

TSHMem: A Temporal-Semantic Hierarchical Memory Graph for Long-Term Conversational Agents

Feiyang Xu¹, Wenhao Yu¹, Xiang Li², Quanyu Dai², Irwin King¹

¹Department of Computer Science and Engineering, The Chinese University of Hong Kong

²Huawei Technologies Ltd.

{fyu25, whyu24, king}@cse.cuhk.edu.hk

{lixiang655, daiquanyu}@huawei.com

Abstract

Long-term conversational agents often face time-sensitive queries, requiring evidence that is both contextually relevant and valid for a specific time, event, or factual update. However, noisy temporal-entity cues, weak coupling between temporal anchors and retrieval paths, and sparse long-tail evidence often lead memory systems to retrieve semantically similar but temporally invalid or outdated evidence. We propose TSHMem, a Temporal-Semantic Hierarchical Memory Graph for long-term conversational agents. TSHMem addresses noisy conversational evidence by organizing dialogue turns, self-contained statements, hierarchical summaries, and entities into a heterogeneous memory graph. To connect temporal constraints with retrieval paths, it parses temporal intent, activates query-relevant entities, resolves event-referenced constraints, and injects temporal relevance into graph transition probabilities. It then combines time-aware structural propagation with semantic matching and hierarchical ranking to retrieve compact evidence that is temporally valid, contextually relevant, and robust to sparse long-tail memories. Experiments on LongMemEval S* and LoCoMo show that TSHMem consistently outperforms strong retrieval and memory-agent baselines, achieving 66.3% accuracy on LongMemEval S*, with an absolute gain of 18.6% over the strongest baseline while retrieving fewer evidence tokens.

1 Introduction

Long-term conversational agents require persistent memory to maintain coherent, personalized, and context-aware interactions across sessions. This need is closely related to personalized LLMs, where user-specific preferences, histories, and evolving contexts play a central role (Liu et al., 2025, 2026a). Although large language models (LLMs) exhibit strong in-context reasoning abilities, they remain largely stateless once the context

window is exhausted, making it difficult to recall user preferences, track factual updates, or reason over events distributed across conversations (Park et al., 2023; Packer et al., 2024). Retrieval-based memory offers a practical solution by storing dialogue histories as text chunks and retrieving them via sparse or dense similarity search (Lewis et al., 2020; Gao et al., 2024). However, flat chunk-based retrieval treats memories as isolated textual units, limiting its ability to capture cross-session dependencies, evolving facts, and temporal constraints.

A central challenge in long-term conversational memory is *time-sensitive retrieval*: an agent must retrieve not only semantically relevant memories, but also temporally valid and contextually updated evidence. Recent graph-based memory systems improve over flat retrieval by organizing past interactions into structured topologies, enabling traversal beyond direct semantic similarity (Edge et al., 2025; Gutiérrez et al., 2024). Nevertheless, as shown in Figure 1, existing memory graphs can still suffer from three temporal-semantic bottlenecks: (1) temporal and entity cues are often buried in noisy, redundant, and ambiguous dialogues, making it difficult to construct reliable memory units from raw conversation chunks; (2) temporal information is usually treated as metadata or decay scores, rather than being integrated into the graph topology that guides retrieval; and (3) time-sensitive evidence is often sparse and long-tailed, where rare entities, isolated events, or abrupt topic shifts can cause activation failures in semantic retrieval or naive graph traversal. As a result, memory systems may retrieve semantically similar but temporally invalid or outdated evidence, while missing the long-tail but crucial evidence needed to answer the query correctly.

To address these challenges, we propose **TSHMem**, a **Temporal-Semantic Hierarchical Memory Graph** for long-term conversational agents. TSHMem constructs a heterogeneous mem-

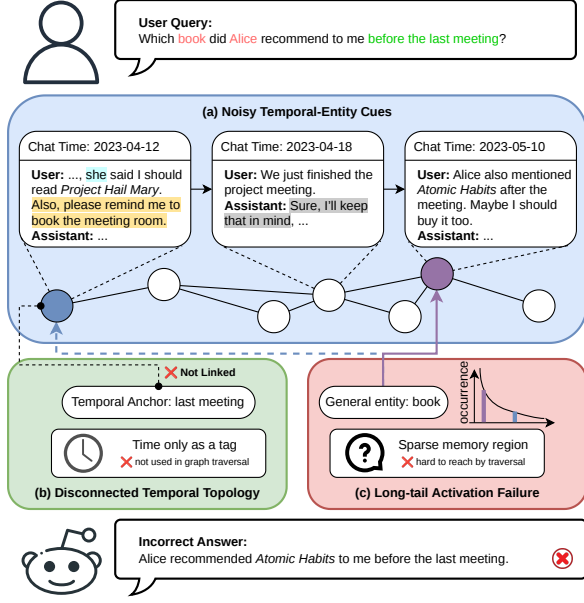


Figure 1: A representative failure case of time-sensitive conversational memory retrieval. The retrieval process selects *Atomic Habits*, which is semantically related but temporally invalid. This illustrates three bottlenecks: (a) noisy and ambiguous dialogues obscure reliable temporal and entity cues; (b) disconnected temporal traces weaken time-sensitive graph traversal; and (c) generic entities make sparse but crucial long-tail evidence hard to reach.

ory graph over dialogue turns, self-contained statements, hierarchical summaries, and entities. It performs query-aware temporal parsing and entity activation to resolve explicit time expressions and implicit event-referenced constraints, and incorporates temporal relevance into graph transition probabilities to guide structural propagation toward temporally valid memory paths. TSHMem further combines semantic matching, time-aware structural propagation, and hierarchical ranking to retrieve compact and relevant evidence from both directly matched and structurally related memory regions. Experiments on LongMemEval S* and LoCoMo show consistent improvements over strong retrieval and memory-agent baselines: TSHMem achieves 66.3% overall accuracy on LongMemEval S* and 68.4% on LoCoMo, with absolute accuracy gains of 18.6% and 8.1% over the strongest baselines, respectively.

Our contributions are summarized as follows: (1) We propose a temporal-semantic hierarchical memory graph that unifies dialogue-level, statement-level, and summary-level memory for long-term conversational agents, reducing conversational noise while preserving fine-grained grounding and

cross-session abstraction; (2) we introduce query-aware temporal parsing and entity activation, together with temporal transition reweighting, enabling retrieval along temporally valid memory paths; (3) we design a dual-track retrieval and hierarchical ranking mechanism that combines semantic relevance with time-aware structural propagation, improving robustness in sparse and long-tail memory regions; and (4) we design extensive experiments on LongMemEval S* and LoCoMo, demonstrating consistent performance with substantial gains in temporal reasoning, knowledge updates, and evidence-efficient retrieval.

2 Related Work

2.1 Long-term Memory for LLM Agents

Long-term memory enables LLM-based agents to maintain user preferences, plans, interaction histories, and evolving states across sessions. Early systems store observations, reflections, and plans for long-horizon behavior (Park et al., 2023), while MemGPT frames memory management as moving information between limited context windows and external storage (Packer et al., 2024). Recent work further explores scalable storage, memory consolidation, lifelong memory, and agentic memory evolution (Chhikara et al., 2025; Liu et al., 2026b; Xu et al., 2025; Fang et al., 2026). These methods highlight the importance of persistent memory, but often focus on storage, updating, or compression, with limited integration of memory granularity, temporal dynamics, and graph-based retrieval. In contrast, TSHMem constructs a temporal-semantic hierarchical memory graph for time-sensitive multi-session conversational retrieval.

2.2 Hierarchical and Graph-structured Retrieval

Retrieval-Augmented Generation retrieves external evidence to support LLM generation (Lewis et al., 2020; Gao et al., 2024). Standard conversational retrieval typically stores previous turns or sessions as chunks and retrieves them with lexical or dense similarity, but flat chunks struggle to capture cross-session dependencies, factual updates, and multi-level abstraction; longer contexts also do not guarantee effective evidence use (Liu et al., 2024). Hierarchical methods such as RAPTOR and MemoRAG build multi-level representations (Sarathi et al., 2024; Qian et al., 2025), while graph-based methods support global-local summa-

rization, associative retrieval, and complex reasoning (Edge et al., 2025; Gutiérrez et al., 2024; Wang et al., 2024). However, these methods are mainly designed for documents or knowledge-intensive retrieval, and reliable graph construction from noisy open-ended conversations remains challenging (Li et al., 2025). TSHMem addresses this by jointly modeling dialogue turns, statement-level facts, summaries, and entities in a heterogeneous conversational memory graph.

2.3 Temporal and Robust Conversational Memory

Temporal reasoning is essential for conversational memory because preferences, factual states, and intentions may evolve over time. Prior work on temporal knowledge graphs models evolving facts and event sequences (Jin et al., 2020; Lin et al., 2025), and recent memory systems such as Zep and MemoTime explore temporal graph structures for agent memory and LLM reasoning (Rasmussen et al., 2025; Tan et al., 2026). MemoryBank applies an Ebbinghaus-inspired exponential decay to model memory strength over elapsed storage time (Zhong et al., 2024). Nevertheless, many conversational memory systems still treat time as metadata or a final ranking feature, rather than incorporating temporal relevance into query-conditioned graph traversal. This limits their ability to retrieve evidence under precedence, recency, and update constraints. TSHMem instead injects temporal relevance into graph transitions and combines semantic matching, structural propagation, and hierarchical ranking to retrieve temporally valid and robust evidence, including sparse long-tail memories.

3 Methodology

We propose TSHMem, a unified temporal-semantic hierarchical memory framework designed to improve long-term conversational memory under noisy, time-sensitive, and long-tailed settings. As shown in Figure 2, our method consists of three main components: (1) we construct a heterogeneous memory graph over dialogues, self-contained statements, hierarchical summaries, and entities; (2) we perform query-aware temporal parsing and entity activation to obtain temporal constraints and seed nodes; and (3) we apply dual-track retrieval with time-aware structural propagation, semantic matching, and hierarchical ranking to produce hybrid evidence for final response generation.

3.1 Unified Memory Construction

We construct a heterogeneous memory graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ from long-term conversational histories, where $\mathcal{V}$ contains dialogue nodes, statement nodes, summary nodes, and entity nodes, and $\mathcal{E}$ contains trace-back, semantic, hierarchical, and entity-memory edges. This design preserves fine-grained conversational evidence while enabling higher-level abstraction and structure-aware retrieval.

