MemWeaver: Weaving Hybrid Memories for Traceable Long-Horizon Agentic Reasoning

Juexiang Ye[1], Xue Li[1], Xinyu Yang[1], Chengkai Huang[2,3], Lanshun Nie[1], Lina Yao[2,4], Dechen Zhan[1]

[1]Harbin Institute of Technology, [2]The University of New South Wales, [3]Macquarie University, [4]CSIRO's Data61
{25B303035, 2021211983}@stu.hit.edu.cn,
{chengkai.huang1, lina.yao}@unsw.edu.au,
{lixuecs, nls}@hit.edu.cn, {dechenzhanhit}@gmail.com

Abstract

Large language model-based agents operating in long-horizon interactions require memory systems that support temporal consistency, multi-hop reasoning, and evidence-grounded reuse across sessions. Existing approaches largely rely on unstructured retrieval or coarse abstractions, which often lead to temporal conflicts, brittle reasoning, and limited traceability. We propose MemWeaver, a unified memory framework that consolidates long-term agent experiences into three interconnected components: a temporally grounded graph memory for structured relational reasoning, an experience memory that abstracts recurring interaction patterns from repeated observations, and a passage memory that preserves original textual evidence. MemWeaver employs a dual-channel retrieval strategy that jointly retrieves structured knowledge and supporting evidence to construct compact yet information-dense contexts for reasoning. Experiments on the LoCoMo benchmark demonstrate that MemWeaver substantially improves multi-hop and temporal reasoning accuracy while reducing input context length by over 95% compared to long-context baselines. Our data and code are available at here.

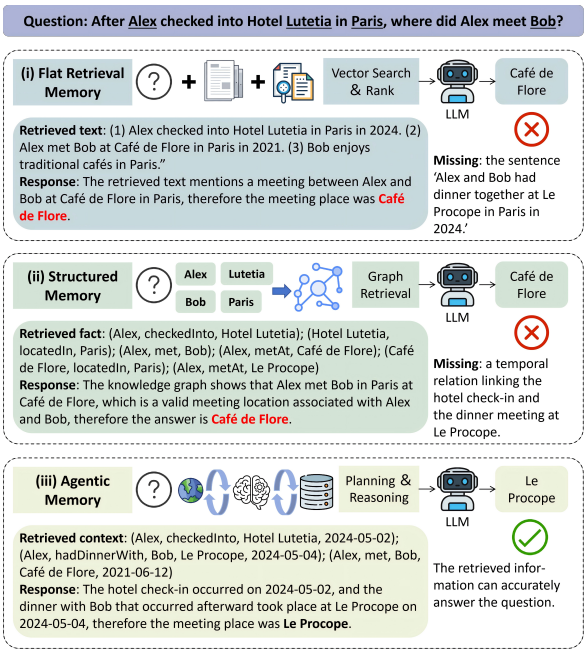


Figure 1: Comparison of flat retrieval memory, structured memory, and agentic memory from a reasoning perspective. While flat and structured memories can store and retrieve factual information, they do not explicitly model temporal relations between events, which limits their ability to answer temporally constrained queries. Agentic memory addresses this limitation by maintaining temporally grounded events and reasoning over their order.

1 Introduction

Large language model-based agents are increasingly deployed in long-horizon interactive settings, such as conversational assistants and personalized systems that span multiple sessions (Park et al., 2023; Maharana et al., 2024; Jiao et al., 2026a). These scenarios require agents to maintain temporal consistency, accumulate user-specific knowledge, and reason over distant past interactions. However, limited context windows make reliance on internal representations alone infeasible, rendering external memory mechanisms essential (Lewis et al., 2020; Lee et al., 2024; Huang et al., 2025a).

Recent work on agent memory has largely focused on *retrieval-based augmentation*. Most approaches store past interactions as text passages or vector embeddings and retrieve relevant items via semantic similarity to enrich the model's input context (Zhong et al., 2024b; Borgeaud et al., 2022; Huang et al., 2025c). These methods have demonstrated effectiveness in improving recall and long-context question answering (Zhong et al., 2024b; Maharana et al., 2024). Beyond unstructured retrieval, several studies explore *structured memory representations*. Knowledge-graph-based memo-

Findings of the Association for Computational Linguistics: ACL 2026, pages 12928–12956
July 2-7, 2026

ries explicitly model entities and relations to support compositional reasoning (Edge et al., 2024; Gutierrez et al., 2024), while experience-based or summary-based memories aim to abstract preferences, habits, or strategies from repeated interactions (Park et al., 2023; Shinn et al., 2023; Huang et al., 2025b). Collectively, these efforts highlight the importance of **memory representation** for long-horizon agent behavior.

Despite progress, as illustrated in Figure 1, existing memory systems exhibit key limitations in long-term settings: flat retrieval memories lack relational and temporal structure, making them brittle for multi-hop and time-constrained queries (Yang et al., 2018); structured memories suffer from noisy extractions and accumulated conflicts due to missing session-level verification (Zhong et al., 2024a); and high-level abstractions are often weakly grounded, hindering traceability and correction (Hong et al., 2024; Yao et al., 2023). Most approaches treat memory as a passive retrieval buffer, overlooking systematic consolidation and update as interactions accumulate.

These limitations suggest that effective long-term agent memory requires a shift in perspective. Rather than viewing memory as a static repository, memory should be modeled as a *consolidation process* that transforms episodic interaction traces into structured, reusable, and verifiable knowledge representations (Xu et al., 2025). This leads to a central research question:

> *How can an agent systematically consolidate long-term interaction histories into memory representations that support temporal consistency, compositional reasoning, cross-session generalization, and evidence-grounded decision making?*

Addressing this question requires jointly designing **memory structures** and the **mechanisms** by which they are written, reviewed, and retrieved.

To this end, we propose **MemWeaver**, a consolidation-centric **tri-layer memory framework** for long-horizon agents. It decomposes long-term memory into three complementary layers, each targeting a distinct reasoning capability. At the **structured fact level**, MemWeaver builds a *temporally grounded knowledge graph* with normalized absolute-time metadata to support *time-aware multi-hop reasoning* and resolve temporal conflicts, together with session-level reconciliation to improve consistency. At the **experience abstraction level**, MemWeaver clusters episodic interaction windows, validates cluster coherence, and abstracts reusable experience items only when supported by multiple coherent interactions, while linking them to source passages and related entities. At the **grounding level**, MemWeaver maintains passage memory that preserves raw textual evidence and links it to entities and experiences for **traceability and verification**.

During inference, MemWeaver adopts a **dual-channel retrieval strategy**. Structured retrieval over the knowledge graph provides precise and compositional context, while evidence retrieval assembles tightly linked passages and experience items, complemented by a small number of globally retrieved passages for recall. The fused context enables the language model to generate responses that are both *structurally informed* and *evidence grounded*.

Our contributions are summarized as follows:

- We formalize long-term agent memory from a reasoning-oriented perspective, highlighting the need for temporal grounding, compositional structure, and evidence traceability to support long-horizon inference.
- We propose MemWeaver, a consolidation-centric tri-layer memory framework that integrates temporally grounded knowledge graphs, experience abstraction, and evidence-linked passage memory through explicit writing and update mechanisms.
- We empirically demonstrate that MemWeaver substantially improves long-horizon reasoning performance, particularly on temporal and multi-hop tasks, while maintaining strong evidence traceability.

2 Related Work

2.1 Memory Systems for LLM Agents

LLM agents operating in long-horizon, multi-step settings must retain information beyond fixed context windows, motivating external memory mechanisms (Liu et al., 2023; Packer et al., 2023). Existing approaches span retrieval-oriented memories that store interaction traces in external stores (e.g., vector databases) for cross-session access (Zhong et al., 2024b; Park et al., 2023; Jiao et al., 2026b), structured or controlled architectures that regulate long-term information via hierarchies or controllers (e.g., MemGPT, SCM) (Packer et al., 2023; Wang et al., 2023), and abstraction-based memories that

distill reusable reasoning patterns or task-level insights (Ouyang et al., 2025; Zhang et al., 2025). More recent agentic frameworks further enable continual memory evolution by adding, revising, or deleting entries across sessions (Xu et al., 2025; Chhikara et al., 2025). Despite this progress, many systems still rely on coarse, LLM-centric updates and provide limited support for temporally constrained reasoning and evidence-grounded correction at scale (Zhong et al., 2024b).

2.2 Knowledge Graph–based Knowledge Organization

Knowledge graphs (KGs) explicitly model entities and relations, and are increasingly used to structure knowledge for LLMs. Prior work leverages LLMs to construct or enrich KGs from unstructured text, or to instantiate task-specific graphs at inference time (Wu et al., 2025). KGs have also been used to organize retrieved evidence and support multi-hop reasoning, including graph-guided inference frameworks such as Think-on-Graph and GraphRAG (Sun et al., 2024; Edge et al., 2024), as well as hybrid retrieval methods that combine vector similarity with graph-based exploration (Gutierrez et al., 2024; Gutiérrez et al., 2025; Yasunaga et al., 2021). However, many KG-assisted pipelines introduce additional overhead and rely on task-specific graph construction and retrieval heuristics, and they typically lack explicit temporal grounding and continual updates, limiting their use in long-term interactive settings.

Despite the effectiveness of vector-similarity and LLM-assisted retrieval, response quality still depends heavily on how memories are constructed. We therefore propose MemWeaver, which unifies graph-structured knowledge, experience summaries, and passage-level evidence with an effective retrieval mechanism for long-horizon agentic reasoning.

3 Problem Formulation

We consider a long-running conversational agent whose interaction history arrives incrementally as a sequence of dialogue units (i.e., QA turns) with metadata. The i-th dialogue unit is defined as

$$x_i = \langle q_i, a_i, s_i, t_i \rangle, \tag{1}$$

where q_i and a_i denote the question and answer texts, s_i denotes the speaker, and t_i denotes the timestamp.

We define the textual content of a dialogue unit as:

$$\text{text}(x_i) = q_i \oplus a_i, \tag{2}$$

where $\oplus$ denotes concatenation, and compute its semantic embedding by:

$$e_i = \phi(\text{text}(x_i)). \tag{3}$$

The resulting embedding is used for clustering, routing, and retrieval.

To support scalable long-term reasoning with traceable evidence, we model the agent memory state as a tri-layer set:

$$M = \{G, E, P\}, \tag{4}$$

where G denotes Graph Memory, implemented as a structured knowledge graph with temporal or conditional attributes; E denotes Experience Memory, consisting of induced reusable experience items distilled from topical dialogue clusters; and P denotes Passage Memory, a dense retrieval channel over original text spans for evidence recall. These three components emphasize, respectively, *structured factual knowledge*, *abstract reuse capability*, and *traceability to original text*.

The system supports two core operations. For **memory writing**, given a newly observed dialogue unit x_i, the memory state is updated as

$$M_{i+1} = \text{Update}(M_i, x_i). \tag{5}$$

For **memory-based reasoning**, given a user query Q and the current memory state M_i, the system retrieves a reasoning context $(C_{\text{KG}}, C_{\text{TXT}})$ and generates the final answer as

$$y = \text{LLM}(Q, C_{\text{KG}}, C_{\text{TXT}} \mid P_{\text{ans}}), \tag{6}$$

where P_{ans} is a fixed prompt template for answer generation.

Our goal is to maintain a high-quality memory state over long dialogues such that generated answers are accurate and supported by retrievable and traceable evidence from memory.

