MemCoRe: Recovering Evidence from Progressively Compressed Factual Knowledge for Agent Memory

Zhenyuan Zhang[1]* **Xianzhang Jia**[1]* **Zhiqin Yang**[1]
Zhenbo Song[2] **Wei Xue**[1]† **Sirui Han**[1]† **Yike Guo**[1]†

[1] The Hong Kong University of Science and Technology
[2] Nanjing University of Science and Technology

zyzhang.bbetter@gmail.com, xjiaaf@connect.ust.hk
yangzqccc@gmail.com, songzb@njust.edu.cn
{weixue,siruihan,yikeguo}@ust.hk

Abstract

Memory systems enable LLM agents to consolidate and retrieve relevant evidence from the factual knowledge accumulated through growing interaction histories for downstream reasoning. Existing approaches have explored diverse strategies for organizing and compressing these histories. However, balancing compression with retrieval effectiveness remains challenging: retaining too much content can cause relevant evidence to be obscured by redundant entries, while discarding too aggressively may remove content that later proves relevant. This amounts to a tradeoff between compressing redundancy and preserving enough structure to retrieve target evidence, for which the information bottleneck provides an analytical framework. To this end, we propose **MEMCORE**, which organizes memory as a compression hierarchy where each level compresses redundancy further while retaining the structure needed for retrieval at that level. In this hierarchy, evidence is progressively compressed from detailed records through extracted keywords to topic groups. This enables retrieval to locate target evidence by searching across levels of the hierarchy. On LoCoMo, MEMCORE achieves the highest reported question-weighted F1 across four backbone models, with the clearest statistical evidence on Qwen3-8B. Code is available at `https://github.com/Liar-zzy/MemCoRe`.

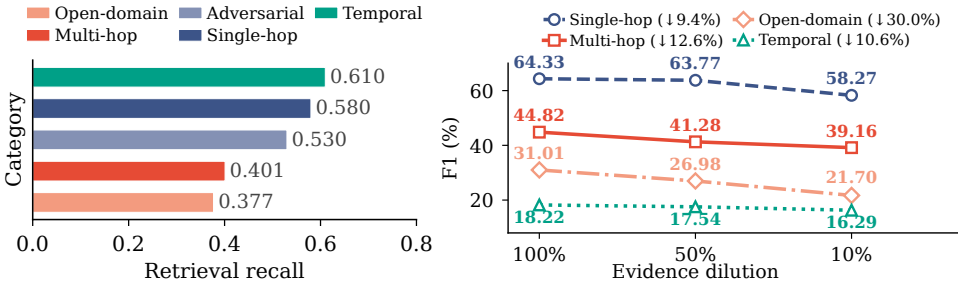
1 Introduction

Memory systems enable LLM agents to consolidate and retrieve relevant evidence from the factual knowledge accumulated through growing interaction histories for downstream reasoning [33, 29, 6]. As interaction histories grow, target evidence is progressively outnumbered by accumulated observations, making accurate retrieval increasingly difficult. To manage this growth, agents construct a structured memory representation and retrieve from it rather than searching the full interaction history. The structure and content of this representation directly govern which evidence can be located at retrieval time, making memory construction inseparable from retrieval effectiveness.

Retrieval-augmented generation has been widely studied for accessing static knowledge bases [13, 7, 16], but agent memory presents a distinct challenge: the retrieval target itself is not given but constructed on the fly from a growing interaction stream. Existing methods have explored diverse strategies for this construction, including structured note generation, associative linking, and selective

*Equal contribution.
†Corresponding authors.

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



(a) Per-category top-20 retrieval recall.

(b) Downstream F1 under evidence dilution.

Figure 1: Diagnostic analysis on LoCoMo (A-MEM, Qwen3-8B). **(a)** Per-category top-20 retrieval recall. 28.2% of results are near-duplicates. **(b)** Downstream F1 when target evidence is diluted to 100%/50%/10% of the retrieved context. Setup details in Appendix A.

forgetting [21, 38, 34, 30, 36, 5]. Our diagnostic analysis on A-MEM [34] (Figure 1a) reveals that even when memory retains nearly all raw information, close to half of the ground-truth evidence fails to appear in retrieval results, as redundant entries crowd out relevant content. This retrieval failure varies systematically with query type: open-domain and multi-hop queries, which rely on different granularities of information, suffer the lowest recall. A controlled experiment (Figure 1b) further shows that this gap propagates to downstream reasoning: as target evidence is diluted among irrelevant entries, response quality degrades, with the steepest decline on the same categories that exhibit the lowest retrieval recall. While retaining all content buries evidence, compressing too aggressively risks discarding it altogether. This amounts to a tradeoff between compressing redundancy and preserving enough structure to retrieve target evidence, viewed through the information bottleneck principle [26] (Sec. 2.1).

These findings motivate organizing memory as a compression hierarchy. We propose MemCoRe (**Mem**ory **Co**mpression and **Re**covery), in which evidence is progressively compressed from detailed records through extracted keywords to topic groups, each retaining the structure needed for retrieval at that level. A gated mechanism consolidates each incoming observation at the evidence level by evaluating its relation to existing content: redundant information is merged, complementary evidence is linked, and novel content is retained. This enables retrieval to locate target evidence by searching across levels of the hierarchy: topic groups provide thematic entry points, keywords enable entity-level matching, and associative links preserve relational context for distinguishing relevant evidence from distractors, while also supporting multi-hop reasoning. For queries requiring multiple pieces of evidence, an iterative refinement protocol progressively updates the evidence pool.

The contributions of this work are as follows:

- We conduct a diagnostic analysis revealing that existing memory strategies fail to surface close to half of target evidence at retrieval time, and that this retrieval failure propagates to downstream reasoning with category-dependent severity. The information bottleneck provides an analytical perspective on the compression-retrievability tradeoff.
- We propose MemCoRe, which organizes memory as a compression hierarchy where evidence is progressively compressed from detailed records through keywords to topic groups. A gated construction mechanism consolidates observations at the evidence level, and retrieval operates across the hierarchy, with iterative refinement for multi-hop queries.
- On LoCoMo, MemCoRe achieves the highest reported question-weighted F1 across four backbone models, with the clearest statistical evidence on Qwen3-8B.

2 The MemCoRe Framework

The diagnostic analysis in Section 1 identifies a fundamental tension in memory construction: retaining too much content buries evidence, while compressing too aggressively discards it. The information bottleneck principle [26] provides an analytical perspective on this tradeoff between compressing redundancy and preserving enough structure to retrieve target evidence. Guided by this perspective, MemCoRe organizes memory as a compression hierarchy where each level compresses

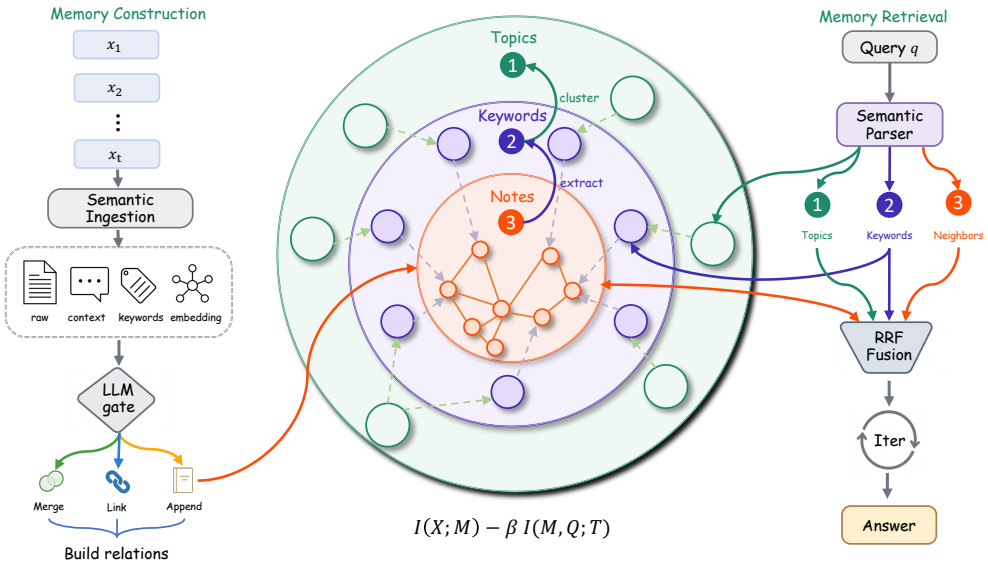


Figure 2: Overview of the MEMCORE framework. **Left**: Memory construction processes incoming observations through semantic ingestion and gated construction, where each observation is stored as a note and gated updates consolidate contexts or establish links based on its semantic relation to existing memory. **Center**: The memory state is organized as a compression hierarchy spanning notes, keywords, and topic groups, with associative edges connecting related notes. **Right**: Memory retrieval searches across levels of the hierarchy via topic, keyword, and neighbor expansion channels, followed by iterative evidence refinement for complex queries.

redundancy further while retaining the structure needed for retrieval at that level. Construction builds this hierarchy incrementally (Sec. 2.2), and retrieval operates across its levels to locate target evidence (Sec. 2.3). Figure 2 illustrates the overall architecture.

2.1 Problem Formulation

We consider the problem of agent memory: locating query-dependent evidence from the factual knowledge accumulated through a growing interaction history. An agent observes a stream of interactions $X_{1:t} = \{x_1, \ldots, x_t\}$. When a query Q arrives, the goal is to locate target evidence $T \subseteq X_{1:t}$. To manage retrieval as $|X_{1:t}|$ grows, the agent maintains a memory representation M_t constructed incrementally from $X_{1:t}$. For the IB analysis, M_t denotes the contexts and indexing structure used to locate evidence, excluding the immutable raw records. The central challenge is to construct M_t such that it compresses $X_{1:t}$ while preserving the structure needed to recover T for diverse queries.

