DyMem: Query-Adaptive Relational Memory for Long-Term Conversation

Anonymous authors
Paper under double-blind review

Abstract

Long-term conversational agents must reason over information accumulated across many interactions, where answering a question often requires connecting distributed memories through temporal, topical, or entity-level relations. Existing flat retrieval methods rank memories independently and may overlook evidence that is weakly aligned with the query but connected to relevant memories through meaningful relations. Graph-based memory systems make such dependencies explicit, yet a persistent graph defines only a space of possible connections, while fixed or query-agnostic traversal can follow irrelevant relational paths and introduce unnecessary context. We propose **DyMem**, a query-adaptive relational memory framework that conditions retrieval not only on memory-node relevance, but also on the semantic meaning of the relations connecting memories. DyMem constructs a persistent heterogeneous memory graph with explicit relation descriptions and dynamically activates query-relevant relations to form a query-specific evidence subgraph. The resulting candidates are further pruned under a context budget and mapped back to the original dialogue turns for source-grounded answer generation. Experiments on LoCoMo and LongMemEval across multiple language model backbones show that DyMem consistently improves long-term conversational question answering over representative memory baselines, with particularly strong gains on multi-hop and temporal queries while maintaining a compact retrieval context. These results highlight the importance of treating relations themselves as query-dependent retrieval signals in long-term conversational memory.

1 Introduction

Long-term conversational memory is essential for autonomous agents to maintain context and accumulate knowledge across extended interactions (Packer et al., 2023; Zhong et al., 2024; Wang et al., 2023). However, answering complex queries often requires reasoning over *distributed evidence*, where relevant information is scattered across different sessions, times, topics, or entities (Maharana et al., 2024). Conventional memory systems typically treat memories as independent units and retrieve them through direct semantic similarity (Lewis et al., 2020). While effective for directly relevant facts, such flat retrieval can miss crucial evidence when its surface form is weakly aligned with the query but its underlying content is connected to other relevant memories through temporal, semantic, or entity-level relations (Jiménez Gutiérrez et al., 2024). For example, answering a temporal question may require connecting a sequence of historical events, even when some intermediate events have little direct semantic similarity to the query.

To capture such dependencies, recent research has increasingly explored structured memory organization. Early persistent memory systems focused on retrieval, context management, and memory consolidation, including MemGPT (Packer et al., 2023), AgenticMemory (Yu et al., 2026), and Mem0 (Chhikara et al., 2025), while methods such as *SimpleMem* (Liu et al., 2026) further improve memory efficiency through structured compression and query-aware consolidation. More recently, graph-based memory architectures have emerged to explicitly represent relations among historical information. Approaches such as GraphRAG (Edge et al., 2024), HippoRAG (Jiménez Gutiérrez et al., 2024), *A-MEM* (Xu et al., 2025), and *Nemori* (Nan et al., 2025) organize information into interconnected nodes or episodes, enabling retrieval through temporal, semantic, or thematic relations. These developments demonstrate that persistent relational structures can expose dependencies that

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

are difficult to recover through independent memory retrieval. However, a relational memory graph defines a space of possible connections rather than the evidence required for a particular query. The same memory node may participate in multiple heterogeneous relations, while different queries may require following substantially different relational paths. Recent studies have therefore begun to explore query-adaptive retrieval and graph traversal (Tian et al., 2026; Lau et al., 2026; Zhou et al., 2026).

This observation suggests that the central challenge in long-term conversational memory is not merely how to *represent* relations, but how to determine which *relations themselves* are relevant to a specific information need. Conversational memory graphs contain heterogeneous dependencies, such as temporal succession, semantic association, and topical connectivity, whose meanings provide useful signals for deciding whether a transition is relevant to the current query. A temporal question may require tracing chronological dependencies across sessions, whereas a preference question may require connecting interactions associated with a shared entity or topic (Wu et al., 2025). Thus, as shown in Figure 1, ***query adaptation should operate not only on memory nodes, but also on the semantic meaning of the relations connecting them***. We formulate this problem as *query-adaptive relational evidence selection*: given a persistent heterogeneous memory graph, the goal is to identify a compact, query-specific evidence subgraph by selectively following relations that support the current information need while suppressing irrelevant relational paths. Moreover, because long-term memory systems often rely on summarized or abstracted representations, retrieval should ultimately remain grounded in the original conversational source to preserve factual details.

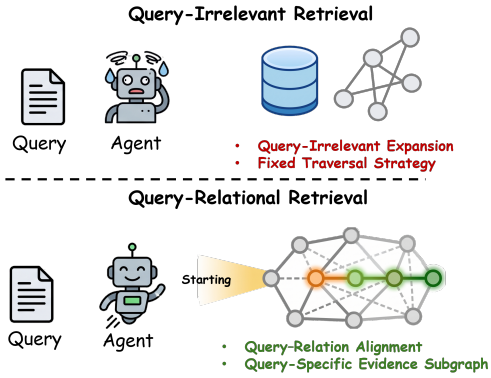

Figure 1: Query-Irrelevant and Query-Relational Retrieval. DyMem dynamically selects query-relevant relations from a same persistent memory graph, forming a query-specific evidence subgraph instead of relying on fixed graph expansion.