4 Methodology

This section presents **MemWeaver**, a tri-layer long-term memory system that integrates Graph Memory (GM), Experience Memory (ExpM), and Passage Memory (PM). Each component plays a complementary role during reasoning: GM supports compositional relational facts, ExpM captures reusable abstractions distilled from repeated interactions, and PM preserves verbatim evidence for robust recall.

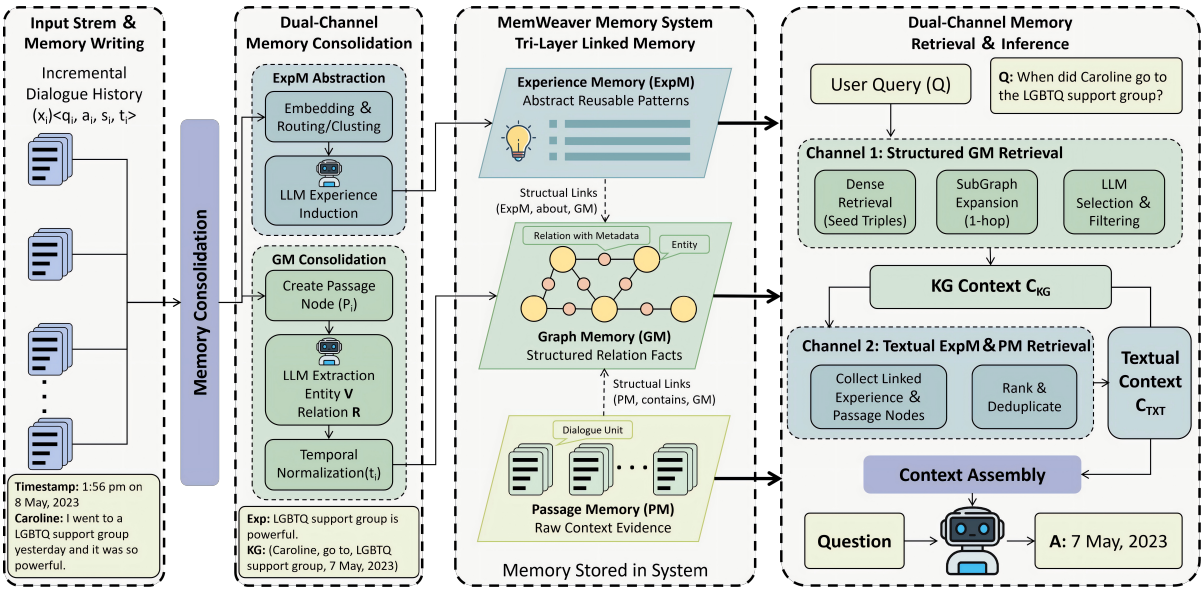


Figure 2: Overview of the MemWeaver framework. MemWeaver integrates Graph Memory (GM), Experience Memory (ExpM), and Passage Memory (PM) into a unified long-term memory system. Dialogue interactions are incrementally written into structured and unstructured memory, while dual-channel retrieval combines relational facts and textual evidence to construct a compact reasoning context for answer generation.

4.1 Tri-layer Memory Consolidation Framework

MemWeaver explicitly links its three memory components rather than treating them as isolated stores. Raw passages and experience items are attached to entity nodes in Graph Memory, allowing retrieved facts to be traced back to supporting evidence and enabling joint use of structured reasoning and evidence grounding.

The three memory carriers are loosely coupled through shared semantic representations and explicit structural links, allowing different memory types to be accessed and combined in a coordinated manner during inference. This coupling also constrains how memory evolves over time: graph updates are consolidated before entering persistent storage, experience abstraction is induced from supported interaction clusters, and retrieval is organized through linked structured and textual channels rather than independent memory buffers.

At a high level, Graph Memory provides the structural backbone for organizing entities and their relations, while Passage Memory and Experience Memory are anchored to this structure as evidence-bearing nodes, together forming a unified memory representation. We describe these components in Sections 4.2, 4.3, and 4.4.

4.2 Graph Memory: Relational Fact Consolidation

Graph Memory is designed as a relational fact consolidation layer that organizes scattered factual statements across long interaction histories into a coherent and reusable structure. Instead of treating dialogue turns as independent evidence units, this layer explicitly consolidates repeated and temporally grounded facts into stable entity–relation representations. Such consolidation enables consistent access to factual knowledge over long time spans and provides a structural foundation for compositional and temporal reasoning, which is difficult to achieve through unstructured memory alone. In MemWeaver, graph writing is treated as a controlled consolidation process, where newly observed facts are normalized, reviewed, and reconciled before being committed to the persistent graph state.

MemWeaver represents its Graph Memory as a directed knowledge graph $G = (V, \mathcal{R})$. During memory construction, the graph consists of *entity nodes* and *semantic relation edges*, representing consolidated factual knowledge expressed in the dialogue history. At inference time, *passage nodes* and *experience nodes* are temporarily attached to entity nodes via *structural edges* (e.g., `contains` and `about`) to provide supporting evidence.

Each semantic relation r is represented as a triple

with metadata $m(r)$:

$$r = \langle h, \rho, u \rangle, \quad m(r) = \{\hat{t}, c, \pi\}, \tag{7}$$

where $\hat{t}$ is a normalized absolute time expression, c is an optional condition, π is a provenance identifier for traceability and m is metadata.

All LLM-based operations are performed using fixed prompt templates, which are denoted as P_{ent}, P_{rel}, and P_{review} for entity extraction, relation extraction, and session-level review, respectively. Under this setting, given a newly observed dialogue unit x_i, the system incrementally writes it into the graph. MemWeaver first creates a passage node that preserves the original text together with its timestamp and speaker information. It then performs a two-stage LLM-based extraction process, where entities are identified in the first stage and relation candidates are extracted under entity constraints in the second stage. To improve temporal consistency, each candidate relation is further processed by a dedicated normalization step that extracts an absolute time expression from the dialogue context:

$$\hat{t} = \eta(x_i, r), \tag{8}$$

Where η denotes the time-normalization function. After the initial write, MemWeaver performs a session-level review step. This normalization-and-review stage reduces unresolved relative-time references and local extraction conflicts that would otherwise accumulate across sessions. All entities and relations introduced within the current session are serialized into a compact textual representation and verified by an LLM, which outputs `add`, `update`, or `deny` operations to complete missing relations, correct erroneous ones, or remove noise. Finally, redundant semantic relations are removed, and the triple index $\mathcal{I}_T$ (a dense vector index over semantic relation triples) is rebuilt. This index serves as the entry point for structured retrieval during inference. The overall procedure for KG writing and maintenance is summarized in Algorithm 1.

4.3 Experience Memory: Experience Abstraction

Experience Memory is designed to abstract reusable patterns from repeated interactions that are not well captured as isolated factual relations. While Graph Memory consolidates explicit facts, many long-term signals such as preferences, tendencies, and recurring intents emerge only when multiple interactions are considered jointly. By abstracting such recurring signals into experience-level representations, this layer supports cross-session generalization and complements fact-centric reasoning with higher-level behavioral knowledge.

Experience Memory abstracts historical dialogues by clustering semantically related units and inducing a compact set of reusable experience texts. Given the embedding of each dialogue unit, MemWeaver first applies DBSCAN clustering (Schubert et al., 2017; Kulkarni and Burhanpurwala, 2024) with cosine distance to obtain candidate topical groups. Since density-based clustering may still produce mixed-topic clusters, each candidate cluster is further validated by an LLM-based coherence check. Inconsistent clusters are returned to a pending queue for re-processing, whereas consistent ones are finalized as clusters and assigned a short theme summary `center_text`, which serves as a semantic anchor for routing.

For each coherent cluster, MemWeaver induces a small set of experience items using an LLM. Each experience item is stored as a lightweight record consisting of an identifier, a semantic type, a content field, a list of supporting dialogue unit identifiers, and optional metadata. Each experience item must be explicitly supported by multiple dialogue units in the cluster and is stored together with its provenance information. To reduce noise, MemWeaver applies filtering and deduplication heuristics to exclude small-talk patterns and remove near-duplicate items, ensuring that the resulting Experience Memory captures reusable signals rather than conversational artifacts.

In the online incremental setting, newly arriving dialogue units are routed to existing clusters according to their similarity to cluster centers. Let $e_i = \phi(\text{text}(x_i))$ denote the embedding of a newly arriving dialogue unit, and let μ_j denote the center vector of cluster j:

$$\mu_j = \frac{1}{|C_j|} \sum_{x \in C_j} \phi(\text{text}(x)), j^\star = \arg\max_j \cos(e_i, \mu_j), \tag{9}$$

MemWeaver performs three-way routing. If $\max_j \cos(e_i, \mu_j) \geq \tau_{\text{high}}$, the dialogue unit is directly merged into the best-matching cluster. If $\tau_{\text{low}} \leq \max_j \cos(e_i, \mu_j) < \tau_{\text{high}}$, the dialogue unit is submitted to an LLM-based router for disambiguation among a shortlist of candidate clusters. Formally, let $\mathcal{J}(x_i)$ denote the set of candidate

clusters. The router predicts the assignment as:

$$j^\star = \mathrm{LLM}\big(x_i,\ \{(\texttt{center\_text}_j, \mathcal{S}_j)\}_{j\in\mathcal{J}(x_i)} \mid P_{\text{route}}\big)\,, \tag{10}$$

where $\mathcal{S}_j$ denotes a small set of representative dialogue units sampled from cluster j. If the router returns `none`, the dialogue unit is appended to a pending buffer; Otherwise, it is assigned to cluster $j^\star$. If $\max_j \cos(e_i, \mu_j) < \tau_{\text{low}}$, the dialogue unit is directly appended to the pending buffer for future re-clustering once the buffer reaches a fixed window size.

To control update cost, each cluster maintains an `add_buffer` and only triggers an update when the buffer size exceeds a threshold, at which point the cluster center is recomputed and experience items are re-induced. This strategy amortizes the expense of LLM-based extraction while maintaining freshness under long-running interactions. Algorithm 2 summarizes the complete induction and online update procedure, including three-way routing, buffered updates, and re-cluster.

4.4 Dual-Channel Memory Retrieval

MemWeaver constructs a dual-channel inference context by jointly retrieving structured relational facts and supporting textual evidence. Structured retrieval over Graph Memory provides precise, compositional access to relations, while textual retrieval over Passage and Experience Memory improves recall and grounding. The two channels are linked through the memory structure: structured retrieval first identifies relevant entities and relations, and textual retrieval then collects supporting evidence attached to those entities, complemented by a small global recall channel. Given a query Q, the retriever outputs a KG context C_{KG} and a textual context C_{TXT}, which are fed to the backbone LM. We denote by k_r, k_p, and k_e the retrieval budgets for triples, passages, and experience items, respectively.

Structured KG Retrieval. We embed semantic relation edges in Graph Memory (excluding structural links such as `contains` and `about`) and retrieve seed triples by cosine similarity. MemWeaver then expands a bounded-hop neighborhood, filters candidates by similarity to control context size, and applies an LLM selector to choose the most useful triples. To avoid overly narrow contexts, we augment the selected set with a few high-similarity triples for backfilling:

$$\mathcal{R}^\star = \mathcal{R}_{\text{LLM}} \cup \mathrm{Top}_{k_r}(\mathcal{R}_{\text{cand}}; Q)\,, \tag{11}$$

where P_{select} is a fixed prompt template for triple selection.

Textual Evidence Retrieval. We collect evidence from two sources. (i) For entities involved in $\mathcal{R}^\star$, we retrieve attached passage nodes via `contains` edges and experience nodes via `about` edges, so that the textual evidence is grounded in the structured graph context. (ii) We additionally query a global dense retriever over all dialogue units to improve recall. All retrieved texts are ranked by similarity to Q and deduplicated by dialogue identifiers and content.