Adapting the Information Bottleneck. The standard information bottleneck framework [26] characterizes a compressed representation Z of input X for predicting target Y through the Lagrangian $I(X;Z) - \beta\, I(Z;Y)$. We adapt this formulation to the memory retrieval setting by replacing the target Y with T and treating Q as an external query. The relevance term becomes $I(M_t, Q;\, T)$, reflecting that locating evidence is not a property of memory alone: the same memory state may surface different evidence depending on the query. The adapted Lagrangian is:

$$\mathcal{L}(M_t) = I(X_{1:t};\, M_t) - \beta\, I(M_t, Q;\, T), \tag{1}$$

where $\beta > 0$ controls the tradeoff between compression and query-relevant information. The first term $I(X_{1:t}; M_t)$ measures how much information about the interaction history is retained in the representation. The second term $I(M_t, Q; T)$ measures the information that the memory representation and query jointly contain about target evidence. A favorable memory representation achieves low $\mathcal{L}(M_t)$ by compressing the history while preserving information relevant to T. We use this formulation as an analytical framework. Evidence recall under a fixed retrieval procedure and budget does not directly measure mutual information.

From Analysis to Design. The diagnostic finding that retrieval failures vary systematically across query types motivates organizing memory at multiple levels of compression. In MEMCORE, evidence is progressively compressed from detailed records through extracted keywords to topic groups (Section 2.2). Within this hierarchy, for a fixed query–evidence distribution, the effect of incorporating a new observation x_{t+1} at the evidence level can be analyzed through the change in $\mathcal{L}$:

$$\Delta\mathcal{L} = \underbrace{I(X_{1:t+1}; M_{t+1}) - I(X_{1:t}; M_t)}_{\Delta I_{\text{compress}}} - \beta \underbrace{\big(I(M_{t+1}, Q; T) - I(M_t, Q; T)\big)}_{\Delta I_{\text{retrieve}}}. \tag{2}$$

When x_{t+1} largely duplicates an existing entry, **merging** their contexts consolidates redundant content while retaining both raw observations. When x_{t+1} is related to an existing entry but carries distinct information, **linking** them preserves both entries while encoding their relationship, providing relational context to help distinguish relevant evidence from distractors and additional retrieval paths for multi-hop reasoning. When x_{t+1} is independent of all existing entries, **appending** it preserves coverage, as x_{t+1} may be the sole evidence for future queries.

The implementation does not estimate or directly optimize these mutual-information terms. Instead, the design principles above depend on assessing the semantic relation between a new observation and existing memory, specifically whether the new content is redundant, complementary, or novel. Large language models, through pre-trained semantic understanding, provide a natural mechanism for this assessment (Section 2.2). Ablation results (Section 3.3) evaluate the contribution of each operation to downstream performance.

2.2 Memory Construction

The compression hierarchy in MEMCORE consists of three levels (Figure 2): notes store detailed evidence and denoised context at the finest level, keywords capture entity-level information as an intermediate compression, and topic groups cluster related keywords at the coarsest level. Associative links record semantic relationships between notes. Construction builds this hierarchy incrementally through three stages: semantic ingestion transforms each raw observation into a structured note, gated construction consolidates the note into existing memory, and topic evolution maintains topic groups over the keyword layer.

Semantic Ingestion. Raw observations often contain elliptical references, syntactic noise, and implicit context. We transform each raw observation x_t into a structured note n_t via an LLM-based ingestion function:

$$n_t = \mathcal{F}_{\text{ingest}}(x_t) = (r_t, c_t, \mathbf{h}_t, \mathcal{K}_t), \tag{3}$$

where r_t preserves the verbatim content, c_t is a denoised context summary generated to enhance retrieval relevance, $\mathbf{h}_t = \text{Embed}(c_t) \in \mathbb{R}^d$ is a dense embedding of the context, and $\mathcal{K}_t \subset \mathcal{K}$ is a set of keywords extracted from x_t. Retaining both r_t and c_t ensures that raw fidelity is preserved while retrieval operates over cleaner representations. Every observation is stored as a separate note, with its raw field r_t kept immutable throughout subsequent updates. Keywords provide an intermediate symbolic layer between notes and topic groups, grounding proximity in shared symbolic substructures rather than relying solely on embedding similarity.

Gated Construction. Following the design principles in Section 2.1, which reflect the diagnostic finding that construction decisions depend on the semantic relation between new content and existing memory, each new note n_t is consolidated through one of three operations. The process begins by retrieving a candidate set $\mathcal{N}_{\text{cand}}$ from existing memory through embedding similarity, restricting the evaluation to the most relevant entries. An LLM then evaluates all candidate pairs (n_t, n_i), where $n_i \in \mathcal{N}_{\text{cand}}$, jointly in a single call, classifying their semantic relation and estimating a connection strength $s(n_t, n_i) \in [0, 1]$ through structured prompting. Based on this evaluation, the mechanism selects one of three operations:

$$M_t \leftarrow \mathcal{O}(n_t, n_i) = \begin{cases} \textsc{Merge}(c_i \leftarrow c_i') & \text{if redundant} \\ \textsc{Link}(n_i \leftrightarrow n_t) & \text{if complementary} \\ \textsc{Append}(n_t) & \text{if novel} \end{cases}. \tag{4}$$

Merge. When n_t largely duplicates an existing entry n_i, related content is consolidated into the anchor's retrieval-facing context:

$$n_i \leftarrow \big(r_i,\ \ \mathcal{F}_{\text{merge}}(c_i, c_t),\ \ \text{Embed}(c_i'),\ \ \mathcal{K}_i\big), \tag{5}$$

where $\mathcal{F}_{\text{merge}}$ is an LLM-based function that synthesizes a unified context c_i' to incorporate relevant information from both notes. Raw content from both notes is retained in their original raw fields, and n_t remains independently stored. Merge consolidates the retrieval-facing context without reducing the number of stored observations.

Link. When n_t is related to n_i but carries distinct information, both notes are retained and an associative edge is established between them:

$$\mathcal{E}_{\text{RELATED}} \leftarrow \mathcal{E}_{\text{RELATED}} \cup \{(n_t, n_i,\ \rho,\ s)\}, \tag{6}$$

where $\rho \in \{\text{SUPPORTS}, \text{CONFLICTS}, \text{RELATED_TO}\}$ denotes the relation type and $s \in [0, 1]$ is the connection strength, both estimated by the LLM. For high-strength connections, the context of the neighboring note n_i is updated to incorporate relevant information from n_t, enabling partial information transfer without full merging.

Append. When n_t is independent of all existing entries, it remains an autonomous note without modifying existing contexts or adding associative links, preserving coverage for potentially novel evidence.

Topic Evolution. Keywords extracted from the same note are connected by co-occurrence edges, forming a keyword graph $\mathcal{G}_{\text{kw}}$. As new notes and keywords accumulate, the topic layer is periodically restructured by partitioning this graph:

$$\max_{\mathcal{T}}\ \mathcal{Q}(\mathcal{T}, \mathcal{G}_{\text{kw}}) \quad \text{s.t.} \quad \delta_{\min} \le |C_i| \le \delta_{\max},\ \forall C_i \in \mathcal{T}, \tag{7}$$

where $\mathcal{T} = \{C_1, \ldots, C_T\}$ is the set of topic clusters, $\mathcal{Q}$ denotes the modularity function, and $\delta_{\min}, \delta_{\max}$ are cardinality bounds. We employ the Leiden algorithm [27] for computational efficiency. Each topic C_i is represented by the centroid embedding of its constituent keywords.

Together, these three stages produce a memory state equipped with keywords, topic groups, and weighted associative links, each serving as input to a dedicated retrieval channel (Section 2.3).

2.3 Memory Retrieval

The compression hierarchy produced during construction (Section 2.2) directly enables multi-level retrieval. Topic groups provide entry points for thematic queries, keywords enable entity-level matching, and associative links preserve relational context for distinguishing relevant evidence from distractors, while also supporting multi-hop reasoning. For queries requiring evidence not directly accessible from initial retrieval, an iterative refinement protocol progressively updates the evidence pool.

Query Decomposition. Given a raw query q, an LLM-based semantic parser extracts two complementary signals: a topical description embedding $\mathbf{h}_{\text{topic}} \in \mathbb{R}^d$ that captures the query's thematic intent, and a set of entity embeddings $\mathbf{H}_{\text{keys}} = \{\mathbf{h}_{k_1}, \ldots, \mathbf{h}_{k_m}\}$ corresponding to the core entities mentioned in the query. These two signals drive the topic and keyword channels respectively.

Tri-Pathway Retrieval. The *topic channel* operates at the coarsest level of the hierarchy. Given $\mathbf{h}_{\text{topic}}$, we identify the top-$K_{\text{topic}}$ topic groups whose centroid embeddings are most similar to the query's thematic description, then retrieve all notes whose keywords belong to those groups. This channel provides broad thematic coverage, particularly for queries that do not reference specific entities.

The *keyword channel* operates at the intermediate level. Each keyword $k_j \in \mathcal{K}$ maintains a dense embedding $\mathbf{e}_j \in \mathbb{R}^d$. The top-$K_{\text{key}}$ keywords matching each entity embedding in $\mathbf{H}_{\text{keys}}$ are identified by cosine similarity, and all notes containing those keywords are retrieved. This channel provides precise entity-level access, particularly for queries that reference specific people, places, or events.

Table 1: Main results on LoCoMo benchmark using closed-source models (GPT series). We report F1 and BLEU-1 (%) scores across five categories. The best performance in each category is marked in **bold**, and the second best is underlined.