Motivated by this perspective, we propose **DyMem**, a query-adaptive relational memory framework for long-term conversation. DyMem maintains a persistent heterogeneous memory graph with explicit relation rationales, enabling retrieval to assess not only memory-node relevance but also the semantic relevance of connecting relations. Given a query, DyMem aligns the query with candidate relation rationales to activate relevant edges, guides bounded graph traversal with the resulting relation-aware weights, and selects a compact evidence set by balancing relevance, coverage, redundancy, and context cost. The selected memories are finally mapped back to their original dialogue turns for source-grounded answer generation. The main contributions of this work are summarized as follows:

- We formulate long-term conversational memory retrieval as a *query-adaptive relational evidence selection* problem, where retrieval decisions are conditioned on the semantic meaning of relations connecting historical memories rather than solely on memory-node similarity.
- We propose **DyMem**, a query-adaptive relational memory framework that represents heterogeneous conversational dependencies with explicit relation rationales, dynamically activates query-relevant relations, and performs bounded evidence traversal followed by utility-driven selection. A source-preserving reading mechanism then grounds the selected relational evidence back to the original dialogue turns.
- We evaluate DyMem on LoCoMo and LongMemEval across multiple language model backbones. DyMem consistently improves long-term memory question answering over representative memory systems, with particularly strong gains on multi-hop and temporal queries, while reducing the amount of retrieved context.

2 Related Work

Long-Term Memory in Autonomous Agents Long-term memory systems enable autonomous agents to retain and retrieve information across extended interactions. MemGPT (Packer et al., 2023) manages unbounded context through memory paging and context management, while AgenticMemory (Yu et al., 2026) and Mem0 (Chhikara et al., 2025) provide persistent memory management and consolidation across interactions. *SimpleMem* (Liu et al., 2026) further introduces structured memory compression and query-aware consolidation for efficient lifelong memory. Other agent architectures explore episodic and reflective memory, including Generative Agents (Park et al., 2023), which maintains a persistent memory stream and higher-level reflections, and Reflexion (Shinn et al., 2023), which stores verbal feedback in an episodic memory buffer for subsequent decisions.

Graph-Structured Memory and Retrieval Graph-based methods organize information into relational structures to facilitate retrieval and multi-hop reasoning. GraphRAG (Edge et al., 2024) constructs graph representations for query-focused retrieval, while HippoRAG (Jiménez Gutiérrez et al., 2024) uses graph-based associations to support long-range retrieval. G-Retriever (He et al., 2024) retrieves compact evidence subgraphs from textual graphs, and LightRAG (Guo et al., 2025) performs hierarchical retrieval over entities and relations. In conversational memory, *A-MEM* (Xu et al., 2025) constructs interconnected memory notes with evolving relations, while *Nemori* (Nan et al., 2025) organizes historical interactions into evolving episodic structures. These methods establish relational memory spaces in which distributed historical information can be connected through explicit or implicit dependencies.

Query-Adaptive Retrieval Recent work further investigates how retrieval can adapt to the current query. Adaptive-RAG (Jeong et al., 2024) selects retrieval strategies according to query complexity, while SwiftMem (Tian et al., 2026) introduces query-aware indexing for agent memory. For graph-based retrieval, QAFD-RAG (Zhou et al., 2026) dynamically adjusts graph propagation according to query relevance and performs query-aware subgraph retrieval, while CatRAG (Lau et al., 2026) explores query-aware dynamic edge weighting for graph traversal. DyMem instead targets long-term conversational memory and makes the semantic content of heterogeneous relations an explicit retrieval signal: query-relation alignment determines which relational paths are activated, after which the selected memory structure is grounded back to the original dialogue turns.

