversational evidence is sufficient to advance the active state rather than merely coexist with earlier information.

Adaptive interaction with memory. Memory effectiveness also depends on how the model accesses what has been stored. IRCOT shows that multi-step retrieval benefits when later retrieval decisions are conditioned on information obtained in earlier reasoning steps (Trivedi et al., 2023). Within agent memory, MemGPT gives the foundation model tool-based control over hierarchical memory resources (Packer et al., 2024), while AgeMem and Memory-R1 explicitly learn policies for storing, updating, retrieving, and utilizing memories through reinforcement learning (Yu et al., 2026; Yan et al., 2026). Adaptive access has also been extended to structured memory: MAGMA performs policy-guided traversal over semantic, temporal, causal, and entity graphs (Jiang et al., 2026). These

works establish memory access as a first-class design dimension: what should be retrieved next may depend on evidence already observed, rather than on the original query alone.

Long-horizon multi-actor memory. Existing benchmarks progressively expose the limitations of these memory paradigms. LoCoMo evaluates memory over extended dyadic conversations (Maharana et al., 2024), while LongMemEval stresses information extraction, multi-session reasoning, temporal reasoning, knowledge updates, and abstention in sustained user–assistant interactions (Wu et al., 2025). More recently, GroupMemBench makes group dynamics, speaker-grounded belief tracking, and audience-adapted language explicit evaluation targets (Yang et al., 2026). EverMemBench further extends long-horizon evaluation to multi-party, multi-group collaboration with cross-context information, evolving decisions, and role-conditioned personas (Hu et al., 2026).

These multi-actor settings couple the preceding challenges: memory must preserve attributed evidence as its active state evolves, while the relevant state may itself need to be resolved through adaptive access. Motivated by this gap, we formulate long-horizon multi-actor memory as a *searchable state machine*. EGMEMORY preserves message-level evidence, advances state only through grounded transitions, and adaptively accesses the same memory when resolving both incoming updates and downstream queries. Unlike approaches that learn a dedicated memory-control policy, EGMEMORY relies directly on the tool-use capability of the foundation model and requires no memory-specific policy training, while state-changing operations remain subject to deterministic verification.

3 Methodology

3.1 Framework Overview

Figure 1 presents the overall architecture of EGMEMORY. We model long-horizon conversational memory as a **searchable state machine**: persistent evidence records what has occurred in the conversation, verified updates advance the currently active state, and adaptive memory access resolves the state or evidence relevant to each new message and query.

The framework couples two processes through a shared versioned memory. At write time, **evidence-grounded state evolution** resolves an incoming

message against the existing state and updates it through a propose–verify–commit protocol. The resulting **versioned state** preserves historical evidence while explicitly tracking what is currently valid. At read time, **adaptive evidence navigation** iteratively resolves the state and supporting evidence required for a query, using conversational structure to localize candidates and content relevance to determine what should be retrieved.

This formulation is guided by two invariants. First, revision is non-destructive: new evidence may change the active state but does not erase the evidence that preceded it. Second, state authority is explicit: a retrieved or semantically similar message does not by itself redefine what is currently valid. These invariants separate two questions that are conflated in flat memory stores—*what happened* and *what is valid now*—and give the writer and reader a shared interface over the same provenance-preserving memory.

3.2 Problem Formulation

We consider a long-horizon multi-actor conversation $\mathcal{C}_t = \{m_1, \dots, m_t\}$, where each message $m_i = (x_i, a_i, t_i, c_i)$ contains its textual content x_i , actor a_i , timestamp t_i , and conversational context c_i . The memory system must continuously incorporate new messages while supporting later queries whose evidence may be distributed across participants, contexts, and evolving conversational states.

We represent the memory at time t as

$$\mathcal{M}_t = (\mathcal{E}_t, \mathcal{S}_t, \mathcal{R}_t), \quad (1)$$

where $\mathcal{E}_t$ is the persistent evidence store, $\mathcal{S}_t$ contains the currently active states, and $\mathcal{R}_t$ records provenance and state-transition relations. Evidence is preserved over time, $\mathcal{E}_{t-1} \subseteq \mathcal{E}_t$, while $\mathcal{S}_t$ may evolve as new evidence establishes or supersedes previous states.