Dialogue-to-statement extraction. Given a sliding window of dialogue turns, we use an LLM to extract self-contained statements from noisy conversational contexts. During extraction, coreference expressions are resolved, relative times are normalized, and each statement s_i is annotated with its temporal scope $t(s_i)$, event context $ev(s_i)$, and mentioned entities $E(s_i)$. Formally, the extraction process is defined as:

$$\{s_i, t(s_i), ev(s_i), E(s_i)\}_{i=1}^n = \text{LLM}_{\text{extract}}(\mathcal{D}), \quad (1)$$

where $\mathcal{D}$ denotes the input dialogue window.

Statement graph. Each extracted statement s_i becomes a statement node in the base graph. We add semantic edges based on the edge weight w_{ij} , defined as the cosine similarity between the embeddings $\phi(s_i)$ and $\phi(s_j)$. To maintain a sparse and efficient graph topology, an edge is created only if $w_{ij} \geq \theta_{act}$, and each node connects only to its top- W most similar neighbors.

Hierarchical summary graph. To support retrieval at higher abstraction levels, we build hierarchical summary nodes over the statement graph. We first identify overlapping statement communities using ego-network decomposition followed by label propagation, allowing a statement to participate in multiple thematic clusters (Li et al., 2025). For each community at hierarchy level l , an LLM generates a summary node at the next level, $l + 1$. Each summary node aggregates the semantic content, temporal scope, and entity states of its member nodes. Parent-child edges are added between a summary node and the statements or lower-level summaries it abstracts. The process is applied recursively until the number of communities converges or a maximum hierarchy depth L is reached.

Entity nodes and dialogue trace-back. We also create entity nodes from the extracted entity mentions. An entity node is connected to all statement and summary nodes in which the entity is

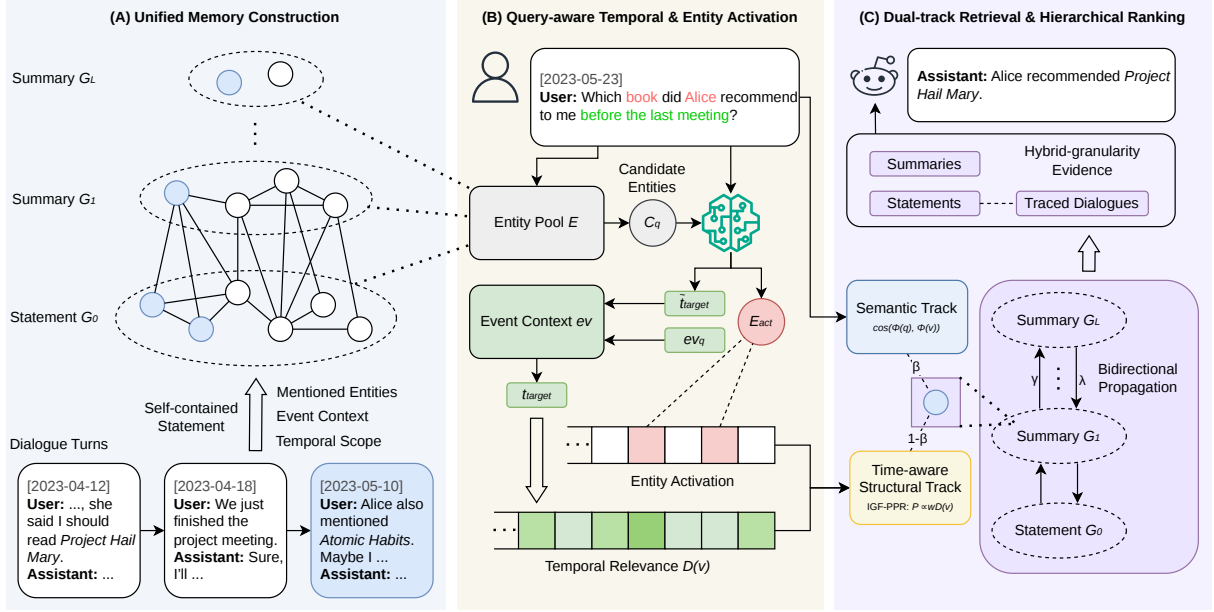


Figure 2: Overview of the proposed temporal-semantic hierarchical memory graph framework.

mentioned or aggregated. These entity-memory edges provide entry points for query-time activation and structural propagation. Each statement node is linked to its source dialogue node through trace-back edges, which are retained mainly for grounding.

The final graph integrates all node and edge types. This unified graph provides the structural basis for the further time-sensitive retrieval process.

3.2 Query-aware Temporal and Entity Activation

Given a user query q at timestamp t_q , the system needs to identify its temporal constraint and activate the corresponding entities in the memory graph. Since directly extracting entities from the query may produce overly general mentions (e.g., “book”), we first construct a candidate entity set by semantically matching the query against the global entity pool E :

$$C_q = \text{TopM}_{e \in E} \text{sim}(\phi(q), \phi(\text{profile}(e))), \quad (2)$$

where $\phi(\cdot)$ denotes the embedding function, and $\text{profile}(e)$ includes the name and associated contexts of entity e . We then obtain the query-side temporal metadata and selected entities as:

$$\tilde{t}_{\text{target}}, ev_q, E_q = \text{LLM}_{\text{query_metadata}}(q, t_q, C_q). \quad (3)$$

Here, $\tilde{t}_{\text{target}}$ contains the temporal relation r_q and any explicit temporal expression t_q^{exp} if available. ev_q denotes the query-side event description. E_q

denotes the entities selected from C_q . Since E_q is constrained by the existing entity pool, the activated entity set can be directly obtained as $E_{\text{act}} = E_q$, providing reliable seed nodes for structural retrieval.

When an explicit temporal expression t_q^{exp} is available, it is directly normalized into the target temporal scope t_{target} . For implicit event-referenced constraints (e.g., “before the last meeting”), we ground ev_q by similarity matching against extracted event contexts $\{ev(s_i)\}_{i=1}^N$, where N is the total number of extracted statements. If the top-1 event exceeds an upper threshold, it is used as the temporal anchor; otherwise, we retain up to five events above a lower threshold. The target scope t_{target} is then resolved from the matched anchor(s) according to the temporal relation r_q .

To couple temporal relevance with graph exploration, we define a dynamic time relevance score $D(v)$ for each node v in the memory graph. Given the resolved temporal target t_{target} and a node v with temporal scope $t(v)$, we set $\Delta t = 0$ if the two scopes overlap; otherwise, $\Delta t = \min_{x \in t_{\text{target}}, y \in t(v)} |x - y|$. We convert this target-conditioned distance into a temporal relevance score $D(v) = \exp(-\Delta t/\tau)$, using an exponential decay form also adopted by MemoryBank (Zhong et al., 2024), where τ is the decay hyperparameter. When no reliable or valid temporal target is identified, we set $D(v) = 1$ for all nodes to avoid incorrect temporal penalization. The activated enti-

ties E_{act} and time relevance scores $D(v)$ are used in the subsequent dual-track retrieval stage.

3.3 Dual-track Retrieval and Hierarchical Ranking

Given the activated entity set E_{act} and the time relevance $D(v)$, we retrieve memories through a dual-track mechanism that combines time-aware structural propagation with semantic matching.

Structural retrieval. We use Personalized PageRank (PPR) to propagate activation from E_{act} over semantic and entity-memory edges within each level-specific memory graph. Trace-back edges do not participate in PPR and are retained only for evidence grounding. To prevent overly general entities from dominating the propagation process, we scale the initial seed distribution with Inverse Graph Frequency (IGF). Let $n(e)$ denote the number of memory nodes associated with entity e , and let $|\mathcal{V}_m|$ be the number of memory nodes. The IGF score is defined as:

$$\text{IGF}(e) = \log \frac{|\mathcal{V}_m|}{n(e) + 1}. \quad (4)$$

The initial PPR seed distribution $\mathbf{r}$ is scaled by $\text{IGF}(e)$:

$$r(v) = \begin{cases} \frac{a(e)\text{IGF}(e)}{\sum_{e' \in E_{act}} a(e')\text{IGF}(e')}, & v = e, e \in E_{act}, \\ 0, & \text{otherwise,} \end{cases} \quad (5)$$

where $a(e)$ denotes the activation score assigned to the selected entity e during query-side entity activation.

To inject temporal guidance into graph traversal, we dynamically weight transition probabilities by the time relevance:

$$P(u \rightarrow v) = \frac{w_{uv}D(v)}{\sum_{k \in N(u)} w_{uk}D(k)}, \quad (6)$$

where w_{uv} is the propagation edge weight and $N(u)$ is the neighbor set of u . PPR is computed as:

$$\mathbf{p}^{(\ell+1)} = (1 - \alpha)\mathbf{r} + \alpha P^\top \mathbf{p}^{(\ell)}, \quad (7)$$

where α is the damping probability. After convergence, we obtain the structural score $S_{struct}(v) = \mathbf{p}(v)$ and min-max normalize it across memory nodes.

Semantic retrieval. In parallel, we compute the semantic similarity ($S_{sem}(v)$) between the query embedding $\phi(q)$ and the embedding of each memory node $\phi(v)$. The semantic scores are also min-max normalized.

The local relevance score is then computed by balancing structural and semantic evidence:

$$S_{local}(v) = (\beta S_{struct}(v) + (1 - \beta) S_{sem}(v)) \cdot D(v). \quad (8)$$

where β controls the trade-off between time-aware structural propagation and direct semantic matching.

Hierarchical ranking. The local score captures semantic relevance and intra-level structural connectivity, but it does not transfer relevance across hierarchical levels. We therefore apply bidirectional context propagation over the hierarchy. A node receives top-down relevance from its parent summaries and bottom-up relevance from its children:

$$S_{td}(v) = \max_{u \in \text{Parents}(v)} S_{local}(u), \quad (9)$$

$$S_{bu}(v) = \frac{1}{|\text{Children}(v)|} \sum_{u \in \text{Children}(v)} S_{local}(u). \quad (10)$$

When a node has no parent or child, the corresponding propagation score is set to 0. The global score is computed as:

$$S_{global}(v) = S_{local}(v) + \gamma S_{td}(v) + \lambda S_{bu}(v), \quad (11)$$

where γ and λ control the strength of top-down and bottom-up propagation.

Since nodes at different hierarchy levels may have different score distributions, we apply level-wise normalization:

$$\hat{S}_{global}(v) = \frac{S_{global}(v) - \mu_l}{\sigma_l + \epsilon}, \quad (12)$$

where node v belongs to level l , μ_l and σ_l are the mean and standard deviation of scores at that level, and ϵ is a small constant for numerical stability. Finally, we rank all memory nodes by $\hat{S}_{global}(v)$ and retrieve the top- K statements and summaries as hybrid evidence. For retrieved statement nodes, we further trace back their associated raw dialogue turns to provide faithful grounding for final response generation.