Context Assembly. We serialize the selected subgraph $\mathcal{R}^\star$ to construct the graph context C_{KG}, and aggregate the retrieved passages and experience items to form the textual context C_{TXT}. The backbone LLM then produces the final prediction conditioned on the assembled memory contexts:

$$\hat{y} = f_{\text{LLM}}(Q,\, C_{\text{KG}},\, C_{\text{TXT}})\,. \tag{12}$$

5 Experiments

5.1 Experiment Preparation

Datasets. Following previous work (Maharana et al., 2024; Zhong et al., 2024b; Lee et al., 2024; Xu et al., 2025), we evaluate MemWeaver on the LoCoMo dataset (Maharana et al., 2024), a benchmark designed for long-term conversational question answering with extended multi-session dialogue histories. Compared with prior conversational datasets that contain around 1K tokens over a small number of sessions, LoCoMo features substantially longer conversations, averaging approximately 9K tokens and spanning up to 35 sessions. This setting makes LoCoMo particularly suitable for evaluating models' ability to retrieve, integrate, and reason over long-range contextual information. LoCoMo includes five question types (Single-Hop, Multi-Hop, Temporal, Open-Domain, and Adversarial), and we focus on the four answerable categories in our main experiments. Detailed dataset statistics and category definitions are provided in Appendix B.

Baselines. For fair comparison, we evaluate MemWeaver against four representative baselines: LoCoMo (Maharana et al., 2024), which relies on long-context prompting over the dialogue history; MemoryBank (Zhong et al., 2024b), a retrieval-based conversational memory framework; ReadAgent (Lee et al., 2024), a reading-based agent that selectively summarizes and retrieves dialogue his-

Model	Method	Category												Overall			
		Multi-Hop			Temporal			Open-Domain			Single-Hop			Ranking			Tokens
		F1	BLEU	RGE-2	F1	BLEU	RGE-2	F1	BLEU	RGE-2	F1	BLEU	RGE-2	F1	BLEU	RGE-2	
GPT -4o-mini	LoCoMo	24.35	16.91	8.50	22.54	16.93	9.03	15.39	13.59	3.86	**42.39**	31.65	**28.29**	2.00	2.25	2.25	21,625
	MemoryBank	6.10	4.52	1.57	5.04	3.02	0.56	6.32	3.27	1.04	8.42	3.96	3.21	4.75	5.00	4.50	1,513
	ReadAgent	9.15	6.48	2.47	12.60	8.87	0.95	5.31	5.12	0.55	9.67	7.66	2.99	4.25	4.00	4.50	643
	A-Mem	23.68	16.74	8.42	38.77	33.71	15.58	12.50	11.29	3.96	35.13	28.96	21.53	2.75	2.75	2.50	2,731
	MemWeaver	**26.00**	**17.08**	**10.90**	**50.83**	**43.72**	**23.36**	**20.73**	**15.63**	**5.23**	39.20	**33.58**	25.96	**1.25**	**1.00**	**1.25**	672
Llama3.2 -3B	LoCoMo	8.09	6.88	2.29	5.00	5.45	0.77	7.08	5.83	0.65	11.92	8.63	5.02	2.75	2.75	4.00	22,312
	MemoryBank	5.04	3.34	1.15	2.06	1.19	0.13	4.52	2.09	0.83	6.78	2.90	2.34	4.50	4.50	4.75	1,553
	ReadAgent	2.47	1.78	2.47	3.01	3.01	3.01	5.57	5.22	**5.07**	3.25	2.51	3.25	4.25	4.50	2.75	461
	A-Mem	9.96	**13.21**	6.62	8.62	6.76	4.41	5.03	5.78	0.87	19.85	19.70	16.98	2.50	2.00	2.25	2508
	MemWeaver	**11.86**	13.07	**6.94**	**11.37**	**13.16**	**8.67**	**9.64**	**7.77**	2.11	**21.22**	**22.27**	**20.28**	**1.00**	**1.25**	**1.25**	1,000
Llama3.2 -1B	LoCoMo	10.37	8.10	2.85	15.51	12.83	1.01	11.94	10.64	2.20	13.90	10.64	4.88	3.25	2.75	3.25	22,312
	MemoryBank	4.80	3.36	1.12	1.89	1.61	0.06	5.72	3.58	0.83	6.42	3.02	2.15	5.00	5.00	4.25	1,553
	ReadAgent	5.96	5.12	0.53	1.93	2.30	0.00	**12.46**	**11.17**	**5.47**	7.75	6.03	1.19	3.25	3.25	4.00	665
	A-Mem	11.03	9.21	3.35	17.76	12.98	3.64	12.34	7.58	2.85	19.20	15.01	7.63	2.25	2.50	2.25	2,647
	MemWeaver	**12.96**	**9.77**	**3.72**	**24.11**	**17.42**	**7.53**	12.41	9.94	3.04	**20.17**	**15.05**	**11.78**	**1.25**	**1.50**	**1.25**	1,000
Qwen2.5 -1.5B	LoCoMo	10.23	7.96	2.14	7.65	6.76	0.21	11.58	9.74	2.37	12.93	9.18	4.14	2.75	3.00	3.00	22,424
	MemoryBank	6.63	5.02	1.77	3.43	2.85	0.20	5.80	3.40	0.82	9.41	4.85	3.38	4.25	4.50	4.00	1,555
	ReadAgent	6.61	4.93	0	2.55	2.51	0	5.31	12.24	0	10.13	7.54	0	4.75	4.00	5.00	752
	A-Mem	12.57	9.94	5.17	14.54	12.45	1.73	10.83	9.21	2.43	20.06	14.86	11.07	2.25	2.50	2.00	2,574
	MemWeaver	**21.91**	**16.47**	**7.25**	**46.07**	**38.47**	**17.74**	**17.94**	**15.86**	**4.23**	**32.80**	**27.42**	**18.52**	**1.00**	**1.00**	**1.00**	734

Table 1: Experimental results on the LoCoMo dataset across four question types (Multi-Hop, Temporal, Open-Domain, and Single-Hop). Results are reported in F1, BLEU-1 (%), and RGE-2 (%). RGE-2 denotes ROUGE-2. Best results within each backbone are in **bold**, and MemWeaver is highlighted in gray. Ranking indicates the average rank across categories (Rank 1 is best; lower is better), computed separately for F1, BLEU-1, and RGE-2. Tokens is the average number of input tokens per query.

tory; and A-Mem (Xu et al., 2025), an agentic memory system based on atomic memory construction. A detailed description of the baselines is provided in Appendix D.

Evaluation Metrics. Following prior work on LoCoMo (Maharana et al., 2024), we employ two primary evaluation metrics. We report token-level F1 to assess answer accuracy by balancing precision and recall, and BLEU-1 to measure lexical overlap between generated responses and ground-truth answers. In addition, we report the average number of input tokens per query, which reflects the inference-time context length and associated computational cost. We further report additional metrics, including exact match (EM), METEOR, ROUGE-L and SBERT similarity, in the Appendix D.

5.2 Implementation Details

We evaluate four backbone LLMs: GPT-4o-mini (Hurst et al., 2024), Llama3.2-3B/1B (Team, 2024), and Qwen2.5-1.5B (Yang et al., 2025). All methods share identical system prompts and output formats. To decouple memory construction from inference-time reasoning, we use DeepSeek-V3.2 (non-thinking) (DeepSeek-AI et al., 2025) to build all MemWeaver memory components *offline* (never at inference), since structured memory writing (entity/relation extraction, temporal normalization, and session-level verification) is unreliable for small LMs. At inference, MemWeaver retrieves C_{KG} and C_{TXT} via top-k retrieval with default $k_r = k_p = k_e = 6$ (category-specific adjustments when needed). We use `all-minilm-l6-v2` (Reimers and Gurevych, 2019) for embeddings, and follow each baseline's original settings while matching inference-time context length to MemWeaver. More details are provided in Appendix B.

5.3 Main Results

Performance Analysis. Table 1 reports category-wise results and overall ranking on LoCoMo under four backbone language models. Ranking is the average rank across the four categories (lower is better), computed separately for F1, BLEU-1, and ROUGE-2. Across backbones, MemWeaver achieves the best overall ranking in most settings, indicating consistently strong performance across diverse question types.

MemWeaver is particularly effective on *Multi-Hop* and *Temporal* questions that require cross-session reasoning. With GPT-4o-mini, MemWeaver improves Multi-Hop F1 from 24.35 (LoCoMo) / 23.68 (A-Mem) to 26.00, and boosts Temporal F1 from 38.77 (A-Mem) to 50.83. Similar trends are observed for smaller backbones: for Qwen2.5-1.5B, MemWeaver improves Temporal F1 from 14.54 to 46.07. We attribute the larger gains on smaller backbones to structured retrieval, which externalizes key factual and temporal cues

and reduces reliance on parametric knowledge and long-context reasoning.

Compared with LoCoMo, MemWeaver achieves higher or comparable accuracy while using substantially shorter inference contexts. MemoryBank performs poorly across most categories, suggesting that flat retrieval over unstructured memories is insufficient for complex long-horizon reasoning. Relative to the agentic baseline A-Mem, MemWeaver yields consistent improvements (notably on *Temporal* and *Open-Domain*), highlighting the benefit of separating temporally grounded Graph Memory from experience-level abstractions.

Token Efficiency. MemWeaver also substantially reduces inference-time context length. LoCoMo typically requires over 22K input tokens per query due to long-context prompting, whereas MemWeaver stays within 1K tokens across all backbones, reducing input length by over 95% while maintaining or improving performance. Compared with A-Mem, MemWeaver further reduces context length by roughly 2–4×. Memory and latency are reported in Table 2, showing that MemWeaver trades modest retrieval overhead for stronger accuracy and substantially shorter inputs, making it suitable for long-running and resource-constrained deployments.

5.4 Ablation Study

We conduct an ablation study to analyze the contribution of the two core components in MemWeaver: Graph Memory (GM) and Experience Memory (ExpM). We evaluate two variants, w/o GM and w/o ExpM, under two backbone models, focusing on answerable question types (Multi-Hop, Temporal, Open-Domain, and Single-Hop). In all settings, the global passage retrieval channel is preserved to ensure that observed performance differences mainly reflect the impact of structured and abstracted memory components rather than access to raw textual evidence.

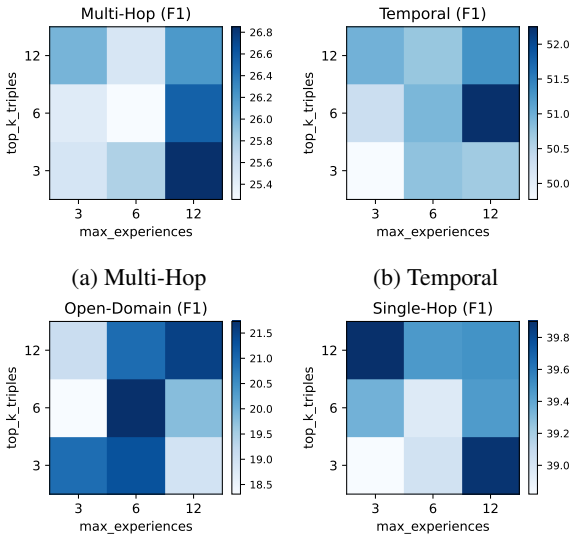
As shown in Table 3, removing ExpM leads to consistent but moderate performance drops across tasks, indicating its role in capturing reusable abstractions that support cross-session reasoning. In contrast, removing GM causes severe degradation, especially on Multi-Hop and Temporal questions, highlighting the importance of graph-based structural and temporal organization for compositional reasoning. Overall, the full MemWeaver model achieves the best and most balanced performance, demonstrating that Graph Memory and Experience Memory play complementary roles: GM provides structured factual reasoning, while ExpM supplies higher-level abstraction, together enabling robust long-term conversational reasoning.