A defining property of our setting is that the relevant memory is not always identifiable from the input alone. We therefore model memory access adaptively. For an input x , starting from $H_0 = \emptyset$,

$$\begin{aligned} r_j &= \pi(x, H_{j-1}), \\ O_j &= \text{Access}(\mathcal{M}, r_j), \\ H_j &= H_{j-1} \cup O_j, \end{aligned} \quad (2)$$

where r_j is the memory request at round j , O_j is the returned evidence, and H_j accumulates the evidence observed so far. Thus, each subsequent

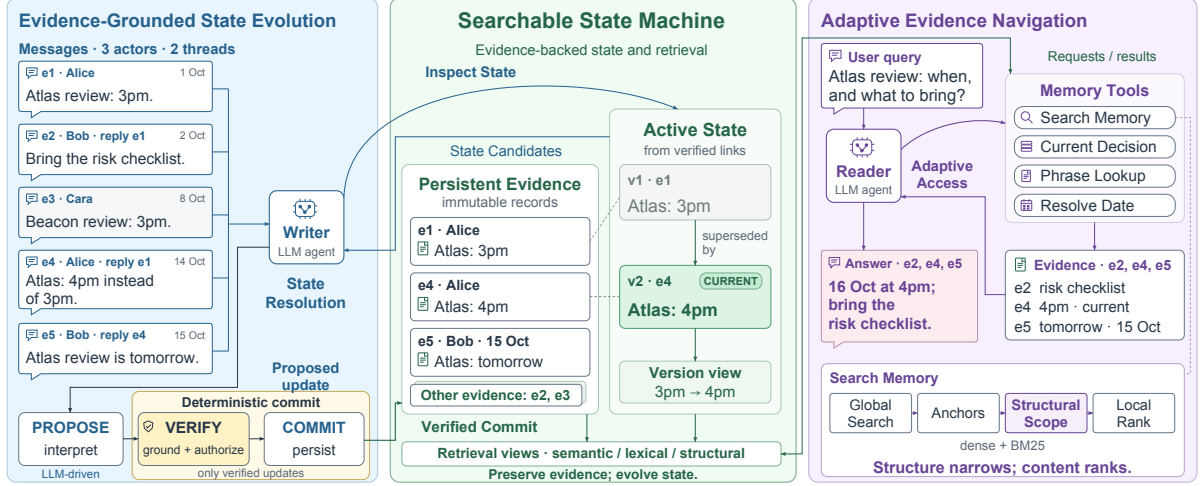


Figure 1: Overview of EGMEMORY. A writer resolves incoming messages against the existing memory and updates a versioned conversational state through a propose–verify–commit protocol. Persistent evidence preserves the underlying interaction history, while explicit state transitions maintain what is currently valid. At query time, a reader adaptively navigates the memory through repeated retrieval and inspection, using structure to narrow the search space and content to rank evidence.

access may depend on what previous accesses have already revealed.

Together, Eqs. 1–2 define a memory that separates persistent evidence from evolving active state while making both adaptively accessible at write and read time.

3.3 Grounded Memory Construction

Given a new message m_t , the first task is to determine how it relates to the existing memory state. A message may introduce a new state, restate an existing one, or revise information established earlier, yet this relation is often only implicit in the message. EGMEMORY therefore treats memory construction as an *adaptive state-resolution* process rather than a one-shot extraction step.

Before proposing an update, the writer may inspect the active memory multiple times. Following Eq. 2, each memory request can depend on state candidates returned in previous rounds. Let H_t^w denote the observations accumulated during this write-time resolution process. The writer can therefore progressively identify which existing state, if any, the incoming message refers to rather than committing to a state association from the message alone.

Once the relevant context has been resolved, EGMEMORY updates memory through a **propose–verify–commit** protocol. The writer first **proposes** a structured update

$$\Delta_t = \text{Propose}(m_t, H_t^w), \quad (3)$$

containing the message interpretation needed for state maintenance and retrieval. This includes the conversational topic, the relation to existing state, a state-bearing value when present, and a context-resolved memory representation.

The proposal is then **verified** against both the source message and the existing memory. State-bearing values must be supported by the source evidence, and a claimed revision must resolve to a valid prior state. We write

$$\hat{\Delta}_t = \text{Verify}(\Delta_t; m_t, \mathcal{M}_{t-1}) \quad (4)$$

for the verified update. This separates semantic interpretation from state authority: the language model proposes how a message should be interpreted, whereas deterministic evidence checks determine whether that interpretation is permitted to modify persistent state. In our implementation, this gate grounds the proposed state-bearing value and prior-state reference against the source evidence using deterministic overlap-based checks; exact thresholds are provided in Appendix B.

Finally, **Commit** converts the verified update into a canonical evidence record e_t and updates the memory state,

$$(e_t, \mathcal{M}_t) = \text{Commit}(\mathcal{M}_{t-1}, \hat{\Delta}_t). \quad (5)$$

The committed record preserves its retrieval representation together with conversational provenance such as actor, timestamp, topic, and structural context. Thus, the output of memory construction is

not an isolated summary, but grounded evidence from which the conversational state can be updated and later reconstructed.