Model	Method	Category										Average	
		Multi Hop		Temporal		Open Domain		Single Hop		Adversarial		F1	BLEU
		F1	BLEU	F1	BLEU	F1	BLEU	F1	BLEU	F1	BLEU		
GPT-4o-mini	LoCoMo	25.02	19.75	18.41	14.77	12.04	11.16	40.36	29.05	**69.23**	**68.75**	39.74	33.47
	ReadAgent	9.15	6.48	12.60	8.87	5.31	5.12	9.67	7.66	9.81	9.02	9.89	7.87
	MemoryBank	5.00	4.77	9.68	6.99	5.56	5.94	6.61	5.16	7.36	6.48	6.99	5.73
	MemGPT	26.65	17.72	25.52	19.44	9.15	7.44	41.04	34.34	43.29	42.73	35.45	30.16
	A-mem	27.02	20.09	45.85	**36.67**	12.14	12.00	44.65	37.06	50.03	49.47	41.97	36.16
	Mem-0	**34.72**	**25.13**	45.93	35.51	22.64	15.58	43.65	37.42	30.15	27.44	38.70	32.07
	LightMem	32.09	24.14	44.84	30.41	22.75	15.89	44.68	36.11	30.13	28.69	38.59	30.85
	MemCoRe	32.11	24.48	**46.61**	31.84	**23.98**	**16.84**	**44.74**	**38.17**	51.48	51.96	**43.76**	**37.27**
GPT-4o	LoCoMo	28.00	18.47	9.09	5.78	16.47	14.80	**61.56**	**54.19**	**52.61**	**51.13**	44.12	**38.70**
	ReadAgent	14.61	9.95	4.16	3.19	8.84	8.37	12.46	10.29	6.81	6.13	9.98	8.07
	MemoryBank	6.49	4.69	2.47	2.43	6.43	5.30	8.28	7.10	4.42	3.67	6.13	5.15
	MemGPT	30.36	22.83	17.29	13.18	12.24	11.87	60.16	53.35	34.96	34.25	41.02	36.23
	A-mem	32.86	23.76	39.41	31.23	17.10	15.84	48.43	42.97	36.35	35.53	40.53	35.36
	Mem-0	35.13	27.56	**52.38**	**44.15**	17.73	15.92	39.12	35.43	25.44	24.19	36.59	32.25
	LightMem	35.21	**30.25**	40.28	26.44	24.57	18.32	51.83	39.2	29.66	21.52	41.31	30.89
	MemCoRe	**35.89**	29.24	39.78	27.12	**25.74**	**19.53**	49.08	43.05	48.24	48.92	**44.39**	**38.70**

The *neighbor expansion* channel operates at the evidence level. Starting from the anchor set $\mathcal{E}_{\text{anc}} = \mathcal{R}_{\text{topic}} \cup \mathcal{R}_{\text{key}}$ produced by the first two channels, we expand along the associative edges $\mathcal{E}_{\text{RELATED}}$ established during gated construction:

$$\mathcal{E}_{\text{expand}} = \{m \in \mathcal{N} \mid \exists\, n \in \mathcal{E}_{\text{anc}},\ (n, m) \in \mathcal{E}_{\text{RELATED}}\}. \tag{8}$$

Connection strength stored on each edge modulates score propagation, prioritizing neighbors with stronger semantic relationships. This channel retrieves evidence that is logically related to anchor notes but may not share surface-level similarity with the query. Results from the topic and keyword channels are combined via Reciprocal Rank Fusion (RRF) [3], which aggregates reciprocal ranks across channels without requiring score normalization. The fused ranking is merged with the expanded neighbor set, and the top-K_{final} notes form the final evidence pool:

$$\mathcal{E}_{\text{pool}} = \text{Top-}\, K_{\text{final}}\big(\text{score}_{\text{RRF}} \cup \mathcal{E}_{\text{expand}}\big). \tag{9}$$

Iterative Evidence Refinement. Complex queries may require evidence not directly accessible from the initial retrieval. We address this through an Iterative Evidence Refinement (IER) protocol that progressively updates the evidence pool. At each iteration, an LLM evaluates whether the current evidence pool sufficiently addresses the query. If gaps are identified, a refined sub-query is synthesized to target missing aspects, and retrieval is re-executed via the tri-pathway mechanism. Newly retrieved notes are added to the accumulated candidates after deduplication. The top-K_{final} notes are then reselected to update the evidence pool, and the process repeats until the LLM judges coverage as sufficient or the maximum iteration count $I_{\max}$ is reached.

3 Experiments

3.1 Experimental Setup

Dataset. We evaluate MemCoRe on the LoCoMo benchmark [19], a dataset designed to assess the long-term information synthesis capabilities of LLM agents. LoCoMo contains long-horizon conversations with interleaved topics and evolving entity states, making it a robust testbed for dynamic memory structures. To provide a granular analysis of memory performance, we evaluate on five distinct reasoning categories: Multi-Hop, Temporal, Open Domain, Single Hop, and Adversarial. More details can be found in Appendix D.1. Additional benchmark results and supplementary analyses are reported in Appendix D.

Evaluation Metrics. Following prior work [34], we report token-level F1 and BLEU-1 [22] to measure lexical overlap between generated responses and reference answers. For ablation studies, we additionally report Recall, measuring the proportion of ground-truth evidence retrieved, and Hit Rate, indicating whether any relevant evidence appears in the candidates.

Implementation Details. We implement MemCoRe using a graph-backed architecture with Neo4j, maintaining the compression hierarchy (notes, keywords, and topic groups) as an integrated

Table 2: Main results on LoCoMo benchmark using open-source models (Qwen series). We report F1 and BLEU-1 (%) scores across five reasoning categories. The best performance in each category is marked in **bold**, and the second best is underlined.

Model	Method	Category										Average	
		Multi Hop		Temporal		Open Domain		Single Hop		Adversarial			
		F1	BLEU	F1	BLEU	F1	BLEU	F1	BLEU	F1	BLEU	F1	BLEU
Qwen3-8B	LoCoMo	25.09	15.73	32.82	27.14	14.47	13.35	20.18	18.39	**46.77**	40.81	28.62	24.22
	ReadAgent	13.17	9.30	34.91	27.04	8.80	7.45	26.44	24.83	29.98	28.34	25.87	22.93
	MemoryBank	21.25	14.53	30.20	21.11	11.33	10.53	32.75	26.33	30.95	30.13	29.27	23.90
	MemGPT	22.13	13.44	31.47	22.16	14.51	13.54	33.49	34.12	34.58	31.44	30.88	27.65
	A-mem	24.30	16.90	34.50	23.10	13.10	12.20	38.10	33.30	31.00	30.10	32.76	27.58
	Mem-0	23.04	19.74	29.65	23.16	**20.63**	13.75	30.46	25.62	26.02	22.48	27.80	23.11
	LightMem	27.19	19.53	34.40	20.62	15.21	10.07	40.21	30.19	35.22	34.79	35.09	27.19
	MemCoRe	**28.24**	**22.76**	**38.39**	**33.64**	15.43	**13.81**	**42.09**	**36.57**	43.79	**43.14**	**38.62**	**34.51**
Qwen3-14B	LoCoMo	**33.37**	**24.26**	31.49	16.42	13.92	11.02	25.46	24.82	**49.17**	**35.00**	32.32	25.00
	ReadAgent	13.16	9.61	18.12	12.33	12.16	9.25	32.83	28.35	5.96	4.2	20.63	16.75
	MemoryBank	25.97	18.16	25.37	18.76	13.52	11.69	34.92	30.6	21.94	17.56	28.16	23.08
	MemGPT	24.12	15.41	25.48	19.04	13.44	12.64	34.74	32.41	27.11	24.32	28.99	25.06
	A-mem	21.36	14.98	23.06	18.04	12.62	11.49	35.43	30.92	26.71	25.78	28.37	24.48
	Mem-0	20.98	16.27	**31.50**	21.73	12.70	**13.22**	24.7	19.14	21.01	19.84	23.86	19.02
	LightMem	29.02	22.23	28.24	16.37	13.88	12.32	38.41	28.46	26.26	24.67	31.52	23.99
	MemCoRe	30.80	23.13	29.25	**24.56**	**14.11**	11.03	**42.25**	**35.52**	26.59	25.02	**33.65**	**28.45**

graph with both vector indices and explicit edges. All embeddings use `Qwen3-Embedding-0.6B`. For memory construction, the LLM classifies the semantic relation between each new observation and candidate entries, and estimates connection strength $s \in [0, 1]$ through structured prompting (Section 2.2). Topic evolution uses the Leiden algorithm with cluster size bounds $\delta_{\min} = 3$ and $\delta_{\max} = 15$. For retrieval, we set $K_{\text{topic}} = 3$ for topic-based retrieval, $K_{\text{key}} = 10$ for keyword matching, and $K_{\text{final}} = 60$ for the final evidence pool budget. Neighbor expansion performs 1-hop traversal along $\mathcal{E}_{\text{RELATED}}$ edges. The iterative evidence refinement protocol uses $I_{\max} = 3$ iterations. Full hyperparameter settings are provided in Appendix E.

Backbone Models and Baselines. We evaluate MemCoRe across four foundation models spanning closed-source (GPT [20]) and open-source (Qwen [23]) families: GPT-4o-mini, GPT-4o, Qwen3-8B, and Qwen3-14B. The generation temperature is set to 0.7 for general reasoning and 0.5 for adversarial tasks. We compare against seven representative methods: LoCoMo [19], ReadAgent [12], MemoryBank [38], MemGPT [21], A-MEM [34], Mem0 [2], and LightMem [5]. All baselines are implemented using their official system prompts and default configurations to ensure a fair comparison.