We further conduct an atomic strand-level ablation on Qwen2.5-1.5B, comparing the full MemWeaver model against single-strand variants (KG, EXP, and PASS) on the two most challenging categories, *Multi-Hop* and *Temporal*. As shown in Table 4, KG is the strongest individual strand, especially on *Temporal* questions, indicating that structured and temporally grounded organization is the primary driver of long-horizon reasoning. EXP and PASS provide complementary abstraction and evidence grounding, respectively. The full MemWeaver model still performs best overall, suggesting that its gains come from the complementary roles of the three memory strands rather than from system complexity alone.

Table 2: Comparison of memory usage and retrieval time across different memory methods.

Method	Memory Usage (MB)			Retrieval Times (ms)
	GM	ExpM	Total	
MemoryBank	-	-	7.23	17.07 ± 3.61
A-Mem	-	-	18.29	16.00 ± 7.18
MemWeaver	5.24	8.07	13.31	41.57 ± 12.85

5.5 Hyperparameter Analysis



(a) Multi-Hop (b) Temporal

(c) Open-Domain (d) Single-Hop

Figure 3: Hyperparameter sensitivity analysis of MemWeaver. Each heatmap reports performance under different retrieval configurations for (a) Multi-Hop, (b) Temporal, (c) Open-Domain, and (d) Single-Hop questions.

Model	Method	Multi-Hop			Temporal			Open-Domain			Single-Hop		
		F1	BLEU-1	RGE-2	F1	BLEU-1	RGE-2	F1	BLEU-1	RGE-2	F1	BLEU-1	RGE-2
GPT-4o-mini	w/o ExpM	24.86	15.70	9.83	48.67	41.60	21.74	19.65	15.31	5.05	38.07	32.66	23.84
	w/o GM	13.97	8.28	4.75	9.58	5.60	1.04	10.30	9.05	3.01	13.12	11.06	6.45
	MemWeaver	**26.00**	**17.08**	**10.90**	**50.83**	**43.72**	**23.36**	**20.73**	**15.63**	**5.23**	**39.20**	**33.58**	**25.96**
Qwen2.5-1.5B	w/o ExpM	19.59	14.13	6.62	45.30	37.75	15.39	17.33	15.07	3.97	32.38	26.74	16.85
	w/o GM	13.48	9.85	4.31	5.78	4.88	0.83	10.33	9.76	1.61	13.84	11.46	6.35
	MemWeaver	**21.91**	**16.47**	**7.25**	**46.07**	**38.47**	**17.74**	**17.94**	**15.86**	**4.23**	**32.80**	**27.42**	**18.52**

Table 3: Ablation study of MemWeaver under two backbone models (GPT-4o-mini and Qwen2.5-1.5B). Results are reported in F1, BLEU-1, and RGE-2 (ROUGE-2) (%).

Method	Multi-Hop F1	Temporal F1
KG	20.53	45.14
EXP	15.09	4.76
PASS	11.90	20.96
MemWeaver	**21.91**	**46.07**

Table 4: Atomic strand-level ablation on Qwen2.5-1.5B. We report F1 on the two most challenging categories, Multi-Hop and Temporal.

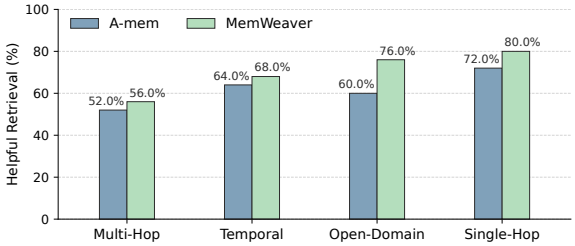


Figure 4: Human evaluation of evidence supports quality for A-Mem and MemWeaver across four question categories. Using GPT-4o-mini as the backbone, human annotators assess, on 25 sampled questions per category, whether the retrieved knowledge provides sufficient support for the generated answers.

We investigate MemWeaver's sensitivity to retrieval budgets, varying the number of retrieved graph triples and textual memory items. Figure 3 reports heatmaps across the four answerable categories under different retrieval configurations. Overall, MemWeaver is stable across a wide range of settings, with only mild performance variation and no abrupt degradation as retrieval sizes change. Increasing the number of retrieved items does not consistently improve performance and often shows diminishing or negligible gains. These results suggest that MemWeaver does not rely on large retrieval volumes; instead, its structured memory design enables efficient and selective retrieval, prioritizing relevance over quantity.

Taken together, the proposed memory system remains both stable and efficient, and its effectiveness stems from precise organization and targeted retrieval rather than brute-force context expansion.This robustness is particularly desirable in long-horizon interactive settings, where retrieval budgets may vary across users, sessions, or deployment constraints. It also suggests that MemWeaver does not depend on careful hyperparameter tuning to remain effective, which improves its practicality for real-world agent systems.

5.6 Human Evaluation

To complement automatic metrics, we conduct a human evaluation to assess whether the retrieved knowledge provides sufficient support for the generated answers. As shown in Figure 4, MemWeaver consistently outperforms A-Mem across all question categories, indicating more reliable and evidence-grounded retrieval. Details are provided in Appendix D.9.

6 Conclusion

In this paper, we propose MemWeaver, a structured long-term memory system for LLM agents that unifies graph-structured knowledge, abstracted experiences, and passage-level evidence to support scalable and traceable conversational reasoning. MemWeaver incrementally maintains a temporally grounded knowledge graph with LLM-based verification, derives reusable experience items from clustered interactions, and retains a global passage retrieval channel for accessing original evidence. Experiments on the LoCoMo benchmark across multiple backbone models show its effectiveness and consistent improvements in long-horizon conversational question answering with efficient inference.

Limitations

Although MemWeaver achieves encouraging results, we recognize several directions for future exploration. First, while the system is able to organize memory in a structured manner, the quality of memory organization may still be influenced by the inherent capabilities of the underlying language

model. Different models may produce slightly different abstractions or associations when constructing memory. Second, the current implementation primarily focuses on textual interactions. Extending the framework to incorporate multimodal information, such as images or audio, is a promising direction for future work.

Acknowledgments

This work was supported by the Young Scientists Fund of the National Natural Science Foundation of China under Grant No. 62407012.

References

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, and 9 others. 2022. Improving language models by retrieving from trillions of tokens. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 2206–2240. PMLR.

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

Aixin Liu DeepSeek-AI, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, and 1 others. 2025. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From local to global: A graph RAG approach to query-focused summarization. *CoRR*, abs/2404.16130.

Bernal Jimenez Gutierrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. Hipporag: Neurobiologically inspired long-term memory for large language models. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From RAG to memory: Non-parametric continual learning for large language models. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*. OpenReview.net.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2024. Metagpt: Meta programming for A multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Chengkai Huang, Junda Wu, Yu Xia, Zixu Yu, Ruhan Wang, Tong Yu, Ruiyi Zhang, Ryan A Rossi, Branislav Kveton, Dongruo Zhou, and 1 others. 2025a. Towards agentic recommender systems in the era of multimodal large language models. *arXiv preprint arXiv:2503.16734*.

Chengkai Huang, Junda Wu, Zhouhang Xie, Yu Xia, Rui Wang, Tong Yu, Subrata Mitra, Julian McAuley, and Lina Yao. 2025b. Pluralistic off-policy evaluation and alignment. *arXiv preprint arXiv:2509.19333*.

Chengkai Huang, Yu Xia, Rui Wang, Kaige Xie, Tong Yu, Julian McAuley, and Lina Yao. 2025c. Embedding-informed adaptive retrieval-augmented generation of large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 1403–1412.

Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Madry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, Alex Paino, and 79 others. 2024. Gpt-4o system card. *CoRR*, abs/2410.21276.

Sagar Imambi, Kolla Bhanu Prakash, and GR Kanagachidambaresan. 2021. Pytorch. In *Programming with TensorFlow: solution for edge computing applications*, pages 87–104. Springer.

Shuguang Jiao, Chengkai Huang, Shuhan Qi, Xuan Wang, Yifan Li, and Lina Yao. 2026a. Doctor-rag: Failure-aware repair for agentic retrieval-augmented generation. *arXiv preprint arXiv:2604.00865*.

Shuguang Jiao, Xinyu Xiao, Yunfan Wei, Shuhan Qi, Chengkai Huang, Quan Z. Sheng, and Lina Yao. 2026b. Prunerag: Confidence-guided query decomposition trees for efficient retrieval-augmented generation. In *Proceedings of the ACM Web Conference 2026, WWW 2026, Dubai, United Arab Emirates, originally scheduled for April 13-17, 2026, rescheduled for June 29 - July 3, 2026*, pages 1923–1934. ACM.

Omkaresh Kulkarni and Adnan Burhanpurwala. 2024. A survey of advancements in dbscan clustering algorithms for big data. In *2024 3rd International conference on Power Electronics and IoT Applications in Renewable Energy and its Control (PARC)*, pages 106–111. IEEE.

Kuang-Huei Lee, Xinyun Chen, Hiroki Furuta, John F. Canny, and Ian Fischer. 2024. A human-inspired

reading agent with gist memory of very long contexts. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

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

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. 2024. Evaluating very long-term conversational memory of LLM agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 13851–13870. Association for Computational Linguistics.

Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long T. Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. 2025. Reasoningbank: Scaling agent self-evolving with reasoning memory. *CoRR*, abs/2509.25140.

Charles Packer, Vivian Fang, Shishir G. Patil, Kevin Lin, Sarah Wooders, and Joseph E. Gonzalez. 2023. Memgpt: Towards llms as operating systems. *CoRR*, abs/2310.08560.

Joon Sung Park, Joseph C. O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology, UIST 2023, San Francisco, CA, USA, 29 October 2023- 1 November 2023*, pages 2:1–2:22. ACM.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Erich Schubert, Jörg Sander, Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. 2017. DBSCAN revisited, revisited: Why and how you should (still) use DBSCAN. *ACM Trans. Database Syst.*, 42(3):19:1–19:21.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel M. Ni, Heung-Yeung Shum, and Jian Guo. 2024. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Llama Team. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Bing Wang, Xinnian Liang, Jian Yang, Hui Huang, Shuangzhi Wu, Peihao Wu, Lu Lu, Zejun Ma, and Zhoujun Li. 2023. Scm: Enhancing large language model with self-controlled memory framework. *arXiv e-prints*, pages arXiv–2304.

Shanglin Wu, Lihui Liu, Jinho D. Choi, and Kai Shu. 2025. Improving factuality in llms via inference-time knowledge graph construction. *CoRR*, abs/2509.03540.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. 2025. A-MEM: agentic memory for LLM agents. *CoRR*, abs/2502.12110.

An Yang, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoyan Huang, Jiandong Jiang, Jianhong Tu, Jianwei Zhang, Jingren Zhou, Junyang Lin, Kai Dang, Kexin Yang, Le Yu, Mei Li, Minmin Sun, Qin Zhu, Rui Men, Tao He, and 9 others. 2025. Qwen2.5-1m technical report. *CoRR*, abs/2501.15383.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2369–2380. Association for Computational Linguistics.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021,*

Online, June 6-11, 2021, pages 535–546. Association for Computational Linguistics.