3.4 Versioned State Maintenance

The committed evidence records form the persistent layer $\mathcal{E}_t$ introduced in Section 3.2. Over this evidence, EGMEMORY maintains the active-state set $\mathcal{S}_t$ and the relations $\mathcal{R}_t$ that connect current states to their provenance and history. This constitutes the central *searchable state machine* in Figure 1.

For each established conversational state, EGMEMORY maintains the evidence record that currently supports its active value. A verified update can establish a new state, restate an existing state, or revise the active value. When a revision is committed, the new evidence record explicitly supersedes the previous holder and becomes the new active state. Crucially, the previous record remains in $\mathcal{E}_t$ rather than being overwritten.

Let

$$e^{(k)} \prec e^{(k+1)} \quad (6)$$

denote that evidence record $e^{(k+1)}$ supersedes $e^{(k)}$. Repeated revisions then induce a version trajectory

$$e^{(1)} \prec e^{(2)} \prec \dots \prec e^{(K)}, \quad (7)$$

where all records remain available as historical evidence while $e^{(K)}$ supports the currently active state. The memory therefore preserves both *what was previously established* and *what is valid now*. This distinction is important because current-state queries and historical queries require different views of the same evidence: the former should expose the active holder, while the latter must retain access to the states it superseded.

This representation also makes state evolution directly searchable. A query about the current value can access the active holder, whereas a query about how a decision changed can follow its version trajectory back to earlier evidence. For longer trajectories, EGMEMORY additionally constructs a compact derived view that places the relevant versions in a single retrieval unit while retaining provenance to the underlying records. These derived views are retrieval aids only: the canonical message-level records remain unchanged and retain the provenance used for grounding and reconstruction.

The state machine therefore serves as the shared interface between construction and retrieval. At write time, the writer searches it to resolve which

state an incoming message refers to before committing an update. At read time, the same evidence, active-state assignments, and transition relations define the memory space explored by the reader.

3.5 Adaptive Evidence Navigation

At query time, the relevant conversational state and its supporting evidence may not be identifiable from the query alone. The reader therefore treats retrieval as an *adaptive evidence navigation* process rather than a single retrieve–read operation: observations from earlier accesses can guide subsequent memory requests.

Following Eq. 2, let H_q^r denote the evidence accumulated during read-time interaction. Each new request is conditioned on the query and the evidence observed so far, allowing the reader to refine an ambiguous search, inspect the current state, or gather additional supporting evidence.

The primary search operation uses a two-pass retrieval procedure. The first pass performs global lexical–semantic retrieval and selects a small set of high-relevance anchors. These anchors induce a query-conditioned candidate pool from topic buckets and provenance neighborhoods, including reply and version relations; optional participant constraints can further restrict the search scope. The second pass reapplies lexical–semantic retrieval within this restricted pool to recover additional evidence.

Conversational structure is therefore used to determine *where to search*, rather than as an independent relevance signal. Structural proximity may indicate where useful evidence resides, but does not by itself imply relevance to the current query. Accordingly, EGMEMORY uses structure to narrow the candidate space and content to rank the resulting evidence.

Beyond general memory search, the reader can invoke targeted operations for current-state inspection, exact lexical evidence, and temporal resolution. These operations share the same interaction history, so evidence surfaced by one tool can guide subsequent access to another. The process terminates once sufficient grounded evidence has been collected, or when the available memory does not support a justified answer.

The final response is generated from the accumulated evidence,

$$\hat{y} = \text{Answer}(q, H_q^r), \quad (8)$$

rather than from a single fixed retrieval result.

4 Experiments

4.1 Overview

Our evaluation focuses primarily on long-horizon *multi-actor* memory. We use GroupMemBench (Yang et al., 2026) and EverMemBench (Hu et al., 2026) as the two primary benchmarks, while LoCoMo (Maharana et al., 2024) provides a complementary test of generalization to conventional dyadic conversations.

We evaluate the system at three levels. We first compare end-to-end performance against one-shot retrieve-then-read controls and representative long-term memory systems, separating the benefit of persistent stateful memory from retrieval alone. We then examine category-level performance across memory capabilities, and finally ablate the write- and read-side mechanisms of EGMEMORY.

4.2 Experimental Setup

Datasets. GroupMemBench contains 745 questions spanning Multi-hop, Temporal, Abstention, Knowledge update, Term ambiguity, and User implicit reasoning, directly stressing distributed evidence and evolving conversational state. EverMemBench contains 2,400 questions covering *Fine-Grained Recall*, *Memory Awareness*, and *Profile Understanding*, providing complementary evaluation of cross-context retrieval, updates, and participant characteristics. We additionally evaluate on 1,986 LoCoMo questions spanning Single-hop, Adversarial, Temporal, Multi-hop, and Open-domain reasoning; LoCoMo serves as a generalization benchmark rather than a primary optimization target.