4 Experiments

We evaluate TSHMem on long-term conversational memory benchmarks through four questions: **RQ1** whether TSHMem outperforms standard retrieval, hierarchical or graph-based RAG, and memory-agent baselines; **RQ2** where the gains come from across multi-session, temporal reasoning, knowledge update, and other question types; **RQ3** how each major component contributes to performance; and **RQ4** whether it retrieves compact and useful evidence under a fixed token budget, especially in sparse or noisy memory regions. We organize the experiments accordingly: overall and fine-grained comparisons, component ablations, and evidence efficiency analysis.

4.1 Experimental Setup

4.1.1 Datasets

We evaluate TSHMem on LongMemEval and LoCoMo, two benchmarks for long-term conversational memory. For LongMemEval, we mainly use **LongMemEval S*** for main comparison (Wu et al., 2025; Hu et al., 2026). **LoCoMo** is used as an additional benchmark to evaluate generalization to long-form conversational QA (Maharana et al., 2024). Detailed dataset statistics are provided in Appendix A.

4.1.2 Baselines

We compare TSHMem with representative baselines from three categories.

Standard RAG. We include sparse and dense retrieval methods that retrieve relevant dialogue chunks before generation. Sparse retrieval is implemented with BM25 (Robertson and Zaragoza, 2009). Dense retrieval includes commonly used embedding-based retrievers, such as Contriever (Izacard et al., 2022), Text-Embedding-3-Large, and Qwen3-Embedding-4B (Zhang et al., 2025).

Hierarchical and Graph-based RAG. We compare with retrieval systems that introduce hierarchical abstraction or graph structures, including RAPTOR (Sarathi et al., 2024), GraphRAG (Edge et al., 2025), and HippoRAG-v2 (Gutiérrez et al., 2025).

Memory Agents. We further compare with memory-augmented agent systems, including Mem0 (Chhikara et al., 2025), Cognee (Markovic et al., 2025), Graphiti/Zep (Rasmussen et al., 2025), A-Mem (Xu et al., 2025), LightMem (Fang et al.,

2026), and SimpleMem (Liu et al., 2026b). Appendix E.1 additionally provides a controlled direct comparison with MIRIX (Wang and Chen, 2025).

4.1.3 Evaluation Metrics

We evaluate both answer quality and retrieval quality. For answer-level evaluation, we report LLM-as-Judge Accuracy (%) as the primary metric, using OAI-4o as the judge model, with the prompt provided in Appendix C.5. We use Accuracy as the primary metric because the answers may be semantically correct despite low lexical overlap. Token F1 is reported as a complementary lexical metric. A robustness analysis across judge models is provided in Appendix E.5.2. For retrieval quality, we conduct budgeted evidence evaluation under a 4096-token limit, reporting Recall@4K, Acc.@4K, and the average number of retrieved evidence tokens. Detailed definitions of the metrics are provided in Appendix D.

4.1.4 Implementation Details

Across all methods, we use OAI-4o-mini as the backbone LLM with the same evaluation protocol. For TSHMem, the main hyperparameters are $\theta_{act} = 0.6$, $W = 10$, $M = 30$, $\tau = 24 \times 3600$, $\alpha = 0.85$, $\beta = 0.1$, $\gamma = 0.1$, and $\lambda = 0.7$. More implementation details are provided in Appendix B.

4.2 Main Results

Overall Performance. The overall results are shown in Table 1. On LongMemEval S*, TSHMem achieves the best accuracy of 66.3% and the highest Token F1, outperforming the strongest baseline, LightMem, with an absolute accuracy gain of 18.6%. The results indicate that hybrid-granularity memory and temporal-semantic graph retrieval provide stronger long-term memory capability than flat retrieval, structured retrieval, and existing memory-agent systems. On LoCoMo, TSHMem achieves 68.4% accuracy, outperforming the strongest evaluated accuracy baseline, Contriever, with an absolute accuracy gain of 8.1%, although SimpleMem obtains higher Token F1. Representative cases in Appendix E.5.3 show that lexical overlap can underestimate semantically correct answers expressed in different surface forms. These results indicate that TSHMem improves semantic answer correctness across long-term conversational memory benchmarks.

Type	Method	LongMemEval S*		LoCoMo	
		Accuracy ↑	F1 ↑	Accuracy ↑	F1 ↑
Standard RAG	BM25	45.7	24.0	59.9	39.2
	Contriever	21.0	14.1	60.3	39.0
	Text-Embedding-3-Large	34.7	21.2	58.9	38.2
	Qwen3-Embedding-4B	43.0	24.1	59.0	38.5
Hierarchical/ Graph RAG	RAPTOR	33.0	26.4	31.3	27.7
	GraphRAG	34.3	24.7	24.1	21.3
	HippoRAG-v2	45.0	38.9	47.7	40.7
Memory Agents	Mem0	13.3	15.0	16.5	15.0
	Cognee	21.3	5.9	33.1	14.4
	Graphiti	16.7	16.8	23.6	12.8
	A-Mem	24.0	19.3	52.5	35.3
	LightMem	47.7	39.8	16.6	15.7
	SimpleMem	32.0	28.0	53.8	43.0
Ours	TSHMem	66.3	47.0	68.4	33.8

Table 1: Overall performance across all categories of LongMemEval S* and LoCoMo.

Fine-grained Performance. Table 2 shows that TSHMem performs best on the memory-intensive LongMemEval S* categories, including Multi-Session (MS), Temporal Reasoning (TR), and Knowledge Update (KU), reaching 46.7%, 69.3%, and 88.9%, respectively. The largest gain appears on TR, supporting the benefit of time-aware retrieval over evolving conversational histories. Table 3 shows similar trends on the four non-adversarial LoCoMo QA categories: TSHMem performs best on MultiHop and Temporal questions, while underperforming A-Mem on OpenDomain questions. These results suggest that TSHMem is particularly effective for memory-intensive and time-sensitive reasoning, with smaller gains on open-domain questions. Appendix E.2.2 examines the OpenDomain results in relation to evidence coverage and conservative answer generation, while Appendix E.2.3 separately analyzes the sensitivity of Adversarial results to distractor exposure.

4.3 Ablation Study

Effect of Hybrid-granularity Memory. Table 4 evaluates the contribution of different memory granularities in the retrieved evidence. Removing summaries causes the largest drop, reducing Overall accuracy from 66.3% to 57.7% and TR from 69.3% to 49.3%, suggesting that summary-level evidence helps aggregate scattered cross-session information. Removing statements or dialogues also degrades performance, showing that fine-grained factual evidence and source-grounded dialogue context are complementary. Overall, these results confirm that combining multiple memory granular-

Method	SU	SA	SP	MS	TR	KU
<i>Standard RAG</i>						
BM25	91.1	86.7	20.0	24.0	24.0	62.2
Qwen3-Embedding-4B	77.8	83.3	20.0	26.7	20.0	62.2
<i>Hierarchical/Graph RAG</i>						
HippoRAG-v2	77.8	73.3	26.7	22.7	30.7	66.7
<i>Memory Agents</i>						
A-Mem	35.6	40.0	16.7	12.0	22.7	28.9
LightMem	82.2	16.7	20.0	42.7	37.3	77.8
SimpleMem	68.9	43.3	10.0	16.0	18.7	43.3
<i>Ours</i>						
TSHMem	91.1	83.3	20.0	46.7	69.3	88.9

Table 2: Fine-grained Accuracy (↑) on LongMemEval S*. SU, SA, SP, MS, TR, and KU denote Single-Session User, Single-Session Assistant, Single-Session Preference, Multi-Session, Temporal Reasoning, and Knowledge Update, respectively.

ities is crucial for balancing abstract cross-session reasoning with fine-grained factual grounding.

Effect of Temporal Retrieval Components. Table 5 shows that both temporal components contribute to the effectiveness of TSHMem. Removing the time-aware edge weight reduces the overall accuracy from 66.3% to 63.7%, with clear drops on both TR and KU questions. This absolute gain of 2.6% confirms the effectiveness of transition-level temporal guidance in steering graph propagation toward temporally relevant memory paths. Removing time decay in local relevance score further reduces accuracy to 63.3%, and leads to a more substantial drop on KU questions, from 88.9% to 80.0%. This suggests that time decay is particularly useful for distinguishing updated memories from outdated evidence. Overall, these results confirm that our

Method	SH	MH	Temp.	OD
<i>Standard RAG</i>				
BM25	79.8	40.4	41.7	46.9
Qwen3-Embedding-4B	80.3	37.9	39.3	50.0
<i>Hierarchical/Graph RAG</i>				
HippoRAG-v2	66.0	34.4	31.5	41.7
<i>Memory Agents</i>				
A-Mem	70.6	35.5	29.0	51.0
LightMem	18.2	13.8	10.3	32.3
SimpleMem	70.3	36.9	55.1	43.8
<i>Ours</i>				
TSHMem	71.3	41.8	66.7	34.4

Table 3: Fine-grained Accuracy ($\uparrow$) on the four non-adversarial LoCoMo QA categories. SH, MH, Temp., and OD denote SingleHop, MultiHop, Temporal, and OpenDomain, respectively.

Variant	Overall	MS	TR	Δ
TSHMem	66.3	46.7	69.3	–
w/o statements	59.7	36.0	57.3	-6.6
w/o summaries	57.7	37.3	49.3	-8.6
w/o dialogues	59.3	41.3	64.0	-7.0

Table 4: Ablation on noise-aware hybrid memory. All values are Accuracy ($\uparrow$) on LongMemEval S*.

temporal retrieval components are important for time-sensitive memory retrieval.

Effect of Long-tail-aware Structural Propagation. Table 6 shows that removing IGF or level normalization reduces Overall accuracy to 63.3% and 58.0%, respectively. The large drop without level normalization suggests that score calibration across hierarchy levels is important, while the IGF ablation shows that down-weighting overly frequent entities improves robust activation in long-tail memory regions.

4.4 Evidence Efficiency Analysis

Beyond final answer accuracy, we evaluate evidence utility under a fixed 4096-token context budget. For each method, we truncate retrieved evidence to 4096 tokens, map it to turn-level evidence, and compute Recall@4K against gold supporting turns. We also generate answers using only the budgeted evidence to compute Acc.@4K. All retrieval metrics are macro-averaged over queries.

Budgeted Evidence Efficiency. As shown in Table 7, chunk-based baselines nearly exhaust the 4096-token budget but cover only about 32% of the gold supporting turns. In contrast, TSHMem achieves 64.7% Recall@4K with only 1,084 tokens on average, and achieves an Acc.@4K of 66.3%.