Guibin Zhang, Muxin Fu, Guancheng Wan, Miao Yu, Kun Wang, and Shuicheng Yan. 2025. G-memory: Tracing hierarchical memory for multi-agent systems. *CoRR*, abs/2506.07398.

Lingfeng Zhong, Jia Wu, Qian Li, Hao Peng, and Xindong Wu. 2024a. A comprehensive survey on automatic knowledge graph construction. *ACM Comput. Surv.*, 56(4):94:1–94:62.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2024b. Memorybank: Enhancing large language models with long-term memory. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pages 19724–19731. AAAI Press.

Appendix Outline

A Environment Details

All experiments were conducted on a Linux server running `Linux 6.8.0-90-generic` with `glibc 2.39`. The software stack was based on `Python 3.10.19` and `PyTorch 2.9.0` (Imambi et al., 2021) compiled with `CUDA 12.8` (`torch 2.9.0+cu128`), with `cuDNN 9.10.2` enabled. All experiments were executed with GPU acceleration.

The hardware configuration consisted of four `NVIDIA A100-SXM4 GPUs` with `80GB` HBM2e memory each. The installed NVIDIA driver version was `575.57.08`, and the CUDA runtime reported by the driver was `CUDA 12.9`. Unless otherwise specified, all memory construction and inference pipelines were executed on this hardware configuration.

B Implementation Details

Dataset and Evaluation Scope. We evaluate MemWeaver on the LoCoMo dataset released by Snap Research,[1] which is designed for long-term conversational question answering over multi-session dialogue histories. LoCoMo categorizes questions into five types: (1) *Single-Hop* questions answerable from a single session; (2) *Multi-Hop* questions requiring information synthesis across sessions; (3) *Temporal* questions that test time-aware reasoning; (4) *Open-Domain* questions that require integrating conversational context with general knowledge; and (5) *Adversarial* questions that are unanswerable from the dialogue history. In total, the dataset contains 7,512 question–answer pairs across these categories. Since our primary goal is to evaluate long-term memory and evidence-supported reasoning over extended dialogue trajectories, we mainly focus on the four answerable categories (Single-Hop, Multi-Hop, Temporal, and Open-Domain) in our main experiments.

The dataset does not provide an official train/dev/test split, and following prior work, we evaluate directly on the full set of annotated question–answer pairs without further partitioning. In our main experiments, we focus on the four answerable question categories: *Single-Hop*, *Multi-Hop*, *Temporal*, and *Open-Domain*. Adversarial questions are excluded in the main experiment part, as they primarily evaluate abstention behavior rather than evidence-supported reasoning. However, we still report the experiments on *Adversarial* dataset in Appendix D.6. We do not fix a random seed, since the system does not involve stochastic training and all memory construction is performed deterministically given the underlying LLM outputs.

Backbone Models and Inference. We evaluate four backbone language models: GPT-4o-mini (Hurst et al., 2024), Llama3.2-3B (Team, 2024), Llama3.2-1B (Team, 2024), and Qwen2.5-1.5B (Yang et al., 2025). GPT-4o-mini is accessed via a commercial API, while the remaining models are served locally using `Ollama`. All methods, including baselines and MemWeaver, share identical system prompts and output formats to ensure fair comparison. Category-specific answer prompts are used at inference time to accommodate differences in question styles (e.g., temporal or multi-hop queries), while keeping the overall prompting strategy consistent across methods. Decoding configurations follow standard practice for each backend and are kept consistent within each model.

Offline Memory Construction. All memory components in MemWeaver are constructed offline prior to inference. We employ DeepSeek-V3.2 (API-based) (DeepSeek-AI et al., 2025) exclusively for memory construction, including entity extraction, relation extraction, experience induction, and session-level verification. This model is *not* used during inference. Once the full memory is built, it remains fixed throughout evaluation, and all backbone models query the same memory state.

Textual embeddings for dialogue units, passages, experience items, and knowledge graph triples are computed using the `all-minilm-l6-v2` (Reimers and Gurevych, 2019) sentence encoder. The same embedding model is used consistently across clustering, routing, and retrieval.

Graph Memory Construction. Graph Memory is implemented as a directed knowledge graph that consolidates relational facts across dialogue sessions. Entity extraction, relation extraction, and session-level review are performed using fixed prompt templates. Implicit temporal expressions in dialogue are normalized into absolute time representations during memory writing. Structured retrieval operates over semantic relation triples only, excluding structural edges. During inference, graph retrieval expands candidate triples using a single-hop neighborhood expansion from seed relations.

Experience Memory Induction. Experience Memory is constructed by clustering dialogue units using DBSCAN with cosine distance. We

[1] https://github.com/snap-research/locomo

set the clustering parameters to $\texttt{eps} = 0.3$ and $\texttt{min\_samples} = 2$. Each candidate cluster is screened for semantic coherence before inducing reusable experience items. For online updates, newly arriving dialogue units are routed to existing clusters based on cosine similarity, with thresholds $\texttt{sim\_high} = 0.8$ and $\texttt{sim\_low} = 0.5$. Each cluster maintains an update buffer, and experience reinduction is triggered when the buffer size reaches 4, amortizing the cost of LLM-based updates in long-running settings.

Retrieval and Context Assembly. MemWeaver employs a dual-channel retrieval strategy. Structured retrieval over Graph Memory provides relational facts, while textual retrieval gathers supporting passages and experience items. Unless otherwise specified, the retrieval budgets are fixed to $k_r = k_p = k_e = 6$. For graph triples, passages, and experience items, respectively, across all question categories. The retrieved structured and textual contexts are assembled into a compact inference input, which is then provided to the backbone language model for answer generation.

C Prompt Templates

Entity Extraction Prompt

You are an entity extraction assistant.
Your task: Extract entities from a dialogue snippet.

Guidelines:

- Extract concise entity names that appear in the text (people, locations, organizations, events, objects, etc.).
- Do not invent entities.
- Prefer a canonical form of the entity name.
- If an entity is expressed as a combined phrase, keep the full phrase intact and do not split it into smaller parts.
- Return a de-duplicated list.

DO NOT EXTRACT:

1. Unclear entities such as `"he"`, `"that"`, `"there"`.
2. Time/Date expressions (relative or absolute) such as `"yesterday"`, `"last week"`, `"next month"`.

Dialogue:
`{dialogue_text}`

Output (STRICT JSON):

```
{
  "entities": ["entity1", "entity2", ...]
}
```

Figure 5: Prompt for entity extraction from a dialogue snippet.

Relation Extraction Prompt

You are a relation extraction assistant.
Your task: Extract meaningful relations between entities as triples from a dialogue snippet.

Guidelines:

- Extract relations **ONLY** between entities in the provided list.
- Each relation must be directly supported by the text.
- `relation_type` should be a short predicate phrase (*lowercase preferred*).
- If the relation happens under certain conditions, you can optionally include a `"condition"` field to describe it briefly.

DO NOT EXTRACT:

- time/location
- unmeaningful or vague relations, e.g., `"is related to"`, `"has something to do with"`, `"is associated with"`, etc.

Dialogue:
`{dialogue_text}`

Detected entities:
`{entity_list_text}`

Output (STRICT JSON):

```
{
  "relations": [
    {
      "source": "entity_name1",
      "target": "entity_name2",
      "relation_type": "short relation phrase",
      "condition": "if mentioned"
    }
  ]
}
```

Figure 6: Prompt for relation extraction between detected entities in a dialogue snippet.

Temporal Normalization Prompt

You are a time expression extractor.
Your task: Extract an **ABSOLUTE** time expression indicating when the event described by the given relation occurred or is scheduled to occur; the absolute time may be inferred from relative time expressions in the dialogue.

Guidelines:

- Identify the time **MOST relevant** to the given relation/event.
- The dialogue may contain:
  - Absolute time expressions (e.g., `"20 May 2022"`, `"May 2022"`, `"2022"`)
  - Relative time expressions (e.g., `"yesterday"`, `"last week"`, `"this weekend"`)
- If a relative time expression is present, you **MAY** use it as a clue and resolve it into an **ABSOLUTE** time.
- The final output **MUST** be an **ABSOLUTE, HUMAN-READABLE** time expression.

What counts as HUMAN-READABLE ABSOLUTE time:

- A specific calendar date (e.g., `"20 May, 2022"`)
- A specific month and year (e.g., `"May, 2022"`)
- A specific year (e.g., `"2022"`)

DO NOT OUTPUT:

- Relative time expressions (e.g., `"yesterday"`, `"last week"`, `"tomorrow"`)
- Vague time references without a calendar anchor (e.g., `"recently"`, `"soon"`, `"later"`)

If there is **NO** clear usable time information for the target relation, return an empty string `""`.

Output format MUST be one of:

- Day-level: `"20 May, 2022"`
- Month-level: `"May, 2022"`
- Year-level: `"2022"`

Given:

- Dialogue: `{dialogue_text}`
- Relation: `{relation_desc}`

Output (STRICT JSON):

```
{
  "absolute_time": ""
}
```

Figure 7: Prompt for temporal normalization of dialogue-based relations into absolute time.

Knowledge Graph Verification Prompt

You are a knowledge graph review assistant.
Your tasks:
Review a knowledge graph extracted from a dialogue session.

Options:

1. **ADD:** Find important relations that are clearly expressed in the dialogue but are **MISSING** from the current relation list.
2. **UPDATE:** For some **EXISTING** relations, refine `relation_type`, `time`/`condition` metadata.
3. **DENY:** If a relation in the current list is clearly **NOT supported** or **contradicted** by the dialogue, mark it as denied (to be removed).

You are given:
The full dialogue text for this session:
The dialogue happened at: `{dialogue_timestamp}`
`{full_dialogue_text}`

Existing entities in the KG for this session:
`{entities_text}`

Existing relations (triples) in the KG for this session:
`{relations_text}`

Output (STRICT JSON):

```
{
  "add": [
    {"source": "A", "relation_type": "predicate", "target": "B", "time": "if mentioned", "
        condition": "if mentioned"}
  ],
  "update": [
    {"relation_id": "rid", "relation_type": "new predicate", "time": "if mentioned", "condition
        ": "if mentioned"}
  ],
  "deny": [
    {"relation_id": "rid"}
  ]
}
```

Figure 8: Prompt for verifying and refining a dialogue-derived knowledge graph.

Experience Extraction Prompt

You extract reusable experiences from short dialogues.
You will see several Q&A items from the same semantic cluster. Each item has an index like `[0]`, `[1]`, etc.

Dialogue samples:
`{qa_context}`

Your task:

- Extract a **SMALL SET** of reusable experiences that can help in similar future cases.
- Each experience must be directly supported by the dialogue (do not invent facts).
- Do **NOT** output vague life advice or generic statements.
- Prefer `"fact"` or `"preference"` when the dialogue states something directly.
- Only use `"strategy"` when the experience clearly generalizes beyond this specific person.
- Avoid generic strategies like `"communicate more"`, `"be kind"`, or `"support is important"`.
- Only mark Q&A indices where the experience is explicitly expressed.
- It is better to output fewer, high-quality experiences than many weak or generic ones.
- `content` $\leq$ 120 characters.