Baselines and retrieval controls. Our one-shot controls use BM25, dense retrieval with text-embedding-v4, or reciprocal-rank fusion of the two (Hybrid RAG), retrieving the top-10 messages before a single answering step. These controls test whether the gains of EGMEMORY can be explained by conventional retrieval without iterative access or explicit state maintenance. We additionally compare against Mem0 (Chhikara et al., 2025), A-MEM (Xu et al., 2025), MemOS (Li et al., 2025), and MemoryBank (Zhong et al., 2024); for MemoryBank, we report variants with (MB-F) and without (MB-NF) forgetting. All systems use the same benchmark-specific answering and judging protocol.

Implementation. We use qwen3.7-max for memory construction and question answering, Kimi-K3 as the evaluation judge, text-embedding-v4 for dense retrieval, and qwen3-rerank for reranking. The writer and reader operate through prompting and tool calls without memory-specific training or fine-tuning. The one-shot controls use the same answering model as EGMEMORY, while EGMEMORY additionally performs the adaptive write- and read-time interactions described in Section 3. Full implementation details and hyperparameters are provided in Appendix B.

Evaluation protocol. We report accuracy as the primary metric together with category-level results for each benchmark and 95% Wilson confidence intervals for aggregate EGMEMORY results. Final answers from all systems are evaluated with Kimi-K3 under the same benchmark-specific judging protocol. Controlled component ablations are conducted on GroupMemBench and reported as absolute percentage-point changes from the full system.

4.3 Overall Performance

Table 1 summarizes the overall results. EGMEMORY achieves the strongest performance on all three benchmarks, reaching **68.2%** on GroupMemBench and **77.9%** on EverMemBench, **22.7** and **21.4** points above the strongest evaluated baselines, respectively. The strongest one-shot controls reach only 45.5% and 32.1%, showing that the gains extend beyond a simple retrieve-then-read pipeline. On LoCoMo, EGMEMORY reaches **73.6%** (95% CI: 71.6–75.5), exceeding the strongest baseline by **4.3 points** and demonstrating dyadic generalization.

4.4 Performance across Memory Capabilities

We next examine category-level performance to characterize how the overall gains are distributed across different memory capabilities.

4.4.1 Multi-Actor Memory

GroupMemBench. Table 2 shows that EGMEMORY achieves the best performance in five of the six categories. Relative to the strongest memory-system baselines, the gains are **+39.0** on Multi-hop, **+33.9** on Temporal, **+22.7** on Term ambiguity, **+20.4** on User implicit, and **+15.9** on Knowledge update. These improvements are largest on tasks requiring distributed, context-dependent, or evolving

Method	Long-Horizon Multi-Actor Memory		Dyadic Generalization
	GroupMemBench	EverMemBench	LoCoMo
<i>One-shot retrieval controls</i>			
Dense RAG	35.8	27.8	39.4
BM25 RAG	45.5	30.4	47.3
Hybrid RAG	43.9	32.1	48.1
<i>Long-term memory systems</i>			
MemoryBank (MB-F)	30.7	1.9	28.3
MemoryBank (MB-NF)	40.7	54.2	68.6
MemOS	20.3	25.0	56.2
A-MEM	40.9	56.5	63.5
Mem0	42.9	51.3	69.3
EGMEMORY	68.2	77.9	73.6
95% CI	[64.8, 71.4]	[76.2, 79.5]	[71.6, 75.5]
Δ vs. best baseline	+22.7	+21.4	+4.3

Table 1: Overall accuracy (%) on GroupMemBench, EverMemBench, and LoCoMo. One-shot controls retrieve the top-10 messages using the same answering and judging models. Confidence intervals are 95% Wilson intervals for EGMEMORY; Δ is the absolute improvement over the strongest baseline.

Category	n	MB-F	MB-NF	MemOS	A-MEM	Mem0	EGMEMORY
Multi-hop	182	19.2	33.5	3.8	36.3	35.2	75.3
Temporal	162	18.5	32.1	3.1	29.6	46.3	80.2
Abstention	139	90.6	87.1	96.4	83.5	92.8	79.1
Knowledge update	107	12.1	30.8	0.0	29.0	16.8	46.7
Term ambiguity	106	11.3	12.3	1.9	22.6	20.8	45.3
User implicit	49	26.5	46.9	6.1	40.8	24.5	67.3

Table 2: Category-level accuracy (%) on GroupMemBench. MB-F and MB-NF denote MemoryBank with forgetting enabled and disabled, respectively.

evidence. Abstention is the only exception. This is consistent with a coverage–abstention trade-off: the reader is designed to continue searching for supporting evidence before refusing, making it less conservative on no-answer cases.