3.2 Main Results

Overall Performance. Tables 1 and 2 present performance comparisons on closed-source and open-source models, respectively. MemCoRe achieves the highest reported question-weighted F1 across all four backbone models. On GPT-4o-mini, it attains 43.76% F1, outperforming the second-best A-MEM by 1.79 points. On GPT-4o, MemCoRe achieves 44.39% F1, a narrower margin of 0.27 points over LoCoMo. This small gap does not establish a statistically reliable gain. However, MemCoRe maintains this edge while achieving a substantially more balanced category profile than LoCoMo, whose 52.61% Adversarial score coexists with only 9.09% on Temporal. On open-source models, the improvement is particularly notable on Qwen3-8B, where MemCoRe reaches 38.62% F1, surpassing the second-best LightMem by 3.53 points. On Qwen3-14B, MemCoRe achieves 33.65% F1, outperforming all memory-based baselines. Paired analysis provides the clearest evidence of improvement on Qwen3-8B (Appendix D.4). The smaller margins over the strongest baselines on the other three backbones do not establish statistically reliable gains.

Category-wise Analysis. MemCoRe shows differentiated strengths across query categories. For Single Hop tasks requiring precise entity matching, MemCoRe achieves top performance on three of four models, including 42.09% and 42.25% F1 on Qwen3-8B and Qwen3-14B respectively. This aligns with the ablation finding that keyword-based retrieval is critical for entity-level queries (Section 3.3). For Open Domain queries, MemCoRe achieves the highest F1 on both GPT models (23.98% and 25.74%), where the topic channel helps localize relevant memory regions. On Multi-Hop queries, MemCoRe achieves the best F1 on GPT-4o (35.89%) and Qwen3-8B (28.24%). Associative links provide retrieval paths across related evidence, although their largest ablation effect is on Adversarial queries (Section 3.3).

Table 3: Ablation study on LoCoMo (Qwen3-8B). The best performance in each category is marked in **bold**, and the second best within each group is underlined.

Phase	Method	F1	BLEU	Recall	Hit Rate
-	MEMCORE	**38.62**	**34.51**	**62.22**	**67.12**
Construction	w/o Update	27.97	24.89	42.11	48.20
	w/o Denoise	36.07	34.33	57.42	62.55
	w/o Link	33.57	31.91	53.19	56.18
	w/o Merge	34.79	31.28	54.85	59.42
Retrieval	w/o Topic	36.79	31.86	53.30	58.91
	w/o Keyword	32.69	30.48	51.28	54.26
	w/o Neighbor	34.26	30.28	51.28	54.35
	w/o IER	32.94	28.38	46.29	51.26

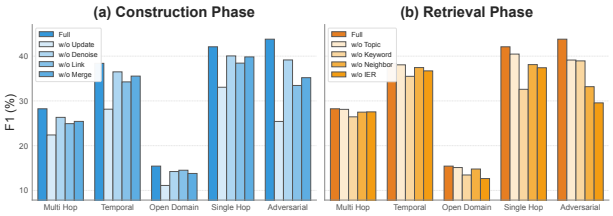


Figure 3: Category-wise F1 scores (%) for ablation variants on LoCoMo (Qwen3-8B). (a) Ablations on memory construction components. (b) Ablations on retrieval pathways and iterative refinement.

Balance and the Compression Tradeoff. Beyond per-category comparisons, MEMCORE achieves a more balanced performance profile across categories than baselines that excel in one area but fail in others. For instance, on GPT-4o, LOCOMO reaches 52.61% on Adversarial but only 9.09% on Temporal, while MEM0 achieves 52.38% on Temporal but only 25.44% on Adversarial. In contrast, MEMCORE maintains competitive scores across all categories without severe degradation on any. This balance reflects the compression hierarchy design: by providing retrieval structure at multiple levels, the system avoids the extremes of over-retention and over-compression. On GPT models, the largest gains appear on the categories identified as most challenging in the diagnostic analysis. On open-source models, improvements concentrate on Single-Hop and Multi-Hop categories. The main exception is Adversarial on Qwen3-14B, where LOCOMO (49.17%) substantially outperforms MEMCORE (26.59%). Our diagnostic analysis suggests that this gap is associated with over-abstention under bounded retrieval context. However, MEMCORE achieves the highest question-weighted F1 on this backbone.

3.3 Ablation Study

We validate the compression hierarchy by ablating its construction and retrieval components separately. Results are shown in Table 3 and Figure 3.

Construction Ablations. We first assess how each construction stage contributes to the quality of the compression hierarchy. Removing the entire gated construction (*w/o Update*), which reduces the system to append-only storage without merge or link operations, causes the most severe degradation: average F1 drops from 38.62% to 27.97% and Recall from 62.22% to 42.11%. As shown in Figure 3(a), this variant exhibits the largest performance gap across all five categories. This recall drop is consistent with redundant entries competing for a limited retrieval budget and supports the contribution of gated construction to evidence recovery.

Among individual operations, *w/o Link* shows a larger F1 decrease than *w/o Merge* (33.57% vs 34.79% F1). Figure 3(a) reveals that Link removal particularly affects Adversarial performance, indicating that associative edges help preserve relational structure that distinguishes relevant evidence from distractors. The larger impact of Link over Merge suggests that associative linking contributes more to downstream F1 than context consolidation in this setting. The *w/o Denoise* variant shows the mildest degradation among construction ablations (36.07% F1), with relatively stable results across all categories as shown in the figure, indicating that semantic preprocessing provides consistent but auxiliary improvements.

Retrieval Ablations. We then assess how each retrieval channel exploits the compression hierarchy. The *w/o Topic* variant has the smallest F1 decrease (36.79% F1), with Figure 3(b) showing relatively uniform impact across categories. This indicates that topic-level retrieval provides general guidance that other channels can partially compensate for. In contrast, *w/o Keyword* (32.69% F1) and *w/o Neighbor* (34.26% F1) show more pronounced impacts. As illustrated in Figure 3(b), Keyword removal causes the most pronounced decline on Single-Hop queries, validating that entity-level matching at the intermediate compression level is essential for precise retrieval. The *w/o Neighbor* variant primarily impacts Adversarial queries, confirming that traversal along associative edges helps retrieve evidence that is logically related but not directly similar to the query.

The *w/o IER* variant (32.94% F1) produces the largest Recall drop among retrieval ablations (62.22% $\rightarrow$ 46.29%), demonstrating that iterative refinement is critical for reaching evidence not directly

accessible from the initial retrieval. Figure 3(b) shows the steepest relative decline on Adversarial and Open Domain categories, where queries often require progressive evidence accumulation. Multi-hop queries are already partially served by the neighbor expansion channel, which traverses associative links between related evidence. Adversarial and Open Domain queries, which require broader evidence coverage without clear link paths, benefit more from iterative re-retrieval.

The most severe degradation overall comes from removing gated construction entirely (-10.65 F1), indicating that the quality of the compression hierarchy built during construction constrains what retrieval can recover. Notably, link removal (construction) and neighbor expansion removal (retrieval) both primarily impact Adversarial, confirming that the associative structure built during gated construction is what the neighbor channel exploits. Keyword-based retrieval primarily impacts Single-Hop, validating that the entity-level structure produced during ingestion enables precise matching.

4 Related Work

4.1 Memory Systems for LLM Agents

A growing body of work addresses long-term memory for language agents [28], with methods differing primarily in how they construct and manage stored representations.

Tiered storage and context management. MemGPT [21] introduces an OS-inspired virtual memory hierarchy that pages information between working context and archival storage, enabling effectively unbounded conversation history. HiAgent [10] extends this idea with a hierarchical working memory that organizes context at multiple granularities. LightMem [5] proposes a lightweight tiered architecture that compresses memory across layers for efficiency. These systems focus on where information resides but leave the decision of what to retain largely to recency or capacity heuristics.

Forgetting and compression. MemoryBank [38] applies the Ebbinghaus forgetting curve to selectively discard older entries, prioritizing recency over completeness. SimpleMem [14] introduces a three-stage pipeline that distills interactions into compact, semantically indexed units. These methods reduce memory footprint but risk discarding content that may later prove relevant.

Associative organization. A-MEM [34] generates structured notes with keywords and tags for each observation, and establishes associative links inspired by Zettelkasten-style knowledge management. O-Mem [30] extends this direction with user-centric profiling, while GMemory [36] maintains hierarchical representations that trace memory evolution across sessions. These approaches enrich memory structure but create a new entry for every observation without consolidating redundant content.

Fact extraction. Mem0 [2] extracts structured facts from interactions and maintains them through add, update, and delete operations over a hybrid vector-graph store. Zep [24] tracks factual evolution through a temporal knowledge graph. These systems operate at the fact level rather than the evidence level, which may lose contextual detail needed for complex reasoning. Our diagnostic analysis (Figure 1) reveals that despite these diverse strategies, close to half of target evidence remains unreachable during retrieval, with the severity varying systematically across query types.

4.2 Retrieval-Augmented Generation

Retrieval-augmented generation (RAG) [13, 7] retrieves relevant passages from static corpora to ground language model outputs. Structural approaches organize documents hierarchically through recursive summarization [25] or graph representations [4, 32, 17, 18, 37] to support multi-granularity retrieval. LightRAG [8] combines graph indexing with dual-level retrieval and incremental updates. HippoRAG [9] uses knowledge graphs and personalized PageRank for associative retrieval. HyperGraphRAG [15] represents n-ary relations with hyperedges. Iterative methods refine queries and filter outputs at inference time [1, 35]. Our focus is on per-observation memory construction from growing interaction streams, with immutable raw records and retrieval-facing context consolidation.