Variant	Overall	TR	KU	Δ
TSHMem	66.3	69.3	88.9	–
w/o time-aware weight	63.7	66.7	84.4	-2.6
w/o time decay	63.3	66.7	80.0	-3.0

Table 5: Ablation on temporal retrieval components. All values are Accuracy ($\uparrow$) on LongMemEval S*.

Variant	Overall	MS	KU	Δ
TSHMem	66.3	46.7	88.9	–
w/o IGF	63.3	44.0	82.2	-3.0
w/o level norm.	58.0	42.7	75.6	-8.3

Table 6: Ablation on long-tail-aware structural propagation. All values are Accuracy ($\uparrow$) on LongMemEval S*.

This suggests that TSHMem improves evidence utility not by retrieving more context, but by selecting relevant memories more effectively.

Retrieval under Evidence Sparsity. Table 8 evaluates Recall@4K across different sparsity levels of golden evidence, following the grouping protocol in Appendix D.2. TSHMem achieves the highest recall across all groups, with gains in Sparse regions showing better recovery of weakly connected long-tail evidence and larger gains in Medium and Dense regions showing stronger use of richer graph connectivity. The lower absolute recall in Sparse regions indicates that retrieving weakly connected evidence remains challenging.

5 Conclusion

We propose TSHMem, a Temporal-Semantic Hierarchical Memory Graph for long-term conversational agents. TSHMem organizes dialogue turns, self-contained statements, hierarchical summaries, and entities into a heterogeneous memory graph, reducing contextual noise while preserving multi-granularity evidence. It incorporates query-specific temporal constraints into retrieval by temporal parsing, entity activation, and time-aware graph transitions. It further combines structural propagation, semantic matching, and hierarchical ranking to retrieve compact and temporally valid evidence. Experiments on LongMemEval S* and LoCoMo show consistent improvements over strong retrieval and memory-agent baselines, especially on temporal reasoning, knowledge updates, and multi-session memory questions. Ablation and evidence efficiency analyses confirm the effectiveness of the main components and show that TSHMem improves evidence recall while using fewer evidence

Method	Recall@4K $\uparrow$	Acc.@4K $\uparrow$	Tokens $\downarrow$
BM25	32.6	30.7	4079
HippoRAG-v2	32.0	31.3	4091
A-Mem	7.0	17.7	4093
TSHMem	64.7	66.3	1084

Table 7: Budgeted evidence utility under a 4096-token evidence budget. Recall@4K measures gold supporting-turn coverage within the budget, and Acc.@4K measures answer accuracy using only the budgeted evidence.

Method	Sparse	Medium	Dense
BM25	9.7	43.8	34.5
HippoRAG-v2	16.2	41.2	33.1
A-Mem	0.6	10.7	7.5
TSHMem	27.6	58.6	68.5

Table 8: Recall@4K ($\uparrow$) under different evidence-entity density levels. Sparse, Medium, and Dense groups are defined according to the average graph degree of evidence-related entities.

tokens. Future work will explore stronger preference modeling and more efficient retrieval for long-term conversational memory.

Limitations

Despite its effectiveness, TSHMem has several limitations.

Preference modeling. TSHMem is better suited to temporal event sequences and explicit factual updates than to implicit preference evolution. It represents user-related information through statements, summaries, and entity-linked memories, but it does not explicitly model preference formation, persistence, or change. This limits its ability to distinguish stable preferences from transient interests or context-specific choices. Since preferences are often implicit, sparse, and distributed across sessions, future work may incorporate preference-aware memory states or update mechanisms to better capture how user preferences evolve over time. This limitation also affects stale-memory handling: when user preferences or personal facts change implicitly, TSHMem may not fully distinguish stable memories, even though temporal relevance reduces temporally mismatched retrieval.

Weakly connected long-tail evidence. Although TSHMem improves retrieval in sparse memory regions, weakly connected evidence remains difficult to recover when relevant entities appear rarely or

are only indirectly linked to the query. In such cases, entity activation and structural propagation may still fail to reach the necessary evidence. Future work may explore stronger long-tail activation, query expansion, or adaptive graph augmentation to improve retrieval for weakly connected memories.

Retrieval efficiency and scalability. The structural retrieval module relies on Personalized PageRank over the memory graph. While this enables activation to reach structurally related evidence beyond direct semantic matches, iterative propagation may become increasingly expensive as conversational histories grow. Future work may improve scalability from two directions: using approximate graph propagation, and maintaining a compact active memory graph through archival strategies.

Backbone model dependence. TSHMem relies on an LLM backbone for statement extraction and query-time temporal and entity parsing, so model errors can propagate to graph construction and retrieval. This dependence may affect robustness and deployment cost, particularly when implicit temporal constraints are difficult to resolve. Future work may combine smaller backbone models with rule-based temporal normalization or hybrid parsing strategies for more robust and lightweight deployment.

Ethical Considerations

Our experiments use only publicly released, synthetic research benchmarks and do not involve private user conversations. Nevertheless, long-term conversational memory systems may retain sensitive personal information, outdated facts, implicit preferences, and temporally structured interaction histories. If such memories are surfaced in real applications, they could lead to incorrect personalization, inappropriate use of past user information, or exposure of sensitive memories. Real-world deployment should therefore obtain informed user consent, minimize data collection, enforce appropriate access controls and retention policies, and provide users with mechanisms to inspect, delete, and correct stored memories. Systems should also make clear distinctions between temporary conversational context and information that users explicitly choose to retain persistently, and should support appropriate auditing of how stored memories are accessed and used. Such safeguards are

important for responsible deployment and cannot be addressed solely through improvements in memory retrieval accuracy.

Acknowledgments

The research presented in this paper was partially supported by the Research Grants Council of the Hong Kong Special Administrative Region, China (CUHK 2300246, RGC C1043-24G), (CUHK 14203425, RGC GRF 2151317), and CUHK 7010870.

References

- Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. 2025. [Mem0: Building production-ready AI agents with scalable long-term memory](#). In *ECAI 2025 - 28th European Conference on Artificial Intelligence, 25-30 October 2025, Bologna, Italy - Including 14th Conference on Prestigious Applications of Intelligent Systems (PAIS 2025)*, volume 413 of *Frontiers in Artificial Intelligence and Applications*, pages 2993–3000. IOS Press.
- Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2025. [From local to global: A graph rag approach to query-focused summarization](#). *Preprint*, arXiv:2404.16130.
- Jizhan Fang, Xinle Deng, Haoming Xu, Ziyang Jiang, Yuqi Tang, Ziwen Xu, Shumin Deng, Yunzhi Yao, Mengru Wang, Shuofei Qiao, Huajun Chen, and Ningyu Zhang. 2026. [Lightmem: Lightweight and efficient memory-augmented generation](#). In *International Conference on Learning Representations*, volume 2026, pages 98706–98729.
- Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. [Retrieval-augmented generation for large language models: A survey](#). *Preprint*, arXiv:2312.10997.
- Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. [Hipporag: Neurobiologically inspired long-term memory for large language models](#). In *Advances in Neural Information Processing Systems*, volume 37, pages 59532–59569. Curran Associates, Inc.
- Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From rag to memory: non-parametric continual learning for large language models. In *Proceedings of the 42nd International Conference on Machine Learning, ICML’25*. JMLR.org.
- Yuanzhe Hu, Yu Wang, and Julian McAuley. 2026. [Evaluating memory in llm agents via incremental multi-turn interactions](#). In *International Conference on Learning Representations*, volume 2026, pages 156259–156291.
- Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. [Unsupervised dense information retrieval with contrastive learning](#). *Preprint*, arXiv:2112.09118.
- Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. 2020. [Recurrent event network: Autoregressive structure inference over temporal knowledge graphs](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6669–6683, Online. Association for Computational Linguistics.
- Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. [Retrieval-augmented generation for knowledge-intensive nlp tasks](#). In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.
- Rui Li, Zeyu Zhang, Xiaohe Bo, Zihang Tian, Xu Chen, Quanyu Dai, Zhenhua Dong, and Ruiming Tang. 2025. [Cam: A constructivist view of agentic memory for llm-based reading comprehension](#). In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pages 113381–113406. Curran Associates, Inc.
- Junhong Lin, Song Wang, Xiaojie Guo, Julian Shun, and Yada Zhu. 2025. [Temporal reasoning with large language models augmented by evolving knowledge graphs](#). *Preprint*, arXiv:2509.15464.
- Jiahong Liu, Zexuan Qiu, Zhongyang Li, Quanyu Dai, Wenhao Yu, Jieming Zhu, Minda Hu, Menglin Yang, Tat-Seng Chua, and Irwin King. 2025. [A survey of personalized large language models: Progress and future directions](#). *Preprint*, arXiv:2502.11528.
- Jiahong Liu, Wenhao Yu, Quanyu Dai, Zhongyang Li, Jieming Zhu, Menglin Yang, Tat-Seng Chua, and Irwin King. 2026a. Perfit: Exploring personalization shifts in representation space of llms. In *The Fourteenth International Conference on Learning Representations*.
- Jiaqi Liu, Yaofeng Su, Peng Xia, Siwei Han, Zeyu Zheng, Cihang Xie, Mingyu Ding, and Huaxiu Yao. 2026b. [Simplemem: Efficient lifelong memory for llm agents](#). *Preprint*, arXiv:2601.02553.
- Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. [Lost in the middle: How language models use long contexts](#). *Transactions of the Association for Computational Linguistics*, 12:157–173.
- Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang.