EverMemBench. Table 3 reports results across *Fine-Grained Recall*, *Memory Awareness*, and *Profile Understanding*. EGMEMORY leads on eight of the nine tasks, including **74.7%** on Multi-hop Trajectory and over 90% on all three Memory Awareness tasks. EGMEMORY leads on eight of the nine tasks. The largest gain occurs on Multi-hop Trajectory, where accuracy rises from 3.6% to **74.7%**, while all three Memory Awareness tasks exceed 90%. These results show that the gains extend beyond factual recall to reasoning over distributed and evolving conversational information. Style is the only exception, suggesting that distributed stylistic preferences are less directly captured by the current state-oriented memory representation.

4.4.2 Generalization to Dyadic Conversations

We additionally evaluate EGMEMORY on LoCoMo as a complementary test of whether a memory architecture designed for multi-actor interactions generalizes to conventional dyadic conversations.

As shown in Table 4, EGMEMORY achieves the best performance on Single-hop, Temporal, Multi-hop, and Open-domain questions. Relative to the strongest memory-system baselines, it improves Single-hop by **6.3** points, Multi-hop by **6.7**, Temporal by **1.6**, and Open-domain by **2.1**. Adversarial questions remain the only category where EGMEMORY does not lead. Together with the overall LoCoMo result, these findings indicate that the proposed architecture generalizes beyond the multi-actor setting without being specifically optimized for dyadic memory.

4.5 Ablation Study

Read-time evidence resolution. Restricting the reader to a single interaction round causes the largest drop (−7.9 points), followed by removing access to versioned state (−5.0) and cross-encoder

Task	n	MB-F	MB-NF	MemOS	A-MEM	Mem0	EGMEMORY
<i>Fine-Grained Recall</i>							
Single-hop Retrieval	213	1.9	84.0	25.8	82.2	79.8	97.2
Multi-hop Trajectory	249	0.0	2.4	0.4	3.6	1.6	74.7
Temporal Duration	300	0.3	14.7	7.3	21.7	11.3	45.0
<i>Memory Awareness</i>							
Constraint	402	4.2	70.1	43.5	71.4	70.6	92.8
Proactivity	427	1.2	77.0	32.6	80.6	73.3	96.7
Update	268	3.0	86.9	35.1	88.1	88.8	97.4
<i>Profile Understanding</i>							
Style	176	0.0	51.1	17.0	52.3	36.4	32.4
Skill	169	1.8	28.4	23.7	34.9	29.6	58.0
Role	196	3.6	46.4	21.9	45.4	37.8	71.4

Table 3: Category-level accuracy (%) on EverMemBench, grouped by Fine-Grained Recall, Memory Awareness, and Profile Understanding.

Category	n	MB-F	MB-NF	MemOS	A-MEM	Mem0	EGMEMORY
Single-hop	841	9.8	73.6	57.3	66.5	75.5	81.8
Adversarial	446	95.1	86.3	85.9	84.8	86.1	85.0
Temporal	321	5.3	67.0	38.9	60.4	70.1	71.7
Multi-hop	282	9.2	36.5	33.3	32.3	36.9	43.6
Open-domain	96	14.6	41.7	34.4	40.6	29.2	43.8

Table 4: Category-level accuracy (%) on LoCoMo, used as the dyadic transfer evaluation.

Configuration	Overall	Δ
EGMEMORY	68.2	–
<i>Reader-side</i>		
w/o Reranker	63.5	−4.7
w/o Versioned state	63.2	−5.0
Single-round reader	60.3	−7.9
<i>Writer-side</i>		
w/o Evidence verification	65.9	−2.3

Table 5: Ablation on GroupMemBench. Δ denotes the absolute accuracy change from the full system.

reranking (−4.7). Together, these results support the three components of adaptive evidence navigation: iterative access, explicit state evolution, and content-based relevance ranking.

Write-time state control. Removing evidence grounding for proposed state values and prior-state references lowers accuracy by **2.3 points**. Its smaller effect is consistent with its role as a state-integrity gate: verification controls whether model-proposed interpretations may modify persistent state rather than directly improving retrieval ranking.

5 Conclusion

We studied long-horizon memory for multi-actor conversations, where evidence is distributed across

participants and contexts while previously established information may continue to evolve. We introduced EGMEMORY, which formulates this setting as a *searchable state machine*: persistent message-level evidence preserves conversational history, while an explicit active state tracks what is currently valid.

EGMEMORY realizes this formulation through two complementary processes. At write time, adaptive state resolution and an evidence-grounded propose–verify–commit protocol govern how persistent state evolves. At read time, adaptive evidence navigation resolves the relevant state and supporting evidence through iterative memory access. The resulting system operates with a tool-capable foundation model and requires no memory-specific policy training.