5 Conclusion

We presented MEMCORE, a memory system for LLM agents built around hierarchical compression. A diagnostic analysis reveals that close to half of target evidence remains unreachable during retrieval, with failures varying systematically across query types. Viewed through the information bottleneck, this amounts to a tradeoff between compression and retrievability. We proposed MEMCORE, which organizes memory as a compression hierarchy where evidence is progressively compressed through keywords to topic groups, each level retaining the structure needed for retrieval. A gated construction mechanism populates this hierarchy by consolidating each observation based on its relation to existing content, and retrieval operates across levels of the hierarchy to locate target evidence. On LoCoMo, MEMCORE achieves the highest reported question-weighted F1 across four backbone models, with the clearest statistical evidence on Qwen3-8B.

Limitations. The current implementation prioritizes memory quality over construction speed, introducing computational overhead from LLM-based semantic evaluation at each construction step. The information bottleneck serves as an analytical framework in this work. Directly optimizing the IB objective for memory construction is a direction we plan to explore. Extending evaluation to additional benchmarks, multi-modal interactions, and domain-specific scenarios also remains future work.

Acknowledgments and Disclosure of Funding

Zhenyuan Zhang, Xianzhang Jia, Zhiqin Yang, Wei Xue, Sirui Han, and Yike Guo were supported by the Hong Kong Generative AI Research & Development Center. This work was also supported by the Theme-based Research Scheme of the Research Grants Council of Hong Kong (T45-205/21-N). The authors declare no competing interests.

References

[1] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection, 2023.

[2] Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory, 2025.

[3] Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, 2009.

[4] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From local to global: A graph rag approach to query-focused summarization, 2025.

[5] Jizhan Fang, Xinle Deng, Haoming Xu, Ziyan Jiang, Yuqi Tang, Ziwen Xu, Shumin Deng, Yunzhi Yao, Mengru Wang, Shuofei Qiao, Huajun Chen, and Ningyu Zhang. Lightmem: Lightweight and efficient memory-augmented generation, 2025.

[6] Mohamed Amine Ferrag, Norbert Tihanyi, and Merouane Debbah. From llm reasoning to autonomous ai agents: A comprehensive review, 2025.

[7] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey, 2024.

[8] Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. LightRAG: Simple and fast retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 10746–10761. Association for Computational Linguistics, 2025.

[9] Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. HippoRAG: Neurobiologically inspired long-term memory for large language models. In *Advances in*

Neural Information Processing Systems, volume 37, pages 59532–59569. Curran Associates, Inc., 2024.

[10] Mengkang Hu, Tianxing Chen, Qiguang Chen, Yao Mu, Wenqi Shao, and Ping Luo. Hiagent: Hierarchical working memory management for solving long-horizon agent tasks with large language model, 2024.

[11] Yuanzhe Hu, Yu Wang, and Julian McAuley. Evaluating memory in LLM agents via incremental multi-turn interactions. In *International Conference on Learning Representations*, 2026.

[12] Kuang-Huei Lee, Xinyun Chen, Hiroki Furuta, John Canny, and Ian Fischer. A human-inspired reading agent with gist memory of very long contexts, 2024.

[13] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021.

[14] Jiaqi Liu, Yaofeng Su, Peng Xia, Yiyang Zhou, Siwei Han, Zeyu Zheng, Cihang Xie, Mingyu Ding, and Huaxiu Yao. Simplemem: Efficient lifelong memory for llm agents. *arXiv preprint arXiv:2601.02553*, 2025.

[15] Haoran Luo, Haihong E, Guanting Chen, Yandan Zheng, Xiaobao Wu, Yikai Guo, Qika Lin, Yu Feng, Zemin Kuang, Meina Song, Yifan Zhu, and Anh Tuan Luu. HyperGraphRAG: Retrieval-augmented generation via hypergraph-structured knowledge representation. In *Advances in Neural Information Processing Systems*, volume 38, pages 152206–152234. Curran Associates, Inc., 2025.

[16] Linhao Luo, Zicheng Zhao, Gholamreza Haffari, Yuan-Fang Li, Chen Gong, and Shirui Pan. Graph-constrained reasoning: Faithful reasoning on knowledge graphs with large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 41540–41565. PMLR, 2025.

[17] Linhao Luo, Zicheng Zhao, Reza Haffari, Dinh Phung, Chen Gong, and Shirui Pan. GFM-RAG: Graph foundation model for retrieval augmented generation. In *Advances in Neural Information Processing Systems*, volume 38, pages 36371–36405, 2025.

[18] Linhao Luo, Zicheng Zhao, Junnan Liu, Zhangchi Qiu, Junnan Dong, Serge Panev, Chen Gong, Thuy-Trang Vu, Gholamreza Haffari, Dinh Phung, Alan Liew, and Shirui Pan. G-reasoner: Foundation models for unified reasoning over graph-structured knowledge. In *International Conference on Learning Representations*, volume 2026, pages 63769–63793, 2026.

[19] Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of llm agents, 2024.

[20] OpenAI. Gpt-4 technical report, 2024.

[21] Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. Memgpt: Towards llms as operating systems, 2024.

[22] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Annual Meeting of the Association for Computational Linguistics*, 2002.

[23] Qwen. Qwen3 technical report, 2025.

[24] Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. Zep: A temporal knowledge graph architecture for agent memory, 2025.

[25] Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. Raptor: Recursive abstractive processing for tree-organized retrieval. In *International Conference on Learning Representations*, 2024.

[26] Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. In *Proceedings of the 37th Annual Allerton Conference on Communication, Control, and Computing*, pages 368–377, 1999.

[27] V. A. Traag, L. Waltman, and N. J. van Eck. From louvain to leiden: guaranteeing well-connected communities. *Scientific Reports*, 9(1), March 2019.

[28] Boyuan Wang, Zhenyuan Zhang, Zhiqin Yang, Peijun Gu, Shuya Wang, Xiaofeng Wang, Xianghui Ze, Yifan Chang, Guosheng Zhao, Jiangnan Shao, Guan Huang, Hengyu Liu, Yonggang Zhang, Wei Xue, Chunyuan Guan, Chenglin Pu, Yike Guo, Xingang Wang, and Zheng Zhu. Zero2Skill: Bootstrapping robot skills through autonomous data collection, training, and deployment, 2026.

[29] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Jirong Wen. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6), March 2024.

[30] Piaohong Wang, Motong Tian, Jiaxian Li, Yuan Liang, Yuqing Wang, Qianben Chen, Tiannan Wang, Zhicong Lu, Jiawei Ma, Yuchen Eleanor Jiang, and Wangchunshu Zhou. O-mem: Omni memory system for personalized, long horizon, self-evolving agents, 2025.

[31] Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. LongMemEval: Benchmarking chat assistants on long-term interactive memory. In *International Conference on Learning Representations*, 2025.

[32] Xiaojun Wu, Cehao Yang, Xueyuan Lin, Chengjin Xu, Xuhui Jiang, Yuanliang Sun, Hui Xiong, Jia Li, and Jian Guo. Think-on-graph 3.0: Efficient and adaptive llm reasoning on heterogeneous graphs via multi-agent dual-evolving context retrieval, 2025.

[33] Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, Rui Zheng, Xiaoran Fan, Xiao Wang, Limao Xiong, Qin Liu, Yuhao Zhou, Weiran Wang, Changhao Jiang, Yicheng Zou, Xiangyang Liu, Zhangyue Yin, Shihan Dou, Rongxiang Weng, Wensen Cheng, Qi Zhang, Wenjuan Qin, Yongyan Zheng, Xipeng Qiu, Xuanjing Huan, and Tao Gui. The rise and potential of large language model based agents: A survey. *ArXiv*, abs/2309.07864, 2023.

[34] Wujiang Xu, Kai Mei, Hang Gao, Juntao Tan, Zujie Liang, and Yongfeng Zhang. A-mem: Agentic memory for llm agents, 2025.

[35] Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. Corrective retrieval augmented generation, 2024.

[36] Guibin Zhang, Muxin Fu, Guancheng Wan, Miao Yu, Kun Wang, and Shuicheng Yan. G-memory: Tracing hierarchical memory for multi-agent systems, 2025.

[37] Zicheng Zhao, Linhao Luo, Junnan Dong, Haoran Luo, Xiaoli Li, Shirui Pan, and Chen Gong. HyperReCo: Retrieving and connecting evidence with hypergraph neural networks for LLM multi-hop reasoning. *arXiv preprint arXiv:2609.32327*, 2026.

[38] Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory, 2023.

A Diagnostic Experiment Details

This section describes the experimental setup for the diagnostic analysis reported in Figure 1. Both experiments use the LoCoMo benchmark (Appendix D.1) and A-MEM [34] as the representative structured memory system.

Shared setup. A-MEM processes each conversational turn into a structured note with metadata and associative links, following the Zettelkasten knowledge management method. Running A-MEM on the full LoCoMo dataset produces $5,882$ memory notes across 10 conversations. All embeddings use `Qwen3-Embedding-0.6B` for both A-MEM and MEMCORE to ensure a fair comparison. To identify ground-truth evidence for each query, we parse the dialogue IDs annotated in LoCoMo's evidence field, reconstruct the corresponding utterances, and match them against stored notes by exact content matching.

A.1 Experiment 1: Retrieval Diagnosis

This experiment characterizes retrieval quality without any LLM inference. For each of the $1,986$ queries, we retrieve the top-20 most similar notes using cosine similarity and measure three quantities:

1. **Storage-level redundancy**: the distribution of pairwise cosine similarity across all 5,882 notes, evaluated at thresholds 0.7, 0.8, and 0.9.
2. **Retrieval-level redundancy**: the fraction of note pairs within each top-20 result set whose cosine similarity exceeds 0.8.
3. **Top-20 recall**: the fraction of ground-truth evidence notes that appear in the top-20 results, reported per query category.