Allowed types:

- `fact`: a stable fact likely to remain true
- `strategy`: a general reusable approach
- `preference`: a stable interest or habit

Few-shot example:

```
[0] Speaker=Alex
    Q: I started weekly therapy recently.
    A:
[1] Speaker=Ben
    Q: Has it helped?
    A:
[2] Speaker=Alex
    Q: Yes, talking regularly helps me feel less overwhelmed.
    A:
```

Example output:

```
{
  "experiences": [
    {
      "type": "fact",
      "content": "Alex attends weekly therapy and feels less overwhelmed.",
      "source_qa_indices": [0, 2]
    }
  ]
}
```

Now output **STRICT JSON** for the current dialogue only:
Output (STRICT JSON):

```
{
  "experiences": [
    {
      "type": "fact | strategy | preference",
      "content": "short experience (<=120 chars)",
      "source_qa_indices": [0, 1]
    }
  ]
}
```

Figure 9: Prompt for extracting reusable experiences from clustered Q&A dialogue samples.

D Experiment

D.1 Detailed Baselines Introduction

LoCoMo (Maharana et al., 2024) takes a direct approach by leveraging foundation models without memory mechanisms for question answering tasks. For each query, it incorporates the complete preceding conversation and questions into the prompt, evaluating the model's reasoning capabilities.

ReadAgent (Lee et al., 2024) tackles long-context document processing through a sophisticated three-step methodology: it begins with episode pagination to segment content into manageable chunks, followed by memory gisting to distill each page into concise memory representations, and concludes with interactive look-up to retrieve pertinent information as needed.

MemoryBank (Zhong et al., 2024b) introduces an innovative memory management system that maintains and efficiently retrieves historical interactions. The system features a dynamic memory updating mechanism based on the Ebbinghaus Forgetting Curve theory, which intelligently adjusts memory strength according to time and significance. Additionally, it incorporates a user portrait building system that progressively refines its understanding of user personality through continuous interaction analysis.

A-Mem (Xu et al., 2025) proposes an agentic memory framework that constructs and maintains atomic memory units for long-horizon interactions. It organizes memories into interconnected notes that can be incrementally updated across sessions, enabling the agent to retrieve and reuse relevant memory entries when answering queries.

D.2 Evaluation Metric

The F1 score represents the harmonic mean of precision and recall, offering a balanced metric that combines both measures into a single value. This metric is particularly valuable when balancing between complete and accurate responses:

$$\mathrm{F1} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}, \tag{13}$$

where

$$\mathrm{precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}, \tag{14}$$

and

$$\mathrm{recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}. \tag{15}$$

In question-answering systems, the F1 score plays a crucial role in evaluating exact matches between predicted and reference answers. This is especially important for span-based QA tasks, where systems must identify precise text segments while maintaining comprehensive coverage of the answer.

BLEU-1 evaluates the precision of unigram matches between system outputs and reference texts:

$$\text{BLEU-1} = \mathrm{BP} \cdot \exp\left(\sum_{n=1}^{1} w_n \log p_n\right), \tag{16}$$

where the brevity penalty BP is defined as

$$\mathrm{BP} = \begin{cases} 1, & c > r, \\ e^{1-r/c}, & c \le r, \end{cases} \tag{17}$$

and

$$p_n = \frac{\sum_i \sum_k \min(h_{ik}, m_{ik})}{\sum_i \sum_k h_{ik}}. \tag{18}$$

Here, c is the candidate length, r is the reference length, h_{ik} is the count of the n-gram i in candidate k, and m_{ik} is the maximum count of that n-gram in any reference. In QA tasks, BLEU-1 evaluates lexical precision and is particularly useful for generative QA systems where exact matching may be overly strict.

ROUGE-L measures the longest common subsequence (LCS) between the generated and reference texts:

$$\text{ROUGE-L} = \frac{(1+\beta^2) R_l P_l}{R_l + \beta^2 P_l}, \tag{19}$$

where

$$R_l = \frac{\mathrm{LCS}(X,Y)}{|X|}, \qquad P_l = \frac{\mathrm{LCS}(X,Y)}{|Y|}. \tag{20}$$

ROUGE-2 computes bigram overlap between the generated and reference texts:

$$\text{ROUGE-2} = \frac{\sum_{\text{bigram} \in \text{ref}} \min(\mathrm{Count}_{\text{ref}}, \mathrm{Count}_{\text{cand}})}{\sum_{\text{bigram} \in \text{ref}} \mathrm{Count}_{\text{ref}}}. \tag{21}$$

ROUGE-L focuses on sequence-level matching, while ROUGE-2 emphasizes local word order. Both metrics are useful for evaluating the fluency and coherence of generated answers.

METEOR computes a score based on aligned unigrams between candidate and reference texts, accounting for synonyms and paraphrases:

$$\text{METEOR} = F_{\text{mean}} \cdot (1 - \text{Penalty}), \tag{22}$$

Method	Multi-Hop			Temporal			Open-Domain			Single-Hop		
	F1	BLEU	RGE-2	F1	BLEU	RGE-2	F1	BLEU	RGE-2	F1	BLEU	RGE-2
MemoryBank	5.03	4.28	1.59	2.43	1.60	0.46	5.54	3.12	1.04	6.84	3.62	2.91
A-Mem	31.13	19.54	11.39	41.36	31.90	15.67	12.31	10.66	3.03	41.58	36.69	26.69
MemWeaver	**31.35**	**20.29**	**11.83**	**55.52**	**45.36**	**26.97**	**21.13**	**16.34**	**5.90**	**45.19**	**39.47**	**28.96**

Table 5: Experimental results on the LoCoMo dataset (DeepSeek-V3.2 backbone). Results are reported in F1 and BLEU-1 (%). Best results in each row are in bold, and MemWeaver is highlighted in gray.

Model	Method	Multi-Hop		Temporal		Open-Domain		Single-Hop	
		F1	BLEU	F1	BLEU	F1	BLEU	F1	BLEU
Qwen2.5-1.5B	MemGPT	10.44	7.61	4.21	3.89	13.42	11.64	9.56	7.34
	MemWeaver	**21.91**	**16.47**	**46.07**	**38.47**	**17.94**	**15.86**	**32.80**	**27.42**

Table 6: Experimental results on the LoCoMo dataset with the Qwen2.5-1.5B backbone, additionally comparing MemWeaver with MemGPT. Results are reported in F1 and BLEU-1 (%). Best results are shown in bold, and MemWeaver is highlighted in gray.

where

$$F_{\text{mean}} = \frac{10PR}{R + 9P}, \tag{23}$$

and

$$\text{Penalty} = 0.5 \cdot \left(\frac{ch}{m}\right)^3. \tag{24}$$

Here, P and R denote precision and recall, ch is the number of chunks, and m is the number of matched unigrams. METEOR captures semantic similarity beyond exact matching and is well-suited for evaluating paraphrased answers.

SBERT Similarity measures the semantic similarity between two texts using sentence embeddings:

$$\text{SBERT}(x, y) = \cos(\mathbf{e}_x, \mathbf{e}_y) = \frac{\mathbf{e}_x \cdot \mathbf{e}_y}{\|\mathbf{e}_x\|\|\mathbf{e}_y\|}, \tag{25}$$

where $\mathbf{e}_x$ and $\mathbf{e}_y$ are SBERT embeddings of the two texts. SBERT Similarity is particularly useful for QA evaluation when lexical overlap is low, but semantic meaning is preserved.

D.3 DeepSeek-V3.2 Backbone

Notably, as shown in Table 5, when all methods use DeepSeek-V3.2 (DeepSeek-AI et al., 2025) as the backbone, MemWeaver still achieves the best performance across all question categories, indicating that its gains are not attributable to backbone choice. Compared with A-Mem, MemWeaver yields substantial improvements on the more reasoning-intensive categories, most notably on *Temporal* questions (F1: 55.52 vs. 41.36; BLEU-1: 45.36 vs. 31.90; RGE-2: 26.97 vs. 15.67), while also improving *Multi-Hop* (F1: 31.35 vs. 31.13) and *Single-Hop* (F1: 45.19 vs. 41.58). In contrast, MemoryBank performs poorly across all categories (e.g., Temporal F1: 2.43), suggesting that flat retrieval over unstructured memories is insufficient even under a strong backbone, whereas MemWeaver benefits from temporally grounded and structured retrieval.

D.4 Additional Baseline Comparison

To broaden the comparison with recent memory-oriented methods, we additionally include MemGPT (Packer et al., 2023) as an extra baseline under the Qwen2.5-1.5B backbone. As shown in Table 6, MemWeaver consistently outperforms MemGPT across all four question categories on LoCoMo, with especially clear gains on *Multi-Hop* and *Temporal* questions. These results further verify the advantage of MemWeaver over additional memory-oriented baselines.

D.5 Comparison Results

Across backbones, MemWeaver consistently improves both exact-match and semantics-aware metrics. As shown in Table 7, under GPT-4o-mini, MemWeaver achieves the best EM across all four categories (e.g., Temporal: 11.21; Single-Hop: 16.77) and ranks first overall for both EM and METEOR (1.00/1.50). Similar trends hold for smaller backbones such as Qwen2.5-1.5B, where MemWeaver substantially increases EM on Temporal (8.10) and Single-Hop (12.96), again achieving the top overall ranking.

Beyond exact matching, Table 8 shows that MemWeaver also yields strong gains on sequence-

level and semantic similarity metrics. In particular, it consistently achieves the highest ROUGE-L and SBERT scores across most categories and backbones (e.g., GPT-4o-mini Temporal: ROUGE-L 49.98, SBERT 76.03; Qwen2.5-1.5B Temporal: ROUGE-L 44.67, SBERT 72.69), indicating that its answers are not only more precise but also more semantically aligned with the references. Together, these results suggest that MemWeaver improves both factual exactness and semantic faithfulness across diverse backbone capacities.

D.6 Results on the Adversarial Dataset

On the LoCoMo adversarial task, MemWeaver consistently outperforms MemoryBank and A-Mem across all metrics with GPT-4o mini. In particular, MemWeaver achieves over 10% gains in EM and large improvements in F1, ROUGE-2, and ROUGE-L, indicating stronger robustness under adversarial perturbations. These results suggest that MemWeaver is more resilient to misleading or conflicting memory signals, preserving both factual correctness and semantic alignment. Overall, the consistent improvements across exact-match and generation-based metrics validate the effectiveness of MemWeaver's structured memory consolidation in adversarial settings.

D.7 Case Study and Failure Analysis

To qualitatively compare MemWeaver with A-Mem, we present four representative examples from different question types. These examples highlight three main strengths of MemWeaver.

(i) Multi-Hop compositional reasoning: MemWeaver leverages graph-based compositional retrieval to connect evidence scattered across multiple sessions, enabling implicit relational chains for multi-hop reasoning.

(ii) Temporal grounding: By explicitly storing normalized time information in memory, MemWeaver can return grounded timestamps rather than unresolved relative expressions, which is important for temporal reasoning.

(iii) High-precision memory retrieval: MemWeaver improves retrieval precision by jointly leveraging structured triples and evidence-linked passages, which helps preserve complete and accurate supporting context for both factual and open-domain questions.

Overall, these examples show that MemWeaver produces answers that are more specific, temporally grounded, and better aligned with the underlying evidence.

To further analyze failure modes, we additionally construct an ablation variant, MEMWEAVER0, in which three core mechanisms are removed: session-level review, temporal normalization, and the LLM-based coherence check. To focus on representative differences, we compute a weighted score $0.45 \cdot \text{F1} + 0.35 \cdot \text{ROUGE-L} + 0.2 \cdot \text{BLEU-1}$, and define a question as *discriminative* when the score gap between MemWeaver and MEMWEAVER0 is at least 0.4. Under this criterion, we identify 217 discriminative failure cases. Among them, *Temporal Grounding Failure* accounts for 45.16%, *Memory Strand Selection Error* for 38.71%, *Incomplete Constraint Integration* for 10.14%, and *Memory Conflict Resolution Failure* for 5.99%.