2024. [Evaluating very long-term conversational memory of LLM agents](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13851–13870, Bangkok, Thailand. Association for Computational Linguistics.
- Vasilije Markovic, Lazar Obradovic, Laszlo Hajdu, and Jovan Pavlovic. 2025. [Optimizing the interface between knowledge graphs and llms for complex reasoning](#). *Preprint*, arXiv:2505.24478.
- Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. 2024. [Memgpt: Towards llms as operating systems](#). *Preprint*, arXiv:2310.08560.
- Joon Sung Park, Joseph O’Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. [Generative agents: Interactive simulacra of human behavior](#). In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, UIST ’23, New York, NY, USA. Association for Computing Machinery.
- Hongjin Qian, Zheng Liu, Peitian Zhang, Kelong Mao, Defu Lian, Zhicheng Dou, and Tiejun Huang. 2025. [Memorag: Boosting long context processing with global memory-enhanced retrieval augmentation](#). In *Proceedings of the ACM on Web Conference 2025*, WWW ’25, page 2366–2377, New York, NY, USA. Association for Computing Machinery.
- Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. 2025. [Zep: A temporal knowledge graph architecture for agent memory](#). *Preprint*, arXiv:2501.13956.
- Stephen Robertson and Hugo Zaragoza. 2009. [The probabilistic relevance framework: Bm25 and beyond](#). *Foundations and Trends in Information Retrieval*, 4(1-2):1–174.
- Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher Manning. 2024. [Raptor: Recursive abstractive processing for tree-organized retrieval](#). In *International Conference on Learning Representations*, volume 2024, pages 32628–32649.
- Xingyu Tan, Xiaoyang Wang, Qing Liu, Xiwei Xu, Xin Yuan, Liming Zhu, and Wenjie Zhang. 2026. [Memotime: Memory-augmented temporal knowledge graph enhanced large language model reasoning](#). In *Proceedings of the ACM Web Conference 2026*, WWW ’26, page 4220–4231, New York, NY, USA. Association for Computing Machinery.
- Yu Wang and Xi Chen. 2025. [Mirix: Multi-agent memory system for llm-based agents](#). *Preprint*, arXiv:2507.07957.
- Yu Wang, Nedim Lipka, Ryan A. Rossi, Alexa Siu, Ruiyi Zhang, and Tyler Derr. 2024. [Knowledge graph prompting for multi-document question answering](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17):19206–19214.
- Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. 2025. [Longmemeval: Benchmarking chat assistants on long-term interactive memory](#). In *International Conference on Learning Representations*, volume 2025, pages 86809–86836.
- Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. 2025. [A-mem: Agentic memory for llm agents](#). In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pages 17577–17604. Curran Associates, Inc.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. [Qwen3 technical report](#). *Preprint*, arXiv:2505.09388.
- Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025. [Qwen3 embedding: Advancing text embedding and reranking through foundation models](#). *Preprint*, arXiv:2506.05176.
- Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2024. [Memorybank: Enhancing large language models with long-term memory](#). In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 19724–19731.

A Dataset Details

Table 9 summarizes the dataset statistics after our preprocessing. We evaluate TSHMem on two main benchmarks, LongMemEval S* and LoCoMo. LongMemEval Oracle is used only for supplementary analysis in the appendix.

All datasets used in this work are publicly released research benchmarks for long-term conversational memory and question answering. The third-party datasets retain their original licenses and terms of use: LoCoMo is released under CC BY-NC 4.0, while LongMemEval settings are released under the MIT license. We use these datasets only for academic evaluation, consistent with their research-oriented benchmark purposes. The benchmark conversations are synthetic rather than collected from real users (Maharana et al., 2024; Wu et al., 2025). We manually inspected the data to check for obvious personally identifying information and offensive content.

LongMemEval S*. LongMemEval S* is a LongMemEval-derived setting introduced by MemoryAgentBench (Hu et al., 2026). It chronologically arranges and concatenates multiple historical

Dataset	#Convs	#Queries	Avg. Turns	Avg. Tokens
LME S*	5	300	1,288.0	343,603.0
LME Oracle	500	500	21.9	5,804.7
LoCoMo	10	1,986	588.2	15,970.5

Table 9: Dataset statistics after our preprocessing. LME denotes LongMemEval. LongMemEval S* and LoCoMo are used for the main evaluation, while LongMemEval Oracle is used only for supplementary diagnostic analysis in the appendix. For LoCoMo, we use the publicly released `locomo10.json` subset.

conversation segments from LongMemEval (Wu et al., 2025) into five long conversation histories. This construction increases the number of questions evaluated per memory construction and reduces the need to rebuild memory for each query. After our preprocessing, LongMemEval S* contains 5 long conversation histories and 300 QA pairs, with an average of 1,288.0 turns and 343,603.0 tokens per sample. We use it as the primary benchmark for evaluating long-term conversational memory because it combines long histories, multi-session evidence, and time-sensitive question types. Following the benchmark setting, we report results on six categories: Single-Session User (SU), Single-Session Assistant (SA), Single-Session Preference (SP), Multi-Session (MS), Temporal Reasoning (TR), and Knowledge Update (KU).

LongMemEval Oracle. LongMemEval Oracle comes from the original LongMemEval benchmark (Wu et al., 2025). It contains 500 samples and 500 QA pairs, with an average of 21.9 turns and 5,804.7 tokens per sample. Since this oracle setting does not require retrieval from long full-history contexts, it is not directly comparable to the main LongMemEval S* setting. Therefore, we use LongMemEval Oracle only for supplementary analysis in Appendix E, rather than as part of the main comparison.

LoCoMo. LoCoMo is a very long-term conversational memory benchmark proposed to evaluate models over extended multi-turn interactions (Maharana et al., 2024). In our experiments, we focus on the QA task and use the publicly released `locomo10.json` as the benchmark data. After our preprocessing, this subset contains 10 conversation samples and 1,986 QA pairs, with an average of 588.2 turns and 15,970.5 tokens per sample. We evaluate five QA categories: SingleHop, MultiHop, Temporal, OpenDomain, and Adversarial. Com-

pared with LongMemEval S*, LoCoMo provides a complementary testbed with more natural long-form conversations and different question distributions, allowing us to examine whether TSHMem generalizes beyond the LongMemEval setting.

B Implementation Details

This section describes the implementation choices used to reproduce the experiments. We first specify the common generation and baseline settings used for fair comparison, then describe the implementation of TSHMem, including model calls, graph construction, package implementations, and retrieval hyperparameters. Table 10 summarizes the main hyperparameters of TSHMem, and Algorithms 1 and 2 summarize the two implementation components for unified memory graph construction and query preprocessing.

For fair comparison, all methods use OAI-4o-mini as the LLM backbone, with temperature 0.0 and a maximum output length of 512 tokens. Here, OAI-4o-mini, OAI-4o, and GDM-3.1-Pro-Preview denote the API model identifiers used in our experiments. For baseline methods, we use the default configurations from their released implementations unless otherwise specified.

For TSHMem, we use API-based models for generation and embedding, namely OAI-4o-mini and Text-Embedding-3-Large; their exact parameter counts are not publicly disclosed. We do not train or fine-tune any neural model, and all experiments are inference-only. The code is executed on a server with 16 vCPUs of Intel(R) Xeon(R) Platinum 8352V CPU @ 2.10GHz, while LLM and embedding calls are served through external APIs.

The memory graph in TSHMem is implemented with NetworkX, where nodes represent dialogue turns, statements, summaries, and entities, and edges represent trace-back, semantic, hierarchical, and entity-memory relations. Personalized PageRank is computed using the NetworkX implementation with damping probability $\alpha = 0.85$. The remaining retrieval and generation steps directly follow the scoring functions described in Section 3.

B.1 Construction and Indexing Cost

Let J denote the number of dialogue windows, and let $N = \sum_{j=1}^J n_j$ be the total number of extracted statements. Let d denote the embedding dimension and W the maximum number of semantic neighbors retained per statement. For each constructed

Hyperparameter	Value
Dialogue window size	2
Semantic edge threshold θ_{act}	0.6
Top semantic neighbors W	10
Candidate entities M	30
Temporal decay τ	24×3600
Event upper threshold θ_{upper}	0.85
Event lower threshold θ_{lower}	0.6
PPR damping probability α	0.85
Structural-semantic weight β	0.1
Top-down hierarchy weight γ	0.1
Bottom-up hierarchy weight λ	0.7
Maximum hierarchy depth L	10
Retrieved evidence K	10
Temperature	0.0
Maximum output tokens	512

Table 10: Main implementation hyperparameters of TSHMem.

Algorithm 1 Unified Memory Graph Construction

Require: Conversation history $\mathcal{H}$
Ensure: Memory graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$

- 1: $\mathcal{G} = (\mathcal{V}, \mathcal{E}) \leftarrow (\emptyset, \emptyset)$
- 2: $\{\mathcal{D}_j\}_{j=1}^J \leftarrow \text{WINDOW}(\mathcal{H}, 2)$
- 3: **for** $j = 1, \dots, J$ **do**
- 4: $\{s_i, t(s_i), ev(s_i), E(s_i)\}_{i=1}^{n_j} \leftarrow \text{LLM}_{extract}(\mathcal{D}_j)$
- 5: $\mathcal{V} \leftarrow \mathcal{V} \cup \{s_i\}_{i=1}^{n_j}$
- 6: $\text{ADDTRACEBACKEDGES}(\{s_i\}_{i=1}^{n_j}, \mathcal{D}_j)$
- 7: **end for**
- 8: $\text{ADDSemanticEdges}(\mathcal{G}, \{s_i\}, \phi, \theta_{act}, W)$
- 9: $\text{ADDENTITYNodes}(\mathcal{G}, \{E(s_i)\})$
- 10: $\text{BUILDHierarchicalSummaries}(\mathcal{G}, L)$
- 11: **return** $\mathcal{G}$

summary level $l \in \{1, \dots, L\}$, where hierarchy construction may stop early, let C_l denote the number of summary nodes and E_l the number of graph edges processed when constructing and indexing that level, including its parent-child edges. Under the implementation used in our experiments, statement-level semantic links are built with approximate nearest-neighbor top- W retrieval. The CPU-side indexing complexity is:

$$O\left(NWd + \sum_{l=1}^L (C_l + E_l)\right), \quad (13)$$

where the first term covers statement-level semantic linking and the second covers hierarchical graph construction and indexing. The LLM-call complexity of statement extraction, parsing, and summary generation is:

$$O\left(J + N + \sum_{l=1}^L C_l\right). \quad (14)$$

Construction cost therefore grows primarily with the number of extracted statements and summary

Algorithm 2 Query-aware Temporal and Entity Activation

Require: Query q , query timestamp t_q , entity pool E , memory graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
Ensure: Activated entities E_{act} , time relevance $\{D(v)\}_{v \in \mathcal{V}}$

- 1: $C_q \leftarrow \text{TOPM}_{e \in E} \text{sim}(\phi(q), \phi(\text{profile}(e)))$
- 2: $(\tilde{t}_{target}, ev_q, E_q) \leftarrow \text{LLM}_{query_metadata}(q, t_q, C_q)$
- 3: $E_{act} \leftarrow E_q$
- 4: **if** $\tilde{t}_{target}$ contains explicit time expression t_q^{exp} **then**
- 5: $t_{target} \leftarrow \text{NORMALIZETIME}(t_q^{\text{exp}})$
- 6: **else if** ev_q can be matched to $\{ev(s_i)\}_{i=1}^N$ **then**
- 7: $t_{target} \leftarrow \text{RESOLVEEVENTTIME}(ev_q, r_q, \mathcal{G})$
- 8: **else**
- 9: $t_{target} \leftarrow \emptyset$
- 10: **end if**
- 11: **for** $v \in \mathcal{V}$ **do**
- 12: **if** $t_{target} = \emptyset$ or $t(v)$ is unavailable **then**
- 13: $D(v) \leftarrow 1$
- 14: **else**
- 15: $D(v) \leftarrow \exp(-\Delta t(v, t_{target})/\tau)$
- 16: **end if**
- 17: **end for**
- 18: **return** $E_{act}, \{D(v)\}_{v \in \mathcal{V}}$

Method	Time (s) ↓
GraphRAG	1354.8
TSHMem	1715.3
HippoRAG-v2	1918.0

Table 11: Average construction/indexing time on Long-MemEval S*. Lower is better.

nodes.