Results are computed using note embeddings to isolate the effect of semantic content on retrieval quality.

A.2 Experiment 2: Evidence Dilution

This experiment measures how downstream reasoning quality degrades when target evidence is diluted among irrelevant entries, isolating the causal effect of evidence concentration on task performance.

Dilution protocol. For each query, we identify its ground-truth notes and control the fraction of target evidence in the input context. Given a target evidence ratio $r \in \{1.0, 0.5, 0.1\}$, we inject $\lfloor |GT| \cdot (1/r - 1) \rceil$ distractor notes, where $|GT|$ is the number of ground-truth notes. Distractors are sampled from non-ground-truth notes within the same conversation and shuffled with the ground-truth notes before concatenation.

Prompt and evaluation. A category-agnostic prompt ("Based on the following memory notes, answer the question. Be concise.") is used to isolate dilution as the sole experimental variable. Responses are generated by Qwen3-8B and evaluated by token-set F1 against the gold answer.

Scope. We evaluate 1,531 queries with matched ground-truth evidence from categories 1–4 (single-hop, multi-hop, temporal, open-domain). Category 5 (adversarial) is excluded because its gold answers are refusal phrases (e.g., "Not mentioned in the conversation"). Under the category-agnostic prompt, the model consistently generates substantive answers rather than appropriate refusals, producing near-zero F1 regardless of dilution ratio (F1 $\leq$ 1.04 across all settings). Including this category would conflate prompt design effects with the dilution signal under study.

A.3 Adversarial Answering Behavior

For MEMCORE on the 446 Adversarial questions, Qwen3-14B abstains in 65.6% of cases with target evidence present in the retrieved context, compared with 44.0% for Qwen3-8B. Conditional on answering, F1 is similar (97.5 versus 97.6), while retrieval hit rates are 60.5% and 65.7%, respectively. The lower 14B category score is associated with more frequent abstention under bounded retrieval context, consistent with the explanation in Section 3.2.

B Detailed Analysis of Construction Operations

This appendix provides a more detailed analysis of the construction operations in relation to the two quantities in the IB Lagrangian (Eq. (1)), expanding on the design principles presented in Section 2.1.

By the chain rule for mutual information,

$$I(M_t, Q; T) = I(Q; T) + I(M_t; T \mid Q).$$

For a fixed query–evidence distribution, $I(Q; T)$ is independent of memory construction and cancels in the relevance difference in Eq. (2). We retain the joint relevance term in Eq. (1).

B.1 Incremental Analysis

Let n_i denote a candidate anchor for the incoming note n_t. All three operations retain n_t and its raw content. They differ in their updates to contexts and links. The mutual-information changes depend on these updates and the query–evidence distribution.

Merge. When n_t is highly redundant with n_i, Merge consolidates their contexts into the anchor context c_i' (Eq. (5)). Both notes and their raw content remain stored. This is intended to concentrate related information for retrieval. Unchanged retrievability is not guaranteed.

Append. When n_t is largely independent of existing entries, appending it as a new entry increases memory size but may capture evidence for future queries. The gate selects this operation based on semantic novelty, without explicitly comparing mutual-information gains and costs.

Link. When n_t is semantically related to n_i but carries distinct information, linking them preserves both entries while encoding their relationship through an associative edge. The edge metadata (relation type ρ and connection strength s) provides an additional retrieval signal: during neighbor expansion, edge weights modulate score propagation, directing retrieval toward more informative paths. This can improve access to evidence for queries requiring multi-hop reasoning, where evidence is distributed across multiple entries.

B.2 Candidate Selection

The gated construction (Section 2.2) requires evaluating the semantic relation between x_t and existing memory entries. Computing this for all entries is impractical, so MEMCORE uses a two-stage filtering approach:

Stage 1: Vector and keyword candidate selection. The candidate pool combines the top five entries by cosine similarity between their context embeddings and $\mathbf{h}_t = \text{Embed}(c_t)$ with the top five by keyword overlap. The fused pool is reduced to at most five candidates for semantic evaluation. This serves as a computationally efficient filter that identifies entries most likely to have high semantic overlap with x_t.

Stage 2: LLM-based semantic evaluation. Up to five top-ranked candidates are evaluated jointly in a single LLM call, which classifies the semantic relation (redundant, complementary, or novel) and estimates connection strength $s \in [0, 1]$. This stage handles cases where embedding similarity alone is insufficient to determine the appropriate operation, such as when two entries discuss the same topic from different perspectives.

B.3 Construction Audits

Merge audit. In the audited memory, 70.4% of notes have undergone a Merge context update. Comparing 250 merged notes with 250 unmerged controls, 78.0% of merged contexts are augmented while retaining the original observation, and 12.4% are displaced (95% CI $[8.9, 17.1]$), meaning that the original content is lost or misattributed in the context. Temporal disorder and contradiction occur in 1.2% and 2.0% of the merged sample, respectively. 98.8% of controls are faithful. The audit compares context with each note's own raw content. It measures context fidelity, while raw observations remain stored independently.

Temporal evidence check. A check of all 5,882 cached notes confirmed that the original observations and session dates were retained. On 42 Temporal questions covered by simple relative-date rules, dates resolved using gold evidence and session dates matched the reference answers in 41 cases (97.6%), including all 27 cases whose evidence notes had undergone Merge context updates. The remaining case involved a discrepancy between the session year and the reference-answer year and was retained in the evaluation. This diagnostic measures date resolution given gold evidence rather than end-to-end Temporal QA performance.

Gate decisions and memory structure. A 200-pair gate audit records 137 Append, 47 Link, and 16 Merge decisions, with 36% agreement with an embedding-similarity rule using the same thresholds. These pair-level counts have a different denominator from the note-level Merge rate above. A structural census of LoCoMo conversation 0 gives 419 notes for 419 observations, 765 distinct keywords, and 128 topic groups. This hierarchy retains one note per observation. Its structure does not establish a reduction in total stored tokens.

Implementation details. Session dates are stored as note metadata. Topic reclustering is counter-triggered with a final pass after construction. Context updates preserve raw records. Regeneration of a context from these records remains an engineering direction.

C Prompting Templates

This appendix provides the complete prompting templates used in MEMCORE. All prompts are designed to elicit structured JSON outputs for reliable parsing.

C.1 Memory Construction Prompts

C.1.1 Semantic Ingestion Prompt

During the ingestion phase (Section 2.2), raw conversational input x_t is transformed into a structured note $n_t = (r_t, c_t, \mathbf{h}_t, \mathcal{K}_t)$. The following prompt instructs the LLM to extract keywords $\mathcal{K}_t$ and generate the denoised context c_t:

Semantic Ingestion Prompt

You are an expert Knowledge Graph Extractor. Your task is to analyze the [TARGET TURN] to extract structured metadata.
Input Text: {content}
Guidelines:
1. Keywords (Entities):
- **GOAL**: Extract 3-5 specific Noun Phrases explicitly present in the text.
- **FOCUS PRIORITIES**:
  1. Proper Nouns: People (e.g., "Melanie"), Locations, Organizations.
  2. Concrete Objects: Physical items (e.g., "painting", "plate", "contract").
  3. Specific Topics: "LGBTQ support group", "deadline".
- **CRITICAL STOP LIST**: Ignore conversational meta-roles, abstract terms, and speaker names acting purely as subjects.
- **ZERO-SHOT RULE**: If the text is purely phatic (e.g., "Wow", "That's cool"), return an empty list.

2. Context (Factual Restatement):
- **GOAL**: Rewrite the text into a self-contained factual statement.
- **CONSTRAINT 1 (Strict Fidelity)**: Only use information present in the Input Text.
- **CONSTRAINT 2 (Safe Resolution)**: Resolve "I/my/we" using the Speaker's name if it appears in the text. For external pronouns where the antecedent is missing, keep the pronoun or use a generic term. Do not guess.
- **CONSTRAINT 3 (No Meta-Language)**: Start directly with the subject. Avoid "The speaker says...".

Output Format (JSON):

```
{"keywords": ["entity1", "entity2"], "context": "Melanie thinks the item is cool."}
```

The extracted keywords are matched against the existing keyword index $\mathcal{K}$ to establish symbolic anchors, while the context is encoded via the embedding model to obtain $\mathbf{h}_t = \text{Embed}(c_t)$. This dual extraction enables both symbolic and semantic access pathways during retrieval.

C.1.2 Gated Construction Prompt

During gated construction (Section 2.2), the LLM evaluates the semantic relation between a new note n_t and each candidate note $n_i \in \mathcal{N}_{\text{cand}}$. The following prompt classifies the relation type and

estimates connection strength $s(n_t, n_i) \in [0, 1]$, determining the appropriate construction operation (Merge, Link, or Append).

Gated Construction Prompt

Role: You are a Knowledge Graph Updater. Your job is to evaluate the relationship between a NEW NODE and existing CANDIDATE NODES.

[NEW NODE]
Content: "{content}"
Context: "{context}"
Keywords: {keywords}

[CANDIDATE NODES]
{candidates_str}

Instructions:
Analyze each candidate and generate a JSON response following these rules:
1. Analyze Relationship:
- Determine `relation_type`: 'SUPPORTS', 'CONFLICTS', or 'RELATED_TO'.
- Assign `connection_strength` (0.0 – 1.0), indicating the degree of semantic overlap or logical connection.