Across the two primary multi-actor benchmarks, EGMEMORY outperforms the strongest evaluated baselines by **22.7** points on GroupMemBench and **21.4** points on EverMemBench, while reaching **73.6%** on the dyadic LoCoMo benchmark. Together, these results support a simple principle for long-horizon memory: *preserve evidence, evolve state, and make both searchable*.

Limitations

EGMEMORY has several limitations. First, deterministic verification relies on source-grounding heuristics rather than full semantic entailment, so strongly paraphrased revisions may be rejected while lexical overlap alone cannot guarantee semantic correctness. Second, the explicit-state abstraction is better suited to factual and decision-like information than to diffuse behavioral signals such as style, consistent with the remaining weakness on EverMemBench. Third, adaptive write- and read-time interaction requires more model calls than one-shot retrieval, and this work does not optimize latency or token cost. Our evaluation also uses a single foundation-model family and English-language benchmarks, leaving cross-model, multilingual, and noisy speaker-attribution settings for future work. Finally, persistent multi-party memory can contain sensitive information; practical deployments should pair retention with appropriate access-control, deletion, and privacy policies.

References

- Pratyay Banerjee, Masud Moshtaghi, Shivashankar Subramanian, Amita Misra, and Ankit Chadha. 2026. [APEX-MEM: Agentic semi-structured memory with temporal reasoning for long-term conversational AI](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16470–16489, San Diego, California, United States. Association for Computational Linguistics.
- Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. 2025. [Mem0: Building production-ready AI agents with scalable long-term memory](#). In *ECAI 2025 - 28th European Conference on Artificial Intelligence, 25-30 October 2025, Bologna, Italy - Including 14th Conference on Prestigious Applications of Intelligent Systems (PAIS 2025)*, volume 413 of *Frontiers in Artificial Intelligence and Applications*, pages 2993–3000. IOS Press.
- Chuanrui Hu, Tong Li, Xingze Gao, Hongda Chen, Yi Bai, Dannong Xu, Tianwei Lin, Xiaohong Li, Yunyun Han, Jian Pei, and Yafeng Deng. 2026. [Evaluating long-horizon memory for multi-party collaborative dialogues](#). In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2, KDD ’26*, page 9082–9090, New York, NY, USA. Association for Computing Machinery.
- Dongming Jiang, Yi Li, Guanpeng Li, and Bingzhe Li. 2026. [MAGMA: A multi-graph based agentic memory architecture for AI agents](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 36848–36865, San Diego, California, United States. Association for Computational Linguistics.
- Zhiyu Li, Chenyang Xi, Chunyu Li, Ding Chen, Boyu Chen, Shichao Song, Simin Niu, Hanyu Wang, Jiawei Yang, Chen Tang, Qingchen Yu, Jiahao Zhao, Yezhaohui Wang, Peng Liu, Zehao Lin, Pengyuan Wang, Jiahao Huo, Tianyi Chen, Kai Chen, and 20 others. 2025. [Memos: A memory os for ai system](#). *Preprint*, arXiv:2507.03724.
- Zhengxuan Lu, Dongfang Li, Yukun Shi, Beilun Wang, Longyue Wang, and Baotian Hu. 2026. [Structured episodic event memory](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6125–6141, San Diego, California, United States. Association for Computational Linguistics.
- Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. 2024. [Evaluating very long-term conversational memory of LLM agents](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13851–13870, Bangkok, Thailand. Association for Computational Linguistics.
- Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. 2024. [Memgpt: Towards llms as operating systems](#). *Preprint*, arXiv:2310.08560.
- Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Xufang Luo, Hao Cheng, Dongsheng Li, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Jianfeng Gao. 2025. [Secom: On memory construction and retrieval for personalized conversational agents](#). In *International Conference on Learning Representations*, volume 2025, pages 91851–91885.
- Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. 2025. [Zep: A temporal knowledge graph architecture for agent memory](#). *Preprint*, arXiv:2501.13956.
- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. [Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037, Toronto, Canada. Association for Computational Linguistics.
- Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. 2025. [Longmemeval: Benchmarking chat assistants on long-term interactive memory](#). In *International Conference on Learning Representations*, volume 2025, pages 86809–86836.
- Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. 2025. [A-mem: Agentic memory for llm agents](#). In *Advances in Neural*

Information Processing Systems, volume 38, Main Conference, pages 17577–17604. Curran Associates, Inc.

Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Jinhe Bi, Kristian Kersting, Jeff Z. Pan, Hinrich Schuetze, Volker Tresp, and Yunpu Ma. 2026. [Memory-r1: Enhancing large language model agents to manage and utilize memories via reinforcement learning](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12805–12825, San Diego, California, United States. Association for Computational Linguistics.