Table 11 shows that TSHMem costs more to index than GraphRAG but less than HippoRAG-v2. This is an offline cost incurred once per conversation history and can be amortized over subsequent queries. Controlling the number of summary nodes, using approximate graph construction, and archiving inactive memories are promising ways to reduce this cost.

C Prompt Templates

C.1 Statement Extraction Prompt

We implement statement extraction in two steps. First, the LLM rewrites a dialogue window into independent and unambiguous factual statements. Second, each statement is parsed into a structured memory entry with temporal, event, and entity metadata.

Statement Extraction Prompt

Your task is to rewrite the following dialogue into a list of independent, unambiguous factual statements.

{context}

[Rules]

1. Replace ALL pronouns (he, she, it, they, this, that) with the actual names or specific nouns.
2. Resolve ALL relative times (yesterday, today, tomorrow, later) into absolute times (YYYY-MM-DDTHH:MM:SS) based on the dialogue timestamps.
3. Write one complete sentence per atomic fact.

[Current Window Dialogues]
{dialogue_text}

[Output Format]

Return a JSON array of unambiguous statements:

```
[
  "Unambiguous statement 1.",
  "Unambiguous statement 2.",
  ...
]
```

Now process the above dialogues. Return ONLY the JSON array, no other explanations.

Metadata Extraction Prompt

Based on the following unambiguous statement, extract the information into a structured memory format.

[Current Time]
{current_time}

[Unambiguous Statement]
{unambiguous_statement}

[Extraction Rules]

1. Map the statement to an entry.
2. Extract 'topic', 'time_info', 'event_info', and 'contextual_keywords' carefully.
3. CRITICAL: Time information goes ONLY in 'time_info' or 'event_time_info'. Never put times/dates in 'contextual_keywords'.

{format_instructions}

C.2 Summary Generation Prompt

This prompt is used to generate hierarchical summary nodes over statement or summary communities. Given a cluster of related memory entries, the LLM produces a higher-level summary together with structured metadata. The prompt also includes an external in-context example following the same output schema.

Summary Generation Prompt

You are an expert summarization agent. Given a cluster of related memory entries (each entry is either a statement-level memory entry or a higher-level summary entry), produce one concise summary plus structured metadata.

[Input]

- Below is the list of entries to summarize. Each entry is a short passage of factual text. The list may mix statements and summaries.

[Entries]
{input_texts}

[Requirements]

1. **Higher-Level Synthesis**: Do not merely concatenate the inputs. Abstract and synthesize the underlying narrative, pattern, or overarching event. The `abstracted_summary` must be a self-contained, completely understandable paragraph that captures the core essence of the cluster while preserving critical factual details and causal links.

2. **Force Disambiguation**: Absolutely PROHIBIT using pronouns (he, she, they, this, it) or relative time (yesterday, later, beforehand). Resolve all references to explicit entity names and absolute timestamps.
3. **Topic Generalization**: Provide a concise `topic` phrase that encapsulates the entire cluster's broader theme.
4. **Temporal Context Aggregation (CRITICAL)**: Determine the overarching time context for this cluster using the `time_info` object. You MUST strictly adhere to the following conditional logic. NEVER output a "relation" without its corresponding time value(s).
 - **Condition A (Specific Point in Time)**: If the event is anchored to a specific, determinable absolute time.
 - Set `relation` to `"now"`, `"before"`, or `"after"`.
 - **Required Field**: You MUST output the `"time"` field (ISO 8601).
 - **Guidance**: Prioritize `"now"` if the event happens exactly ON a known date/time. Only use `"before"` or `"after"` if the text implies a sequence relative to an absolute time anchor without providing the exact event time.
 - **Condition B (Duration/Span)**: If the event spans a continuous, bounded period.
 - Set `relation` to `"span"`.
 - **Required Fields**: You MUST output both `"start_time"` and `"end_time"` fields (ISO 8601). DO NOT output the `"time"` field.
 - **Condition C (Timeless or Unresolvable)**: If there is absolutely no temporal information, if it is a timeless fact, OR if you cannot resolve a valid ISO 8601 timestamp from the context.
 - Set `time_info` entirely to `null`. (Do not output an empty object `{}`).
5. **Contextual Keyword Synthesis**: Extract, merge, and deduplicate the most important keywords (personas, entities, locations, concepts) across the cluster. Provide a `contextual_description` that reflects their overarching role or state in this *newly abstracted* narrative, rather than their isolated role in the lower-level entries.
 - **CRITICAL**: DO NOT extract dates, times, or timestamps as keywords.

[Output Format]

Return a single JSON object only:

```
{
  "abstracted_summary": "A coherent, high-level synthesis of the clustered entries...",
  "topic": "Concise overarching topic phrase",
  "time_info": {
    "relation": "now|before|after|span",
    "time": "YYYY-MM-DDTHH:MM:SS", /* Include only if relation is not span */
    "start_time": "YYYY-MM-DDTHH:MM:SS", /* Include only if relation is span */
    "end_time": "YYYY-MM-DDTHH:MM:SS" /* Include only if relation is span */
  } /* or null */,
  "contextual_keywords": [
    {
      "keyword": "keyword name",
      "category": "persona|entity|location|concept",
      "contextual_description": "Updated description of this keyword's role in the abstracted narrative"
    }
  ]
}
```

[Example]

{Examples}

Now summarize the provided clustered entries. Return ONLY the JSON object, nothing else.

C.3 Query Metadata Parsing Prompt

We implement query metadata parsing in two parallel steps. The LLM extracts the query-side temporal intent, time relation, event anchor, and normalized target time. Simultaneously, the LLM extracts

the query topic and contextual keywords by considering both the user query and candidate keywords retrieved from the entity pool.

Temporal and Event Information Extraction Prompt

Role
You are an expert Temporal and Event Information Extraction Agent for a memory retrieval system. Your specific task is to analyze the user's query and extract its temporal intent, event anchors, and standardized time constraints.

Context
Current System Time: {current_time}
(You MUST use this reference time to properly resolve any relative time expressions like "yesterday", "last year", "3 days ago").

Category Definitions

1. [Intent Categories] (`temporal\_intent`)
- EXPLICIT_TIME: The query targets a specific date, time, or relative time expression (e.g., "in May 2023", "yesterday", "since last month").
- RELATIVE_EVENT: The query asks what happened BEFORE or AFTER a specific event (e.g., "What did she do before the charity race?").
- NONE: A purely factual query with no temporal aspect or event sequence.

2. [Time Relation] (`time\_relation`)
- BEFORE: Information prior to the time/event.
- AFTER: Information subsequent to the time/event.
- AT: Information at the exact time/event.
- NONE: No specific relation.

3. [Event Anchor] (`event\_anchor`)
- If the intent is RELATIVE_EVENT, extract the core event being used as an anchor in the format "[Subject] [Action] [Object/Target]" (e.g., "Melanie run charity race", "Caroline go to support group").
- Otherwise, return null.

Time Parsing Rules (CRITICAL)
- If `temporal\_intent` is EXPLICIT_TIME, you MUST convert relative time expressions into absolute dates based on the Current System Time.
- Standardize the `target\_time` to an ISO 8601 date format (YYYY-MM-DDTHH:MM:SS).
- Fallback rules: If only the year is mentioned (e.g., "after 2021"), default to "YYYY-01-01T00:00:00". If only year and month are mentioned, default to "YYYY-MM-01T00:00:00".

Output Format
You must output ONLY a valid JSON object. Do not include any markdown formatting, explanations, or conversational text. Use the following schema:

```
{
  "temporal_intent": "EXPLICIT_TIME" | "RELATIVE_EVENT" | "NONE",
  "time_relation": "BEFORE" | "AFTER" | "AT" | "NONE",
  "event_anchor": "Subject Action Object" | null,
  "target_time": "YYYY-MM-DDTHH:MM:SS" | null
}
```

Examples
{Examples}

Task
User Query: {query}
Output:

Topic and Keyword Extraction Prompt

Your task is to extract structured topic and semantic metadata from the following user query, using the provided candidate keywords.

[User Query]

```
{query_text}

[Candidate Keywords]
{candidate_keywords}

[Requirements]
1. Topic Specification: Provide a concise `topic` phrase that summarizes the core subject or intent of the user query.
2. Force Disambiguation: Absolutely PROHIBIT using pronouns (he, she, it, they, this, that) in your extracted outputs. Infer the actual entity based on the context.
3. Comprehensive Keyword Extraction (Query + Candidates):
  - Extract explicit key entities (personas, locations, objects, concepts) from the User Query.
  - CRITICAL CANDIDATE EXPANSION: You MUST actively scan the [Candidate Keywords]. If the user query is looking for a specific type of target (e.g., asking "Which book...", "What movie...", "Who..."), you MUST extract ALL specific instances of that target (e.g., all specific book titles) found in the candidate list to expand the search evidence.
  - Provide a `contextual_description` explaining each keyword's specific role in the query or why it is a relevant candidate evidence.
4. Temporal Exclusion: DO NOT extract dates, times, days of the week, or temporal concepts as keywords. Ignore time context completely.

[Output Format]
Return ONLY a strictly valid JSON object. No other explanations.

{
  "topic": "Concise topic phrase summarizing the query intent",
  "contextual_keywords": [
    {
      "keyword": "keyword name",
      "category": "persona|entity|location|concept",
      "contextual_description": "Brief description of this keyword's specific role or state in this query"
    }
  ]
}

[Example]
{Examples}

Now process the above query. Return ONLY the JSON object, no other explanations.
```

C.4 Answer Generation Prompt

This prompt is used to generate the final answer to a user query given the interaction time.

Answer Generation Prompt

You are an expert analytical assistant. Please answer the user's query by analyzing the provided evidences.