2. Determine Operation (Based on Strength):
- **CASE A: Strength $\geq$ 0.7 (High Redundancy) $\rightarrow$ MERGE**
  - Action: Integrate details from the New Node into the candidate's context.
  - Template: "[Original Context]. Specifically, [New Node Info]..."
- **CASE B: Strength $\in$ [0.5, 0.7) (Complementary) $\rightarrow$ LINK**
  - Action: Establish associative edge; keep contexts separate.
  - Template: "[Original Context]. (Related: [New Node Keyword])"
- **CASE C: Strength $<$ 0.5 (Distinct) $\rightarrow$ APPEND**
  - Action: Add New Node as autonomous unit; no modification.
- **CASE D: relation_type is 'CONFLICTS' $\rightarrow$ LINK with Contrast**
  - Action: Note the conflicting information explicitly.
  - Template: "[Original Context]. However, [New Node] indicates that..."

3. Output: Return strictly valid JSON matching the schema.

Mapping to Paper Notation. The `connection_strength` output corresponds to $s(n_t, n_i) \in [0, 1]$ in Section 2.2, and `relation_type` corresponds to $\rho \in \{\text{SUPPORTS}, \text{CONFLICTS}, \text{RELATED_TO}\}$. The thresholds specified in the prompt (0.7 for merge, 0.5 for link) guide the LLM's classification into the three operation regimes defined in Eq. (4). When the Merge operation is triggered, the LLM generates a unified context $c_i' = \mathcal{F}_{\text{merge}}(c_i, c_t)$ following the template specified in Case A, aiming to preserve all distinct information from both notes while reducing redundancy. Full hyperparameter settings are provided in Appendix E.

C.2 Memory Retrieval Prompts

C.2.1 Query Intent Analysis Prompt

During the query decomposition step (Section 2.3), the raw query q is processed by a semantic parser to extract retrieval signals. The following prompt extracts a topical description and entity keywords for driving the topic and keyword channels.

Query Intent Analysis Prompt

Analyze the user query to identify its **Target Taxonomy Category**, extract key entities, and detect time-related intent.

Task 1: topic_desc (Target Category)
- Predict the **Taxonomy Category** or **Subject Heading** this query falls under.
- **Style**: Strict Noun Phrase (like a book chapter title or library category).
- **Constraint**: Keep it under 8 words.
- Do NOT describe the user's intent (e.g., avoid "how to...", "techniques for..."). Instead, name the topic itself.

Task 2: Keywords
- Extract 3-5 core entities, technical terms, or specific concepts.
- **CRITICAL**: Convert terms to their canonical singular form (e.g., "transformers" $\rightarrow$ "Transformer").
- Exclude generic verbs or stop words.

Query: {query}

Output Format (JSON):

```
{"topic_desc": "Concise Noun Phrase",
 "keywords": ["keyword1", "keyword2", "keyword3"]}
```

Mapping to Paper Notation. The `topic_desc` field is encoded via the embedding model to obtain $\mathbf{h}_{\text{topic}} \in \mathbb{R}^d$, which drives the topic channel described in Section 2.3. The `keywords` are similarly embedded to form $\mathbf{H}_{\text{keys}} = \{\mathbf{h}_{k_1}, \ldots, \mathbf{h}_{k_m}\}$, driving the keyword channel.

C.2.2 Iterative Evidence Refinement Prompts

The Iterative Evidence Refinement protocol (Section 2.3) employs two complementary prompts: a *Sufficiency Evaluator* that assesses whether the current evidence pool adequately addresses the query, and a *Sub-query Generator* that synthesizes targeted follow-up queries when gaps are identified.

Sufficiency Evaluation Prompt. At each iteration i, the following prompt evaluates whether the current evidence pool $\mathcal{E}^{(i)}$ sufficiently addresses the query.

Sufficiency Evaluation Prompt

You are a reflector agent that evaluates whether the current context and answer is sufficient to answer a question.

Question: {question}

Current Context and Current Answer:
{context}

Evaluate whether the provided context and answer contains enough information to answer the question comprehensively.
If the context and answer is insufficient, identify what specific information is missing.

Output Format (JSON):

```
{"sufficient": true or false,
 "missing_info": "description of missing information",
 "confidence": 0.0 to 1.0}
```

Sub-query Generation Prompt. When the sufficiency evaluation returns `sufficient: false`, the following prompt generates a refined sub-query $q^{(i+1)}$ targeting the identified information gaps.

Sub-query Generation Prompt

You are a Query Evolution Agent. Your goal is to decompose a complex user question into a specific, actionable sub-query to retrieve missing information from a Knowledge Graph.

Original Question: "{query_str}"

Current Known Information (Context):
{context_str}

History of Reasoning Steps (Q&A):
{prev_reasoning}

CRITICAL: The Reflector Agent has identified the following MISSING INFORMATION needed to answer the main question:
{missing_info}

Task: Based on the "Missing Information" and "History", formulate the **NEXT** single sub-question to retrieve this missing info.
- The sub-question must be specific.
- It should act as a search query for the next hop.
- If we have enough information to answer the main question, return 'None'.

Sub-question:

IER Protocol Flow. The two prompts work in tandem: the Sufficiency Evaluator determines whether to terminate (when `sufficient: true` or iteration count reaches $I_{\max}$), while the Sub-query Generator drives evidence expansion by producing targeted queries that are re-executed through the tri-pathway retrieval mechanism (Section 2.3). The `confidence` score from the Sufficiency Evaluator can optionally be used for early termination when confidence exceeds a predefined threshold.

D Dataset Statistics and Supplementary Results

D.1 LoCoMo Dataset Statistics

Table 4 presents the sample distribution across the five reasoning categories. The categories are designed to test distinct memory capabilities: *Multi-Hop* requires synthesizing evidence across multiple memory units. *Temporal* tests reasoning about time-dependent information and event ordering. *Open Domain* evaluates retrieval of general knowledge from conversation history. *Single*

Hop assesses precise entity matching and direct fact retrieval. *Adversarial* challenges the system with distractors and misleading information.

As shown in Table 4, the category distribution is imbalanced, with Single Hop comprising the largest proportion (42.3%) and Open Domain the smallest (4.8%). To account for this imbalance, the average scores reported in Table 1 and Table 2 are computed as **weighted averages** based on category sample sizes, ensuring that performance on larger categories contributes proportionally to the overall evaluation. The BLEU-1 scores in Table 3 use the same weighting.

Table 4: LoCoMo benchmark category distribution.

Category	Samples	Proportion
Multi-Hop	282	14.2%
Temporal	321	16.2%
Open Domain	96	4.8%
Single Hop	841	42.3%
Adversarial	446	22.5%
Total	1,986	100%

D.2 LongMemEval

We evaluate all 500 questions of LongMemEval-S [31], covering six question types, with Qwen3-8B and `Qwen3-Embedding-0.6B` for embedding-based retrieval. All methods share the same judge and benchmark evaluation rubric. MEMCORE retains its LoCoMo hyperparameters. Table 5 reports judge accuracy. The full-history baseline uses a 32K-token context window for histories averaging approximately 115K tokens, so its result reflects this context limit.

Table 5: LongMemEval-S judge accuracy (%, 500 questions, Qwen3-8B). All eight methods use the same judge and evaluation rubric.

Method	Accuracy
Full-history (32K)	34.8
ReadAgent	32.8
MemoryBank	40.4
MemGPT	34.2
A-MEM	54.6
Mem0	28.6
LightMem	59.6
MEMCORE	**64.2**

D.3 MemoryAgentBench

Fact consolidation. We evaluate single-hop (FC-SH) and multi-hop (FC-MH) FactConsolidation on MemoryAgentBench [11], with a 262K-token history ingested in 512-token chunks. We use GPT-4o-mini, the benchmark's latest-fact query template, and substring exact match. Ingestion extracts keywords exhaustively for fact-dense chunks, and retrieval covers approximately one eighth of stored notes ($k = 64$). The FactConsolidation results in Table 6 use single-shot retrieval with IER disabled. In a separate FC-MH evaluation, enabling IER increases accuracy from 12% to 25%, with 2.74 refinement rounds per query on average.

Additional capabilities. Table 6 also reports accurate retrieval (AR), test-time learning (TTL), and long-range understanding (LRU) on their full evaluation sets. MCC averages accuracy across five classification tasks. Recom. uses Recall@5, and Summ. uses fluency-gated F1. LME(S*) reports judge accuracy over 300 questions, distinct from the 500-question LongMemEval-S evaluation above. The remaining QA tasks use exact-match metrics.

Table 6: MemoryAgentBench results (%, GPT-4o-mini). Baseline scores are taken from MemoryAgentBench [11]. FC denotes FactConsolidation. Highest reported values among the listed methods are bold. †These tasks use the benchmark rubrics with different judging models for our results and the published baselines.

	AR				TTL		LRU		FC	
Method	SH-QA	MH-QA	LME(S*)†	EventQA	MCC	Recom.	Summ.†	DetQA	FC-SH	FC-MH
HippoRAG-v2	**76.0**	66.0	**50.7**	67.6	61.4	10.2	14.6	57.7	54.0	5.0
BM25	66.0	56.0	45.3	**74.6**	75.4	13.6	19.0	52.1	48.0	3.0
Long-context	64.0	43.0	30.7	59.0	**82.0**	15.1	28.9	63.4	45.0	5.0
MemoRAG	29.0	33.0	20.0	56.0	77.0	13.1	9.2	50.7	21.0	7.0
Contriever	22.0	31.0	15.7	66.8	70.6	15.2	17.2	42.3	18.0	7.0
MEMCORE	70.0	**71.0**	44.3	62.8	75.4	**15.8**	**51.0**	**74.6**	**59.0**	**12.0**

D.4 Statistical Analysis and Decoding Variability

Table 7 reports paired F1 differences and bootstrap confidence intervals on LoCoMo. All paired outputs are re-scored with the same F1 implementation, so the resulting differences may differ from those computed from the original main-table aggregates. Five answer-decoding runs give a standard deviation of 0.54 F1 points with unchanged rankings. Setting the answer-generation temperature to zero changes weighted F1 by -0.08 points relative to the default setting. These repeats characterize answer-decoding variability.