Jingbo Yang, Kwei-Herng Lai, Xiaowen Wang, Shiyu Chang, Yaar Harari, and Evgeniy Gabrilovich. 2026. [Groupmembench: Benchmarking llm agent memory in multi-party conversations](#). *Preprint*, arXiv:2605.14498.

Yi Yu, Liuyi Yao, Yuexiang Xie, Qingquan Tan, Jiaqi Feng, Yaliang Li, and Libing Wu. 2026. [Agentic memory: Learning unified long-term and short-term memory management for large language model agents](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 21457–21483, San Diego, California, United States. Association for Computational Linguistics.

Wanjuan Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2024. [Memorybank: enhancing large language models with long-term memory](#). In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI’24/IAAI’24/EAAI’24. AAAI Press.

A Appendix

B Implementation Details

This section provides implementation details and hyperparameters for the experiments in Section 4. Unless otherwise specified, the same configuration is used across all reported evaluations.

B.1 Model Configuration

We use qwen3.7-max for the memory writer and answering agent, Kimi K3 as the evaluation judge, text-embedding-v4 for dense representations, and qwen3-rerank for cross-encoder reranking. Final answers from all evaluated systems are judged by Kimi K3 under the same benchmark-specific evaluation protocol.

Component	Model
Memory writer	qwen3.7-max
Answering agent	qwen3.7-max
Evaluation judge	Kimi K3
Embedding model	text-embedding-v4
Reranker	qwen3-rerank

Table 6: Model configuration used in the reported experiments.

B.2 One-Shot Retrieval Controls

The one-shot controls in Table 1 isolate the effect of conventional retrieve-then-read pipelines from the persistent and adaptive memory mechanisms of EGMEMORY. All controls retrieve ten messages and provide them directly to the same qwen3.7-max answering agent used by EGMEMORY.

BM25 RAG. The lexical control ranks the verbatim message content using BM25 and returns the ten highest-ranked messages.

Dense RAG. The dense control embeds the query and message content using text-embedding-v4 and retrieves the ten messages with the highest embedding similarity.

Hybrid RAG. The hybrid control combines BM25 and dense retrieval using reciprocal-rank fusion (RRF) with constant $k_0 = 60$. Each retrieval channel contributes up to 100 candidates before fusion, after which the ten highest-ranked messages are passed to the answering agent.

For all three controls, retrieval operates over the original message content rather than the rewritten memory representation used by EGMEMORY. Retrieved messages retain their corpus-native speaker and conversational metadata when presented to the answering agent. No iterative memory access or explicit versioned state is used.

B.3 Memory Construction

Each retained message is represented by one canonical message-level record. The writer predicts seven fields: keep, topic, phase, verdict, value_quote, prior_ref, and rewritten. The verdict is one of {new, restate, revision, null} and specifies whether the message establishes, repeats, revises, or does not express a memory state.

The language model proposes these fields, while state mutation is performed deterministically. A proposed value_quote must be grounded

Hyperparameter	Value
Value-grounding threshold	0.9
Prior-reference grounding threshold	0.8
Maximum writer-agent loops	6
Maximum registry searches	3
Maximum rewrite retries	1
Registry retrieval depth	6
Slot-subject length	60 chars

Table 7: Memory-construction hyperparameters.

in the source message with a minimum token-overlap threshold of 0.9. For revision references, `prior_ref` uses a grounding threshold of 0.8. A decision without a grounded value is not committed as an authoritative decision state.

Memory construction maintains a channel-specific registry of active decision states. The writer may retrieve up to six candidate states from this registry using lexical and dense retrieval. A revision is committed only when it can be associated with an existing active state; otherwise, the message is treated as establishing a new state rather than revising an unobserved one.

Registry retrieval. The write-side registry uses both lexical and dense signals. The dense channel compares the writer’s retrieval query with the rewritten representation of the current state holder, whereas the lexical channel operates on source-message tokens and the stored state description. The two rankings are combined using weighted reciprocal-rank fusion.

The lexical channel uses BM25 with $k_1 = 1.5$ and $b = 0.75$. The reciprocal-rank fusion constant is 60, with lexical weight 2.0 for write-side registry retrieval.

Version relations. A committed revision creates a supersedes relation to the previous active state. Revision links are assigned one of two confidence levels. A link is high-confidence when the revision message contains a grounded reference to the value it replaces; otherwise, it is assigned the lower confidence tier.

At read time, only high-confidence revision links are retained as authoritative state transitions. When a revision edge is rejected, its associated stale-state payload is also removed, preventing an uncertain relation from marking an earlier value as definitively obsolete.