[Current System Time]
{current_time}

Provided Evidences:
{evidences}

Task Rules (CRITICAL):

1. **Chronological Priority (Knowledge Update)**: Pay strict attention to the timestamps attached to the evidences. If you encounter conflicting information regarding states, counts, preferences, or habits, the evidence with the most recent timestamp represents the current and correct truth. You MUST rely on the latest information and override older data.

2. **Temporal Reasoning**: Use the [Current System Time] provided above as your anchor to resolve any relative

time references in the User Query (e.g., "last Saturday", "two months ago", "recently").

3. ****Anti-Hedging & Premise Acceptance****: Treat the premises in the user's query as absolute facts. Do not hedge, condition your answers (e.g., avoid saying "If Max is your dog..."), or provide unsolicited advice/lectures. Employ semantic flexibility when matching query terms to evidence (e.g., understanding "jewelry" might refer to an inherited "crystal chandelier" or similar heirloom).
4. ****Strict Grounding & Anti-Hallucination****: Construct your response using ONLY the provided evidences. You are strictly forbidden from performing calculations with assumed, external, or fabricated numbers.
5. ****Missing Information****: If a variable or fact required to fully answer the question is missing from the evidences, you must explicitly state: "Information is missing." Do not attempt to guess or estimate.
6. ****Tone and Format****: Keep the tone professional, clear, and direct. At the very end of your response, provide a concise answer directly addressing the User Query wrapped in `<answer>` and `</answer>` tags.

User Query:
{query}

Final Answer:

C.5 LLM-as-Judge Prompt

This prompt is used to compute LLM-as-Judge accuracy.

LLM-as-Judge Prompt

After reading some text, John was given the following question about the text:
{question}

John's answer to the question was:
{prediction}

The ground truth answer was:
{answer}

Does John's answer agree with the ground truth answer? (Please only answer YES or NO)

D Evaluation Protocol

We evaluate TSHMem from two perspectives: answer correctness and budgeted evidence utility. All methods use the same backbone model, decoding setting, and judge prompt for fair comparison. Unless otherwise specified, all reported results are obtained from a single run with fixed hyperparameters, and we report macro-averaged metrics over evaluation queries.

D.1 Answer Evaluation

For answer-level evaluation, we report LLM-as-Judge Accuracy as the primary metric and Token F1 as a complementary lexical metric. For each query, the judge model is given the query q , the reference answer a , and the predicted answer $\hat{a}$, and outputs a binary correctness label. A prediction is considered correct if it is semantically consistent with the reference answer and directly answers

the query. The judge prompt is provided in Appendix C.5.

Token F1 is computed between the normalized prediction and reference answer. We lowercase the text, remove common punctuation marks, split it into whitespace-separated tokens, and measure token-level overlap between the prediction and reference. We use it only as a complementary metric, since lexical overlap may underestimate semantically correct answers in long-term conversational QA.

D.2 Budgeted Evidence Evaluation

To fairly compare retrieval quality across methods, we conduct a budgeted evidence evaluation with a 4096-token evidence limit. Since different methods retrieve evidence at different granularities and the benchmarks provide gold supporting evidence at the turn level, we first map all retrieved evidence to turn-level support. For TSHMem, retrieved memory entries are mapped to their source dialogue turns through trace-back edges. For chunk-based baselines, each retrieved chunk is expanded to the dialogue turns it contains.

For each method, retrieved evidence is ranked by its retrieval score. We keep evidence items in ranked order until the total evidence length reaches the 4096-token budget or no retrieved item remains. We report Recall@4K, Acc.@4K, and the average number of retrieved evidence tokens. Recall@4K measures the coverage of gold supporting turns within the budgeted evidence. Acc.@4K measures answer accuracy when the model generates answers using only the retained budgeted evidence. The token count measures the average evidence length after budget truncation and is used to compare retrieval efficiency across methods.

Sparsity grouping. We further analyze budgeted evidence utility under different entity sparsity levels. For each query, we first identify evidence-related candidate entities from the memory graph using the turn-level gold supporting evidence. We then use the LLM, with the keyword extraction prompt in Appendix C.3, to prune them into a smaller set of contextual entities that are directly relevant to the query. We use $n(e)$ to denote the number of memory nodes associated with retained entity e . For query q_i , we define its entity-density score d_i as the average degree of its retained entities. If no entity is retained after pruning, we set $d_i = 0$. We divide queries into three sparsity

groups based on d_i :

$$\begin{aligned} \text{SPARSE} : & \quad d_i \leq 2, \\ \text{MEDIUM} : & \quad 2 < d_i \leq 5, \\ \text{DENSE} : & \quad d_i > 5. \end{aligned} \quad (15)$$

The SPARSE group corresponds to long-tail entities that are associated with only a few memory nodes, while the DENSE group corresponds to high-degree entities that appear in many memory nodes and are more likely to introduce retrieval noise.

E Additional Experimental Results

This section reports supplementary analyses that expand the main experimental results. We first add an extended baseline comparison, then provide detailed category-level results, additional component sensitivity analyses, evidence-efficiency results under sparsity, and robustness checks for the backbone and evaluation protocol.

E.1 Extended Baseline Comparison

We compare TSHMem with MIRIX under a unified evaluation protocol.

E.1.1 Direct Comparison with MIRIX

MIRIX (Wang and Chen, 2025) is a multi-agent memory system. Since its published LoCoMo result uses a different backbone and evaluation subset from our main all-category LoCoMo evaluation, we rerun MIRIX with the same OAI-4o-mini answer model, generation prompt, and judge protocol used for TSHMem on LongMemEval S* and LoCoMo. On LoCoMo, both systems are evaluated on the same non-adversarial QA subset with valid outputs. Tables 12 and 13 report the comparison.

On LongMemEval S*, TSHMem achieves an absolute overall accuracy gain of 18.0% over MIRIX and performs better across all six categories, with particularly large gains on SU, SA, TR, and KU. On the non-adversarial LoCoMo QA subset, TSHMem achieves an absolute overall accuracy gain of 4.5%, driven mainly by gains on SingleHop and Temporal, while MIRIX performs better on MultiHop and OpenDomain. Overall, TSHMem maintains higher accuracy after extending the baseline set to MIRIX.

E.2 Category-Level Results and Diagnostics

We report detailed category-level results on LongMemEval S* and LoCoMo, then examine OpenDomain and Adversarial behavior on LoCoMo.

E.2.1 Detailed Fine-grained Results

Tables 14 and 15 report detailed category-level results on LongMemEval S* and LoCoMo, including both LLM-as-Judge Accuracy and Token F1.

Overall, the fine-grained results are consistent with the main findings: TSHMem is strongest on categories that require cross-session evidence aggregation and temporal reasoning. Improvements are smaller on localized or preference-oriented categories, suggesting that single-session evidence and implicit preference signals may require different retrieval or modeling strategies. LLM-as-Judge Accuracy and Token F1 do not always rank methods consistently, especially on LoCoMo. This difference is expected because they assess semantic correctness and lexical overlap, respectively; representative cases are analyzed in Appendix E.5.3.

E.2.2 OpenDomain Analysis

TSHMem generates answers strictly from retrieved evidence and returns “Information is missing.” when the required facts are unavailable. This conservative strategy is intended to limit unsupported responses but may be less effective for OpenDomain questions that benefit from broader conversational context or background knowledge. To examine the effect of evidence coverage, we evaluate all methods using fixed 4096-token evidence contexts and a unified answer style. Under this setting, TSHMem reaches 46.9% accuracy and outperforms BM25 and A-Mem (Table 16), indicating that OpenDomain performance is partly associated with evidence coverage and answer style.

E.2.3 Adversarial QA Analysis

As shown in Table 15, TSHMem obtains 88.1% accuracy on Adversarial QA. Table 17 further compares BM25 and A-Mem in their original setting and a 4096-token evidence budget setting with answer style matched to TSHMem. Both methods improve substantially in the controlled setting, consistent with reduced distractor exposure.

E.3 Additional Component Sensitivity

We analyze the sensitivity of the local structural weight β and hierarchy weights γ and λ . We conduct this sweep on LongMemEval Oracle, whose shorter evidence contexts make repeated runs more tractable. Each hyperparameter configuration is evaluated over three runs, and we report the mean performance. The results characterize sensitivity around the default hyperparameter setting.

Method	Overall	SU	SA	SP	MS	TR	KU
MIRIX	48.3	66.7	60.0	13.3	32.0	52.0	66.7
TSHMem	66.3	91.1	83.3	20.0	46.7	69.3	88.9

Table 12: Direct MIRIX comparison on LongMemEval S*. All values are LLM-as-Judge Accuracy ($\uparrow$).

Method	Overall	SH	MH	Temp.	OD
MIRIX	56.8	63.8	46.4	57.7	35.8
TSHMem	61.3	70.4	42.3	67.3	33.7

Table 13: Direct MIRIX comparison on LoCoMo under a unified non-adversarial evaluation setting. All values are LLM-as-Judge Accuracy ($\uparrow$).

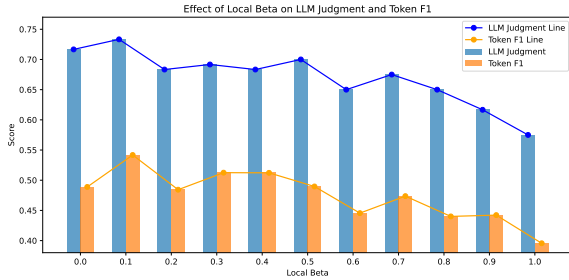


Figure 3: Impact of the local structural weight β on LLM-as-Judge Accuracy and Token F1 on LongMemEval Oracle.

Effect of local structural weight β . Figure 3 shows the effect of the structural-semantic fusion weight β . When $\beta = 0$, retrieval removes the PPR structural contribution and retains semantic matching over hierarchical memory units together with the query-conditioned temporal relevance $D(v)$; this approximates dense retrieval plus a temporal relevance filter. Moving to the default $\beta = 0.1$ increases LongMemEval Oracle accuracy from 71.7% to 73.3%, an absolute gain of 1.6%. This indicates that direct semantic matching remains important while even a small amount of PPR structural propagation recovers useful evidence beyond surface similarity. When $\beta = 1$, retrieval relies only on structural propagation. Performance generally decreases when β becomes too large, suggesting that excessive graph propagation may introduce irrelevant neighboring memories.