Table 7: Paired LoCoMo F1 gains (percentage points) and 95% bootstrap confidence intervals. Each row compares MEMCORE with the indicated baseline.

Backbone	Baseline	Paired gain	95% CI
Qwen3-8B	A-MEM	$+6.17$	$[4.45, 8.10]$
Qwen3-8B	LightMem	$+3.35$	$[1.41, 5.27]$
Qwen3-14B	A-MEM	$+5.46$	$[3.28, 7.50]$

D.5 LLM-as-Judge Evaluation

We complement lexical-overlap metrics with an LLM-as-judge evaluation of five memory systems on LoCoMo, using Qwen3-8B and 1,986 questions per method. All systems share the judging protocol. Table 8 reports the question-weighted score and the unweighted mean across categories. MEMCORE and LightMem have the same weighted score at the reported precision, with a small macro-average difference. These results indicate comparable performance under semantic judging in this setting.

Table 8: Supplementary LoCoMo judge scores (%, Qwen3-8B).

Method	Weighted	Macro
ReadAgent	37.2	30.9
MemoryBank	44.1	41.4
A-MEM	40.8	37.1
LightMem	54.2	50.1
MEMCORE	54.2	50.3

Table 9 extends the question-weighted judge comparison to all four backbones for MEMCORE, A-MEM, and LightMem. MEMCORE scores higher than A-MEM on all four backbones and higher than LightMem on three, with nearly identical scores on Qwen3-8B.

D.6 IER on A-MEM

We apply the same IER prompts and maximum of three iterations to A-MEM on all 1,986 LoCoMo questions with Qwen3-8B. The experiment reuses A-MEM's cached memories and preserves its

Table 9: Question-weighted LoCoMo judge accuracy (%) across four backbones. Bold marks the highest score among the three methods for each backbone.

Method	Qwen3-8B	Qwen3-14B	GPT-4o-mini	GPT-4o
A-MEM	40.79	40.68	42.70	46.30
LightMem	**54.23**	45.20	50.10	48.20
MEMCORE	54.18	**48.69**	**53.22**	**51.91**

retrieval and answer-generation code. The answerer receives at most ten notes at each step. Refinement changes their selection without increasing this budget. In the same evaluation harness, IER improves A-MEM from 28.86 to 31.88 F1, a paired gain of 3.03 points (95% CI $[1.88, 4.19]$). The 28.86 baseline is specific to this experiment and is distinct from the A-MEM run in Table 2. The corresponding gain in MEMCORE's own ablation is 5.68 points (32.94 to 38.62).

A further difference-in-differences analysis re-scores all four arms with one token-F1 implementation. On 1,974 aligned questions after duplicate collapse, the interaction is 3.12 points, with a 95% conversation-cluster bootstrap interval of $[1.64, 4.76]$ and a sign-flip permutation $p = 0.001$. This supports a larger IER gain in the MEMCORE configuration. A constant offset between evaluation harnesses cancels in this comparison, but an interaction between the harness and IER cannot be excluded.

D.7 Macro-Averaged F1 on LoCoMo

To complement the question-weighted averages in Tables 1 and 2, Table 10 reports the unweighted arithmetic mean of F1 across the five LoCoMo categories. Values are computed from the reported category scores and rounded to two decimal places. MEMCORE has the highest macro F1 on GPT-4o-mini, GPT-4o, and Qwen3-8B. On Qwen3-14B, LoCoMo ranks first and MEMCORE second. Thus, the ranking on Qwen3-14B depends on category weighting.

Table 10: Macro F1 (%) on LoCoMo, with equal weight for each of the five categories. Method order follows Tables 1 and 2. The best score per backbone is bold and the second best is underlined.

Method	GPT-4o-mini	GPT-4o	Qwen3-8B	Qwen3-14B
LOCOMO	33.01	33.55	27.87	**30.68**
READAGENT	9.31	9.38	22.66	16.45
MEMORYBANK	6.84	5.62	25.30	24.34
MEMGPT	29.13	31.00	27.24	24.98
A-MEM	<u>35.94</u>	34.83	28.20	23.84
MEM0	35.42	33.96	25.96	22.18
LIGHTMEM	34.90	<u>36.31</u>	<u>30.45</u>	27.16
MEMCORE	**39.78**	**39.75**	**33.59**	<u>28.60</u>

E Hyperparameter Settings

Table 11 summarizes selected hyperparameters used in MEMCORE. These hyperparameters are organized by the two main phases of our framework: memory construction and memory retrieval.

For memory construction, the merge threshold $\tau_m = 0.7$ and link threshold $\tau_l = 0.5$ guide the gated structural update decisions in the LLM prompt (Eq. (4)). A higher merge threshold is intended to ensure that only highly redundant information is consolidated, preserving fine-grained distinctions between memory units. The link threshold is set lower to capture complementary relationships that support multi-hop reasoning.

For memory retrieval, we set $K_{topic} = 3$ to balance navigation precision with coverage, allowing the system to explore multiple relevant topic clusters. The keyword retrieval parameter $K_{key} = 10$ provides sufficient anchor points for entity-centric queries. The maximum number of IER iterations, $I_{max} = 3$, prevents excessive retrieval loops while allowing sufficient evidence expansion for complex queries.

The values in Table 11 are fixed empirical settings. No held-out validation set was used to tune them.

Table 11: Hyperparameter settings.

Phase	Parameter	Value
Construction	Merge threshold τ_m	0.7
	Link threshold τ_l	0.5
Retrieval	Topic retrieval K_{topic}	3
	Keyword retrieval K_{key}	10
	Max IER iterations I_{max}	3
Generation	Temperature (general)	0.7
	Temperature (adversarial)	0.5

E.1 Threshold Sensitivity of Operation Assignments

We held LLM scores for 200 memory pairs fixed and varied the operation thresholds in code. Pairs are assigned to Merge for $s \geq \tau_m$, Link for $\tau_l \leq s < \tau_m$, and Append otherwise. Table 12 reports the resulting offline assignments. Identical counts at $\tau_m = 0.6$ and 0.7 reflect this sample's score distribution. The scoring prompt retains its numerical anchors. This check measures sensitivity of score-based assignments without rebuilding the memory graph or evaluating Recall@20 and F1.

Table 12: Offline operation assignments for 200 fixed LLM scores, varying one threshold at a time.

τ_m	τ_l	Append	Link	Merge
Varying τ_m with $\tau_l = 0.5$				
0.5	0.5	138	0	62
0.6	0.5	138	46	16
0.7	0.5	138	46	16
0.8	0.5	138	58	4
0.9	0.5	138	62	0
Varying τ_l with $\tau_m = 0.7$				
0.7	0.4	138	46	16
0.7	0.5	138	46	16
0.7	0.6	184	0	16

F Compute Resources

All experiments were conducted on a single server equipped with Intel Xeon Platinum 8480C CPUs and NVIDIA H800 80GB GPUs. Memory construction, retrieval, and generation for all backbone models and baselines were executed on this hardware. The primary computational cost arises from LLM-based semantic evaluation during memory construction (Section 2.2). Each observation requires one LLM call for semantic ingestion and at most one additional call for gated classification, which jointly evaluates up to five candidate entries. For closed-source models (GPT-4o, GPT-4o-mini), construction and generation costs are dominated by API latency rather than local compute.

F.1 Query Cost

Table 13 reports the supplementary query-stage cost profile. Tokens aggregate all query-stage LLM calls, including intermediate reasoning. Construction is excluded. IER increases the query budget in this setting. The approximately 6.6K-token final generation prompt is a separate quantity and should not be interpreted as total query consumption.

Table 13: Mean query-stage cost on LoCoMo in the supplementary cost evaluation. The IER setting permits at most three refinement iterations.

Configuration	LLM tokens/query	LLM calls/query	Latency (s/query)
Without IER	5,183	2.0	0.6
With IER ($I_{\max} = 3$)	27,568	7.4	9.9

G Broader Impact

This work improves memory systems for LLM-based agents, enabling more effective evidence retrieval from long-term interaction histories, which can benefit applications such as personal assistants and educational tools. However, enhanced long-term memory raises privacy concerns, as systems that consolidate detailed user interaction records may inadvertently expose sensitive information if not properly governed. The associative linking mechanism may also surface unintended connections between pieces of personal information. Additionally, the LLM-based construction components inherit biases from the underlying models, which could propagate through the memory hierarchy to downstream reasoning. We encourage practitioners to implement data governance policies, user-controlled forgetting mechanisms, and bias auditing when deploying such systems in practice.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The contributions are clearly outlined in both the abstract and the introduction.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We include the limitations in the main paper.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

Justification: This paper have no theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: Section 3.1 and Appendices C and E describe the experimental setup, prompts, and hyperparameter settings.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: The implementation and instructions for reproducing the experiments are available at `https://github.com/Liar-zzy/MemCoRe`.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Training and evaluation details are specified in the experiment section and appendix.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: Appendix D.4 reports bootstrap confidence intervals for paired F1 differences and variability across answer-decoding runs. Appendix D.6 reports confidence intervals and a permutation test for the IER interaction.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We list all the experiments in detail in the main paper and the appendix section.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conducted in the paper conforms to the NeurIPS Code of Ethics

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We include the broader impacts in the appendix.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A]

    Justification: This paper have no such risk.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We use the open source dataset.

    Guidelines:

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

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: The code is released under the MIT License at `https://github.com/Liar-zzy/MemCoRe`, with installation instructions, configuration files, and LoCoMo evaluation commands.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing or research with human subjects.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: This work does not involve human subjects.

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [Yes]

Justification: We describe all the LLM components in the main paper.

Guidelines:

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