Temporal Grounding Failure. Without temporal normalization, relative time expressions can be incorrectly anchored to nearby conversation timestamps. For example, for the question *"When did Caroline and Melanie go to a pride festival together?"*, the retrieved knowledge includes "last year," while nearby passages are dated 2023. Without normalization, the system incorrectly outputs 2023 instead of the correct year 2022. Explicitly resolving relative expressions into absolute time nodes stabilizes this type of reasoning.

Memory Strand Selection Error. Without session-level review, predicate-level distinctions across turns can be missed, leading to the selection of the wrong memory strand. For example, for *"When did Melanie sign up for a pottery class?"*, the retrieved context contains both a participation event on Aug 25, 2023 and a sign-up event on Jul 2, 2023. Without sufficient strand verification, the model selects the attendance-related evidence and returns the incorrect date.

Incomplete Constraint Integration. Some failures arise when retrieved evidence is partially correct but the generated answer does not fully satisfy the query constraints. For instance, for *"Did Jon and Gina both participate in dance competitions?"*, the evidence supports participation for both individuals, but the answer only mentions Gina. This type of error becomes more likely when session-level review and coherence checking are removed.

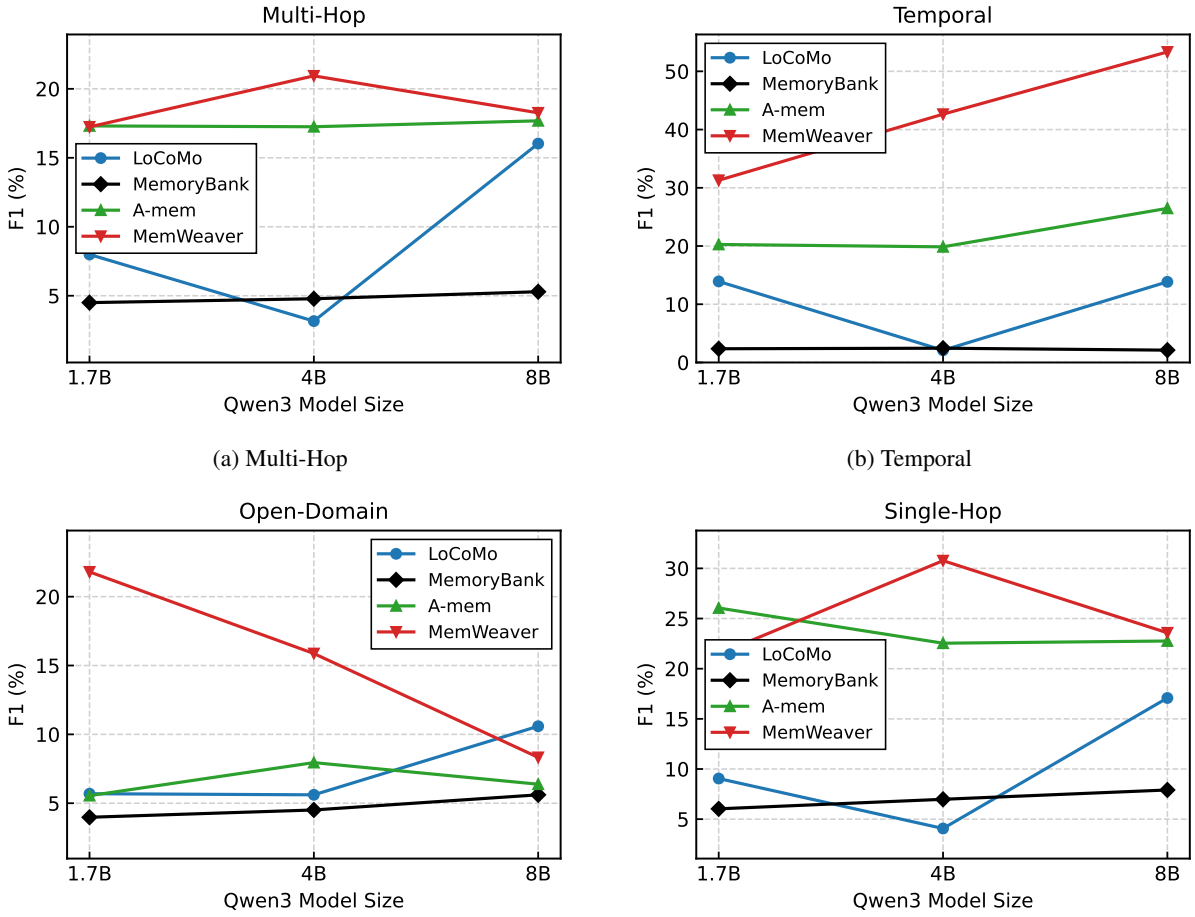
Memory Conflict Resolution Failure. When multiple similar events coexist in memory, removing temporal anchoring and strand-level verification can lead to incorrect event selection. For example, for *"When did Nate take his turtles to the*

Model	Method	Category								Overall	
		Multi-Hop		Temporal		Open-Domain		Single-Hop		Ranking	
		EM	METEOR	EM	METEOR	EM	METEOR	EM	METEOR	EM	METEOR
GPT -4o-mini	LoCoMo	0.35	**15.56**	0.00	9.87	2.08	7.67	6.54	**39.75**	3.50	2.00
	MemoryBank	0.00	7.57	0.00	3.95	2.08	7.26	0.00	13.04	4.25	4.25
	ReadAgent	0.35	5.46	0.00	4.76	0.00	3.69	0.00	8.01	4.25	4.75
	A-Mem	1.77	13.74	3.74	19.35	4.17	8.65	9.27	32.79	2.00	2.50
	MemWeaver	**4.26**	14.20	**11.21**	**25.25**	**6.25**	**9.70**	**16.77**	33.46	**1.00**	**1.50**
Llama3.2 -3B	LoCoMo	0.71	4.71	0.00	3.17	1.04	4.57	0.71	9.47	3.25	3.25
	MemoryBank	0.00	6.51	0.00	2.30	0.00	**6.39**	0.00	11.15	4.63	2.75
	ReadAgent	0.00	1.21	0.62	2.33	1.04	3.39	0.00	2.46	3.63	4.75
	A-Mem	0.71	5.61	0.62	4.42	1.04	3.51	3.92	**16.68**	2.50	2.50
	MemWeaver	**1.06**	**6.60**	**2.49**	**5.48**	**2.08**	3.98	**5.35**	16.26	**1.00**	**1.75**
Llama3.2 -1B	LoCoMo	**0.71**	5.92	0.00	5.81	3.12	7.25	1.07	10.77	1.75	3.00
	MemoryBank	0.00	6.06	0.00	1.76	1.04	6.66	0.00	9.27	3.00	3.75
	ReadAgent	0.00	2.97	0.00	1.31	1.04	7.13	1.07	5.36	2.75	4.75
	A-Mem	0.00	5.97	**0.62**	7.50	0.00	**7.26**	0.00	12.33	3.00	2.00
	MemWeaver	0.00	**6.11**	**0.62**	**8.23**	**4.17**	7.21	**1.78**	**13.14**	**1.25**	**1.50**
Qwen2.5 -1.5B	LoCoMo	0.35	6.01	0.00	3.74	1.04	9.44	1.07	13.48	3.25	3.25
	MemoryBank	0.00	7.58	0.00	3.49	0.00	7.06	0.00	14.08	4.63	3.75
	ReadAgent	0.00	3.67	0.00	1.88	1.04	8.97	0.71	5.52	3.88	4.50
	A-Mem	0.71	8.14	1.25	7.01	1.04	7.51	3.80	20.55	2.25	2.50
	MemWeaver	**2.84**	**11.34**	**8.10**	**20.84**	**7.29**	**10.99**	**12.96**	**25.66**	**1.00**	**1.00**

Table 7: Experimental results on the LoCoMo dataset across four question types (Multi-Hop, Temporal, Open-Domain, and Single-Hop). Results are reported in EM and METEOR (%). EM denotes Exact Match. Note that due to the strictness of the EM metric, some methods receive zero scores in certain categories. Best results within each backbone are in bold, and MemWeaver is highlighted in gray. Ranking indicates the average rank across categories (Rank 1 is best; lower is better), computed separately for EM and METEOR.

Model	Method	Category								Overall	
		Multi-Hop		Temporal		Open-Domain		Single-Hop		Ranking	
		RGE-L	SBERT	RGE-L	SBERT	RGE-L	SBERT	RGE-L	SBERT	RGE-L	SBERT
GPT -4o-mini	LoCoMo	24.88	45.78	23.09	40.56	16.65	40.09	39.77	51.85	2.25	2.50
	MemoryBank	5.08	32.25	4.21	26.10	5.06	32.79	6.92	32.04	5.00	4.25
	ReadAgent	9.45	28.67	13.12	45.07	5.76	26.72	9.92	26.78	4.00	4.50
	A-Mem	22.41	45.44	38.38	67.86	13.93	36.89	35.75	50.73	2.75	2.75
	MemWeaver	**25.68**	**46.51**	**49.98**	**76.03**	**22.44**	**42.71**	**40.82**	**54.63**	**1.00**	**1.00**
Llama3.2 -3B	LoCoMo	9.01	27.33	7.45	18.84	7.35	28.76	12.31	26.33	3.00	4.00
	MemoryBank	3.85	31.53	1.52	19.65	3.08	**32.10**	5.29	31.68	4.50	2.50
	ReadAgent	1.78	17.40	3.01	12.02	5.22	19.63	2.51	14.63	4.50	5.00
	A-Mem	19.05	38.51	15.43	30.57	7.73	30.47	28.90	41.72	2.00	**1.75**
	MemWeaver	**20.22**	**38.82**	**19.41**	**34.53**	**11.37**	29.35	**34.26**	**44.61**	**1.00**	**1.75**
Llama3.2 -1B	LoCoMo	11.06	30.65	15.54	46.56	13.23	37.84	14.63	31.55	3.25	3.25
	MemoryBank	3.68	29.62	1.58	17.25	4.51	31.30	5.20	26.39	5.00	4.75
	ReadAgent	6.49	29.26	4.62	26.45	14.29	39.19	8.03	26.44	3.50	4.00
	A-Mem	11.53	**35.17**	17.35	50.99	13.74	**42.88**	20.29	34.46	2.25	**1.50**
	MemWeaver	**13.99**	34.08	**26.71**	**58.69**	**15.17**	40.60	**23.92**	**36.83**	**1.00**	**1.50**
Qwen2.5 -1.5B	LoCoMo	10.42	29.68	7.45	25.59	12.12	35.66	12.70	30.14	3.00	3.75
	MemoryBank	5.70	31.33	3.02	18.83	4.75	31.22	8.11	31.84	4.50	4.00
	ReadAgent	7.14	28.20	2.81	27.27	12.63	35.13	7.88	26.33	4.00	4.25
	A-Mem	14.69	36.91	24.32	60.23	10.93	37.22	21.63	38.44	2.50	2.00
	MemWeaver	**21.29**	**47.66**	**44.67**	**72.69**	**18.81**	**43.41**	**34.04**	**48.67**	**1.00**	**1.00**

Table 8: Experimental results on the LoCoMo dataset across four question types (Multi-Hop, Temporal, Open-Domain, and Single-Hop). Results are reported in RGE-L and SBERT (%). RGE-L denotes ROUGE-L. Best results within each backbone are in bold, and MemWeaver is highlighted in gray. Ranking indicates the average rank across categories (Rank 1 is best; lower is better), computed separately for RGE-L and SBERT.