Effect of hierarchy weights γ and λ . Figure 4 shows the effect of the top-down weight γ and bottom-up weight λ in hierarchical ranking. The best result is achieved when $\gamma = 0.1$ and $\lambda = 0.7$, suggesting that bottom-up evidence from fine-grained child nodes is more useful than strong top-

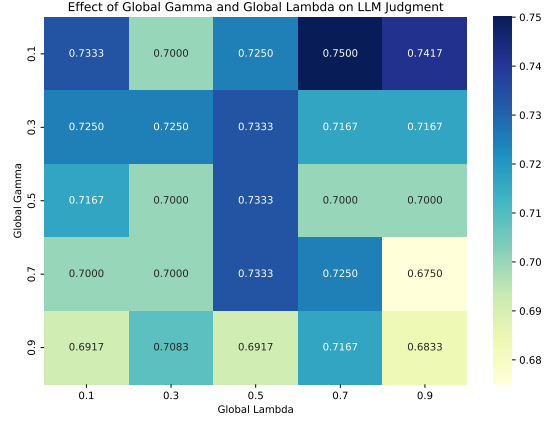


Figure 4: Effect of top-down weight γ and bottom-up weight λ on LLM-as-Judge Accuracy on LongMemEval Oracle.

down propagation from abstract summaries.

E.4 Evidence Efficiency under Sparsity

Table 18 reports Acc.@4K under different evidence-entity density levels. The grouping follows the protocol described in Appendix D.2.

The Acc.@4K results are consistent with the Recall@4K trends reported in the main text. This suggests that the improved retrieval quality also translates into better answer accuracy. Meanwhile, the lower accuracy in the Sparse group indicates that weakly connected evidence remains more challenging than evidence in denser memory regions.

E.5 Robustness and Evaluation Validity

We report robustness checks for two sources of variation: the backbone model and the judge model. We also include representative Token F1 cases to clarify metric behavior on LoCoMo.

E.5.1 Backbone Model Robustness

To evaluate robustness to the choice of backbone, we replace TSHMem’s OAI-4o-mini backbone with the open-source Qwen3-8B model (Yang et al., 2025) while leaving the remaining evaluation protocol unchanged. Table 19 shows modest absolute decreases of 3.3% in Accuracy and 1.0% in Token F1 on LongMemEval S*, indicating that TSHMem remains effective with an open-source backbone,

Type	Method	SU		SA		SP		MS		TR		KU	
		Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
Standard RAG	BM25	91.1	36.5	86.7	55.4	20.0	13.9	24.0	9.7	24.0	24.8	62.2	20.1
	Contriever	33.3	12.4	33.3	26.2	13.3	13.8	9.3	6.5	20.0	20.9	26.7	9.4
	Text-Emb-3-Large	64.4	31.9	60.0	41.6	20.0	14.5	17.3	10.2	20.0	20.0	51.1	21.7
	Qwen3-Emb-4B	77.8	33.9	83.3	52.1	20.0	14.4	26.7	12.8	20.0	23.7	62.2	21.8
Hierarchical/ Graph RAG	RAPTOR	66.7	49.8	76.7	58.2	10.0	9.5	18.7	14.6	10.7	15.6	46.7	30.6
	GraphRAG	51.1	36.1	63.3	51.3	13.3	8.1	16.0	7.4	30.7	27.0	48.9	31.4
	HippoRAG-v2	77.8	67.5	73.3	61.3	26.7	9.9	22.7	23.8	30.7	33.6	66.7	48.7
Memory Agents	Mem0	17.8	12.5	13.3	25.6	13.3	6.9	5.3	8.5	14.7	25.0	20.0	10.3
	Cognee	28.9	4.3	36.7	10.8	3.3	7.6	9.3	3.3	26.7	8.8	26.7	2.7
	Graphiti	22.2	16.0	16.7	27.5	13.3	9.1	8.0	7.6	14.7	22.8	31.1	21.0
	A-Mem	35.6	18.8	40.0	34.8	16.7	10.8	12.0	9.7	22.7	28.1	28.9	16.3
	LightMem	82.2	67.8	16.7	23.3	20.0	10.4	42.7	33.9	37.3	38.8	77.8	53.7
	SimpleMem	68.9	57.2	43.3	39.3	10.0	9.8	16.0	14.9	18.7	23.7	51.1	32.1
Ours	TSHMem	91.1	74.5	83.3	68.8	20.0	15.9	46.7	32.5	69.3	42.3	88.9	57.5

Table 14: Detailed fine-grained results on LongMemEval S*. Acc. (↑) denotes LLM-as-Judge Accuracy, and F1 (↑) denotes Token F1. Text-Emb-3-Large and Qwen3-Emb-4B denote Text-Embeddings-3-Large and Qwen3-Embeddings-4B, respectively. SU, SA, SP, MS, TR, and KU denote Single-Session User, Single-Session Assistant, Single-Session Preference, Multi-Session, Temporal Reasoning, and Knowledge Update, respectively.

Type	Method	SH		MH		Temp.		OD		Adv.	
		Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
Standard RAG	BM25	79.8	50.6	40.4	33.3	41.7	23.7	46.9	22.8	50.4	36.0
	Contriever	80.3	51.4	39.7	31.8	42.1	24.0	53.1	24.5	50.2	34.0
	Text-Emb-3-Large	79.5	51.1	39.0	31.0	35.8	21.9	49.0	24.6	51.3	33.3
	Qwen3-Emb-4B	80.3	51.7	37.9	30.1	39.3	23.5	50.0	23.3	48.2	33.1
Hierarchical/ Graph RAG	RAPTOR	39.7	33.9	24.8	33.6	32.1	23.1	34.4	19.4	18.2	17.6
	GraphRAG	27.5	23.2	13.5	18.6	19.6	15.4	38.5	25.4	24.7	22.6
	HippoRAG-v2	66.0	54.6	34.4	38.7	31.5	27.1	41.7	19.6	34.8	30.0
Memory Agents	Mem0	17.8	16.2	13.1	16.0	10.9	8.6	37.5	21.7	15.7	15.3
	Cognee	43.2	18.2	18.4	11.5	10.9	4.5	30.2	6.0	40.1	18.2
	Graphiti	16.8	16.1	13.1	15.8	5.6	4.8	28.1	19.8	55.2	8.9
	A-Mem	70.6	47.4	35.5	29.4	29.0	16.2	51.0	24.0	46.2	32.5
	LightMem	18.2	17.7	13.8	15.8	10.3	7.7	32.3	21.4	16.6	16.3
	SimpleMem	70.3	52.4	36.9	36.3	55.1	53.8	43.8	21.8	34.5	26.3
Ours	TSHMem	71.3	36.2	41.8	24.4	66.7	25.0	34.4	13.3	88.1	45.9

Table 15: Detailed fine-grained results on LoCoMo. Acc. (↑) denotes LLM-as-Judge Accuracy, and F1 (↑) denotes Token F1. Text-Emb-3-Large and Qwen3-Emb-4B denote Text-Embeddings-3-Large and Qwen3-Embeddings-4B, respectively. SH, MH, Temp., OD, and Adv. denote SingleHop, MultiHop, Temporal, OpenDomain, and Adversarial, respectively.

Method	OD Acc. (4K Context) ↑
TSHMem	46.9
BM25	39.5
A-Mem	37.5

Table 16: LoCoMo OpenDomain accuracy with the evidence context fixed at 4096 tokens.

Method	Acc.@All ↑	Acc.@4K ↑	Δ
BM25	50.4	73.8	+23.4
A-Mem	46.2	88.3	+42.1

Table 17: LoCoMo Adversarial accuracy in the original full-context and controlled 4K-context settings.

although backbone capability affects overall performance.

E.5.2 LLM-as-Judge Robustness

The main results use OAI-4o as the judge model. We repeat the LongMemEval S* answer evaluation with OAI-4o and additionally evaluate it with GDM-3.1-Pro-Preview to assess robustness to the choice of judge model. Table 20 reports the original result together with the mean and standard deviation over three repeated evaluation runs. The ranking is stable across judges: TSHMem remains the strongest method and exhibits low variance, supporting the reliability of the headline result.

E.5.3 Token F1 Case Analysis

Token F1 measures lexical overlap and can underestimate correct answers that use a different surface form or provide a more complete response. Figure 5 gives two representative LoCoMo cases.

In the first case, TSHMem gives the correct date in ISO format and receives zero token overlap, whereas SimpleMem gives the wrong day but retains overlapping month and year tokens. In the second, SimpleMem omits one required modification but has high overlap, while TSHMem gives the complete answer with additional sentence context. These cases motivate treating Token F1 as a complementary lexical metric rather than the primary measure of semantic correctness.

Method	Sparse	Medium	Dense
BM25	25.0	34.3	30.6
HippoRAG-v2	27.8	28.6	32.8
A-Mem	27.8	14.3	15.7
TSHMem	44.4	68.6	68.1

Table 18: Acc.@4K (↑) under different evidence-entity density levels. Sparse, Medium, and Dense groups are defined according to the average degree of pruned evidence-related entities in the memory graph.

Backbone	Accuracy ↑	Token F1 ↑
OAI-4o-mini	66.3	47.0
Qwen3-8B	63.0	46.0

Table 19: LongMemEval S* robustness to the backbone model.

Method	Original	OAI-4o	GDM
A-Mem	24.00	24.11 ± 0.16	26.00 ± 0.27
BM25	45.67	45.44 ± 0.31	47.11 ± 0.16
HippoRAG-v2	45.00	44.67 ± 0.27	46.11 ± 0.31
Qwen3-Emb-4B	43.00	42.44 ± 0.42	44.56 ± 0.31
LightMem	47.67	48.33 ± 0.54	49.00 ± 0.47
SimpleMem	32.00	32.89 ± 0.63	35.56 ± 0.31
TSHMem	66.33	65.78 ± 0.42	66.78 ± 0.16

Table 20: LongMemEval S* LLM-as-Judge robustness. Original denotes the reported OAI-4o result; the OAI-4o and GDM columns report mean ± standard deviation over three runs, where GDM denotes GDM-3.1-Pro-Preview. Qwen3-Emb-4B denotes Qwen3-Embeddings-4B. Values are Accuracy (↑).

Case: conv-26_qa16

Question: When did Melanie sign up for a pottery class?
Reference: “2 July 2023”
SimpleMem (Acc.=0, F1=0.667): “03 July 2023”
TSHMem (Acc.=1, F1=0.000): “2023-07-02”

Case: conv-50_qa95

Question: What kind of modifications has Dave been working on in the car mod workshop?
Reference: “engine swaps, suspension modifications, and body modifications”
SimpleMem (Acc.=0, F1=0.909): “Engine swaps and suspension modifications”
TSHMem (Acc.=1, F1=0.545): “Dave has been working on engine swaps, suspension modifications, and body modifications in the car mod workshop.”

Figure 5: Representative LoCoMo cases where lexical overlap and semantic correctness disagree.