Version-chain materialization. A sequence of validated revisions is additionally represented by

Hyperparameter	Value
RRF constant	60
Dense/BM25 fusion depth	200
Cross-encoder reranking depth	50
Maximum reranker document length	4,000 chars
Initial retrieved passages	5
Structural anchors	3
Second-pass passages	10
Maximum passages per tool return	12
Maximum answering-agent rounds	10

Table 8: Read-side retrieval and answering hyperparameters.

a derived retrieval unit containing the rewritten representations of all states in chronological order. The canonical message-level records remain unchanged. The embedding of the derived unit is obtained by averaging and renormalizing the existing message embeddings, without an additional embedding-model call.

B.4 Retrieval Configuration

The read-side retriever follows three stages: SCOPE, FUSION, and SELECTION.

Scope. Optional metadata constraints over channel, topic, phase, and author are applied as hard candidate filters. When multiple constraints are specified, their candidate sets are intersected. Without an explicit scope, retrieval operates over the full memory store.

Fusion. Within the candidate pool, dense and BM25 rankings are combined using reciprocal-rank fusion with equal channel weights. Both retrieval channels operate on the rewritten memory representation. Up to 200 candidates are retrieved from each channel before fusion.

Selection. The top 50 fused candidates are reranked using `qwen3-rerank`. Each candidate is truncated to at most 4,000 characters before reranking, and the reranked order determines the final retrieval ranking.

B.5 Structure-Constrained Retrieval

For each memory search, the first-stage retriever returns the five highest-ranked passages. The top three are used as anchors for an automatic structure-constrained second pass.

The second-pass candidate pool is formed from the union of the anchors’ topic buckets and provenance neighborhoods. The latter includes reply-chain context, sibling replies, and forward and

backward version relations. Dense and lexical retrieval are then reapplied within this restricted pool, returning up to ten additional passages.

Structural proximity is not added as an independent relevance score. Instead, structure determines the candidate set, while lexical and semantic matching determine the ordering within that set. Thus, conversational structure controls *where* the system searches, while content relevance controls *what* is returned.

B.6 Read-Side Tools

The answering agent has access to four memory operations:

- `search_memory`: hybrid dense–lexical retrieval with optional topic, phase, and author constraints;
- `lookup_phrase`: lexical lookup for localized expressions that may be poorly represented by passage-level embeddings;
- `get_current_decision`: retrieves the current state associated with a versioned decision chain; and
- `resolve_date`: deterministically resolves relative temporal expressions from message or phase timestamps.

Retrieved evidence is presented chronologically after relevance-based selection. Passages already shown for the same question are suppressed in subsequent retrieval calls, allowing later accesses to expose additional evidence.

B.7 Evidence Representation

Dense retrieval, read-side BM25 retrieval, reranking, and the passage body presented to the answering agent operate on the `rewritten` representation. The original source message remains in the persistent evidence store for grounding and auditability but is not normally shown directly to the answering agent.

Each retrieved passage additionally exposes structured metadata, including speaker, role, channel, topic, phase, timestamp, reply information, and version status. For validated revisions, a passage may explicitly mark a superseded value as no longer current.

B.8 Ablation Configurations

The ablations in Table 5 modify one part of the reported system while retaining the remaining configuration.

w/o Reranker. The cross-encoder reranking stage is disabled. Candidates are returned according to the fused dense–BM25 ranking; structure-constrained retrieval, versioned memory, and adaptive multi-round tool use remain unchanged.

w/o Versioned state. The writer constructs the normal memory, but read-time access to versioned structure is removed. Specifically, derived version-chain retrieval units and supersedes relations are excluded, while the underlying message records, reply structure, reranker, and adaptive reader remain available.

Single-round reader. The answering agent is restricted to one interaction round instead of the default maximum of ten. The same tools and retrieval operations remain available within that round; the ablation therefore removes iterative adaptation across successive observations rather than changing the underlying retriever.

w/o Evidence verification. Source-grounding thresholds for the proposed `value_quote` and `prior_ref` are set to zero, allowing non-empty model-proposed values and references to pass the grounding gate. The remaining verdict, state-association, and commit constraints are retained. This ablation therefore targets evidence grounding rather than disabling the complete state-update procedure.

B.9 Persistence and Reproducibility

Memory construction commits each message and any derived version-chain update within the same database transaction. The persistent message records are sufficient to reconstruct the active-state registry without additional language-model inference when an interrupted run is resumed.

To prevent silent configuration drift, the implementation records fingerprints for the write-side code and the writer and judging prompts. The question-answering pipeline also probes the reranker before evaluation and records reranking failures. Evaluation failures are retained and scored as incorrect rather than being removed from the result set.

The principal frozen constants used in the reported configuration are summarized in Tables 7 and 8.