(a) Multi-Hop

(b) Temporal

(c) Open-Domain

(d) Single-Hop

Figure 10: Model scaling results on Qwen3-1.7B, 4B, and 8B backbones. Each panel reports the F1 performance across backbone scales for one LoCoMo question category, comparing LoCoMo, MemoryBank, A-Mem, and MemWeaver. Lines indicate how each method's category-level F1 changes as the backbone model size increases.

Metric	MemoryBank	A-Mem	MemWeaver
EM	26.68	22.42	**37.00**
F1	29.13	25.14	**72.20**
ROUGE-2	24.78	20.88	**70.83**
ROUGE-L	29.03	25.39	**72.22**
BLEU-1	28.46	24.36	**64.61**

Table 9: Results on the LoCoMo **Adversarial** category using GPT-4o mini. All scores are reported as percentages (%). Best results are in bold, and MemWeaver is highlighted in gray.

beach?", the retrieved context contains both a general outing event on Oct 25 and a beach event on Nov 10. Without explicit disambiguation, the system outputs the former instead of the correct latter date.

These analyses suggest that the three mechanisms play complementary roles: temporal normalization supports chronological grounding, session-level review improves strand consistency, and coherence checking promotes semantic completeness. Together, they reduce wrong-strand retrieval and unresolved memory conflicts, leading to more robust long-horizon reasoning.

D.8 Model Scaling Analysis

Figure 10 shows that MemWeaver generally outperforms the baselines across Qwen backbones of different scales on all question categories. Overall, MemWeaver demonstrates stable and competitive performance as the backbone model size varies.

To further examine whether this advantage persists under a larger-capacity backbone, we additionally evaluate MemWeaver on Qwen2.5-32B-Instruct and compare it with A-Mem. As shown in Table 11, MemWeaver continues to outperform A-Mem across all question categories, with particularly clear gains on reasoning-intensive settings such as *Multi-Hop* and *Temporal*. These results

Category	Question	A-Mem	MemWeaver (Ours)	Reference
Multi-Hop	What Jon thinks the ideal dance studio should look like?	A place where people can express themselves through dance.	**By the water.**	By the water, with natural light and Marley flooring.
Temporal	When is Jon's group performing at a festival?	Next month.	**February, 2023.**	February, 2023.
Open-Domain	What would Caroline's political leaning likely be?	LGBTQ activist group, passionate about rights and community support.	**Likely liberal.**	Liberal.
Single-Hop	What kind of flooring is Jon looking for in his dance studio?	The context does not specify what kind of flooring Jon is looking for in his dance studio.	**Marley flooring.**	Marley flooring.

Table 10: Case study examples comparing A-Mem and MemWeaver across four question types.

Model	Method	Multi-Hop		Temporal		Open-Domain		Single-Hop	
		F1	BLEU	F1	BLEU	F1	BLEU	F1	BLEU
Qwen2.5-32B	A-Mem	17.04	13.84	25.39	22.89	8.96	7.34	28.52	22.39
	MemWeaver	**23.32**	**18.44**	**48.68**	**41.48**	**18.04**	**16.46**	**37.35**	**33.16**

Table 11: Experimental results on the LoCoMo dataset with the Qwen2.5-32B backbone. Results are reported in F1 and BLEU-1 (%). Best results are shown in bold, and MemWeaver is highlighted in gray.

further confirm that the advantage of MemWeaver is not limited to smaller models, but remains consistent as backbone scale increases.

D.9 Human Evaluation Details

We conduct a human evaluation to assess the quality of retrieved knowledge, focusing on whether the retrieved evidence is sufficient to answer the question correctly. Following prior work, we randomly sample 25 questions per category (Multi-Hop, Temporal, Open-Domain, and Single-Hop) from the LoCoMo dataset.

For each question, annotators are provided with:

- The question,
- The knowledge retrieved by the memory system, and
- The ground-truth reference answer.

We recruit five NLP experts who are Master's or PhD students actively working in the NLP field, with research experience in retrieval-augmented generation and LLM agents. Each question is independently annotated by all five experts. Annotators are asked to make a binary judgment (*helpful* / *not helpful*) indicating whether the retrieved knowledge contains sufficient information to derive the reference answer. The annotators come from diverse geographic regions, with three based in Asia and two based in Oceania. We report the average helpfulness rate across the five annotators. Annotators are compensated at a rate of $0.10 per annotation, for a total of $50 in human evaluation costs.

E Algorithm

E.1 Memory Construction

Algorithm 1 KG Writing and Maintenance

Require: New dialogue unit $x_i = \langle q_i, a_i, s_i, t_i \rangle$, KG G
Ensure: Updated KG G and triple index $\mathcal{I}_T$

Create passage node p_i with text and metadata (s_i, t_i)
$\mathcal{V} \leftarrow \mathrm{LLM}(x_i \mid P_{\text{ent}})$ ▷ entities
$\mathcal{R} \leftarrow \mathrm{LLM}(x_i, \mathcal{V} \mid P_{\text{rel}})$ ▷ candidate triples
for all $r = \langle h, \rho, u \rangle \in \mathcal{R}$ **do**
    $\hat{t} \leftarrow \eta(x_i, r)$; attach metadata $m(r)$
    Insert/merge entity nodes; add semantic edge r into G
end for
Link p_i to involved entities via structural edges
$\Omega \leftarrow \mathrm{LLM}(G, \text{session} \mid P_{\text{review}})$
Apply Ω; remove redundant relations; rebuild $\mathcal{I}_T$

Algorithm 2 Experience Induction and Online Update

Require: New dialogue unit x_i; clusters $\{C_j\}$ with centers $\{\mu_j\}$; pending buffer $\mathcal{B}$; thresholds $(\tau_{\text{high}}, \tau_{\text{low}})$; candidate shortlist size K; update trigger B_{add}; recluster window B_{re}

Ensure: Updated clusters $\{C_j\}$, centers $\{\mu_j\}$, and experience items $\mathcal{E}$

$e_i \leftarrow \phi(\text{text}(x_i))$
$j^\star \leftarrow \arg\max_j \cos(e_i, \mu_j)$; $s^\star \leftarrow \max_j \cos(e_i, \mu_j)$
if $s^\star \geq \tau_{\text{high}}$ **then**
    $C_{j^\star} \leftarrow C_{j^\star} \cup \{x_i\}$
    $\texttt{add\_buffer}(C_{j^\star}) \leftarrow \texttt{add\_buffer}(C_{j^\star}) \cup \{x_i\}$
else if $\tau_{\text{low}} \leq s^\star < \tau_{\text{high}}$ **then**
    $\mathcal{J}(x_i) \leftarrow \text{Top}_K(\{\mu_j\}, e_i)$
    $\hat{j} \leftarrow \text{LLM}\big(x_i,\ \{(\texttt{center\_text}_j, \mathcal{S}_j)\}_{j \in \mathcal{J}(x_i)} \mid P_{\text{route}}\big)$
    if $\hat{j} \neq$ none **then**
        $C_{\hat{j}} \leftarrow C_{\hat{j}} \cup \{x_i\}$
        $\texttt{add\_buffer}(C_{\hat{j}}) \leftarrow \texttt{add\_buffer}(C_{\hat{j}}) \cup \{x_i\}$
    else
        $\mathcal{B} \leftarrow \mathcal{B} \cup \{x_i\}$
    end if
else
    $\mathcal{B} \leftarrow \mathcal{B} \cup \{x_i\}$
end if
for all clusters C_j with $|\texttt{add\_buffer}(C_j)| \geq B_{\text{add}}$ **do**
    $\mu_j \leftarrow \frac{1}{|C_j|} \sum_{x \in C_j} \phi(\text{text}(x))$
    $\texttt{center\_text}_j \leftarrow \text{LLM}(C_j \mid P_{\text{sum}})$
    $\mathcal{E}_j \leftarrow \text{LLM}(C_j \mid P_{\text{ind}})$
    $\mathcal{E}_j \leftarrow \text{Filter}(\mathcal{E}_j)$
    clear $\texttt{add\_buffer}(C_j)$
end for
if $|\mathcal{B}| \geq B_{\text{re}}$ **then**
    $\{C'_k\} \leftarrow \text{DBSCAN}(\mathcal{B})$; clear $\mathcal{B}$
    for all candidate clusters C'_k **do**
        $b_k \leftarrow \text{LLM}(C'_k \mid P_{\text{coh}})$
        if $b_k =$ yes **then**
            $C_{\text{new}} \leftarrow C'_k$
            $\mu_{\text{new}} \leftarrow \frac{1}{|C_{\text{new}}|} \sum_{x \in C_{\text{new}}} \phi(\text{text}(x))$
            $\texttt{center\_text}_{\text{new}} \leftarrow \text{LLM}(C_{\text{new}} \mid P_{\text{sum}})$
            $\mathcal{E}_{\text{new}} \leftarrow \text{LLM}(C_{\text{new}} \mid P_{\text{ind}})$
            $\mathcal{E}_{\text{new}} \leftarrow \text{Filter}(\mathcal{E}_{\text{new}})$
        else
            $\mathcal{B} \leftarrow \mathcal{B} \cup C'_k$
        end if
    end for
end if

E.2 Retrieval and Reasoning

Algorithm 3 Inference Retrieval and Context Assembly

Require: Query Q; triple index $\mathcal{I}_T$; graph G (with attached passage/experience nodes); Passage Memory P; budgets (k_r, k_p, k_e)

Ensure: $(C_{\mathrm{KG}}, C_{\mathrm{TXT}})$

$\mathcal{R}_{\mathrm{seed}} \leftarrow \mathrm{Retrieve}(\mathcal{I}_T, Q, k_r)$

$\mathcal{R}_{\mathrm{cand}} \leftarrow \mathrm{Expand}(G, \mathcal{R}_{\mathrm{seed}}, 1)$

$\mathcal{R}_{\mathrm{cand}} \leftarrow \mathrm{Filter}(\mathcal{R}_{\mathrm{cand}}, Q)$

$\mathcal{R}_{\mathrm{llm}} \leftarrow \mathrm{LLM}(Q, \mathcal{R}_{\mathrm{cand}} \mid P_{\mathrm{select}})$

$\mathcal{R}^\star \leftarrow \mathcal{R}_{\mathrm{llm}} \cup \mathrm{Top}_{k_r}(\mathcal{R}_{\mathrm{cand}}; Q)$

$\mathcal{R}^\star \leftarrow \mathrm{Deduplicate}(\mathcal{R}^\star)$

$\mathcal{P}_{\mathrm{kg}} \leftarrow \mathrm{Collect}(G, \mathcal{R}^\star, \text{passage})$

$\mathcal{E}_{\mathrm{kg}} \leftarrow \mathrm{Collect}(G, \mathcal{R}^\star, \text{experience})$

$\mathcal{P}_{\mathrm{glob}} \leftarrow \mathrm{Retrieve}(P, Q, k_p)$

$\mathcal{P}^\star \leftarrow \mathrm{RankDedup}(\mathcal{P}_{\mathrm{kg}} \cup \mathcal{P}_{\mathrm{glob}}, Q)$

$\mathcal{E}^\star \leftarrow \mathrm{RankDedup}(\mathcal{E}_{\mathrm{kg}}, Q)$

$C_{\mathrm{KG}} \leftarrow \mathrm{Serialize}(\mathcal{R}^\star)$

$C_{\mathrm{TXT}} \leftarrow \mathrm{Assemble}(\mathcal{P}^\star_{1:k_p}, \mathcal{E}^\star_{1:k_e})$ **return** $(C_{\mathrm{KG}}, C_{\mathrm{TXT}})$