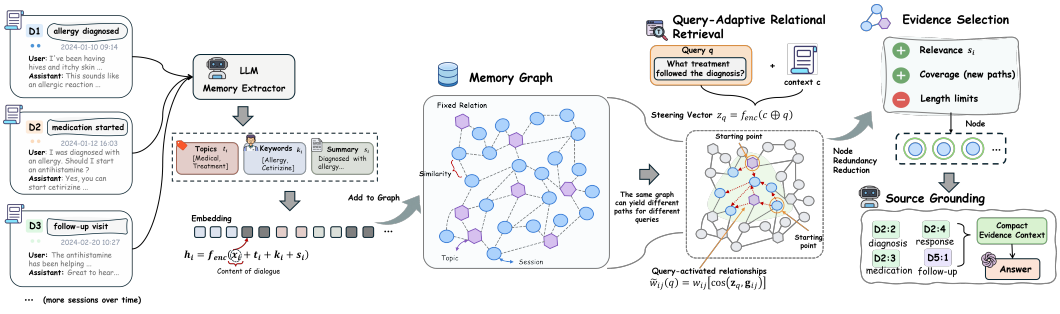

Figure 2: Overview of DyMem. Incoming dialogue sessions are processed by an LLM to extract topics, keywords, summaries, and memory representations, which are organized into a persistent relational memory graph. Given a query q and context c, DyMem constructs a steering vector and query-conditionally activates graph relations according to their semantic compatibility, allowing the same graph to yield different retrieval paths for different queries. Evidence selection balances relevance, complementary path coverage, and context length. Finally, source grounding maps the selected relational evidence back to compact verbatim dialogue turns for answer generation.

3 METHOD

DyMem maintains an evolving relational memory graph and, for each query, activates relevant relations to construct a compact, source-grounded evidence subgraph. Implementation details and hyperparameters are provided in Appendix A.

3.1 PROBLEM FORMULATION

Let $\mathcal{H} = \{H_1, \ldots, H_T\}$ be a multi-session dialogue history and q the current query. Given memory notes $\mathcal{M} = \{m_1, \ldots, m_N\}$, our goal is to retrieve a compact evidence set $\mathcal{E}_q$ sufficient to answer q. Unlike flat retrieval, which ranks memories independently, DyMem represents persistent memory as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ contains episodic memory nodes and $\mathcal{E}$ contains typed memory–memory relations. An auxiliary topic index supports seed discovery but is not traversed.

3.2 MEMORY GRAPH CONSTRUCTION

Each memory m_i retains its verbatim content x_i and timestamp τ_i. An LLM extracts a summary s_i, keywords k_i, and topics t_i, yielding

$$\mathbf{h}_i = f_{\text{enc}}(x_i \oplus s_i \oplus k_i \oplus t_i), \tag{1}$$

where f_{enc} is initialized from all-MiniLM-L6-v2 (Wang et al., 2020; Reimers & Gurevych, 2019) and post-trained for our setting.

The graph grows as memories arrive. Each memory is linked bidirectionally to its immediate predecessor, and a semantic edge is added when

$$\text{sim}(m_i, m_j) = [\cos(\mathbf{h}_i, \mathbf{h}_j)] \geq \delta. \tag{2}$$

The query-independent base weight of a candidate edge is

$$w_{ij}^{(\sigma)} = \begin{cases} \text{sim}(m_i, m_j), & \sigma = \text{semantic}, \\ \text{clip}_{[0,1]}(w_{ij}^{\text{LLM}}), & \sigma = \text{LLM}, \end{cases} \tag{3}$$

where w_{ij}^{LLM} is proposed by the memory controller. If multiple edges connect the same ordered pair, the strongest is retained and denoted w_{ij}. Each retained edge has a relation type r_{ij} and textual description d_{ij}, represented as

$$\mathbf{g}_{ij} = f_{\text{enc}}(r_{ij} \oplus d_{ij}). \tag{4}$$

Descriptions are either LLM-generated rationales or structured descriptions derived from the relation and endpoint metadata.

For a topic label l_u, we compute $\mathbf{h}_u = f_{\text{enc}}(l_u)$ and its membership strength to memory m_i as

$$a_{ui} = [\cos(\mathbf{h}_u, \mathbf{h}_i)]\,. \tag{5}$$

3.3 QUERY-ADAPTIVE RELATIONAL RETRIEVAL

Given optional conversational context c, the query representation is $\mathbf{z}_q = f_{\text{enc}}(c \oplus q)$. DyMem obtains seeds from both memories and topics:

$$s_i^{\text{node}} = [\cos(\mathbf{z}_q, \mathbf{h}_i)]\,, \tag{6}$$

$$s_{ui}^{\text{seed}} = [\cos(\mathbf{z}_q, \mathbf{h}_u)]\, a_{ui}. \tag{7}$$

DyMem then conditions each traversable relation on the query:

$$\widetilde{w}_{ij}(q) = w_{ij}\,[\cos(\mathbf{z}_q, \mathbf{g}_{ij})]\,. \tag{8}$$

Thus, the persistent graph remains fixed during retrieval, while its effective connectivity changes with the query. Starting from the seeds, a bounded multi-source search expands only the highest-activation neighbors subject to a maximum depth, branch factor, and candidate limit. Repeated states are identified by memory ID and depth; the traversal bounds are specified in Section A.3.

For a discovered path $p = (v_0, \ldots, v_d)$, its relational relevance is

$$S_{\text{path}}(p \mid q) = \prod_{\ell=1}^{d} \widetilde{w}_{v_{\ell-1}v_\ell}(q). \tag{9}$$

Let $\mathcal{P}_i^{\text{search}}(q)$ be the explored paths reaching memory i, and let $d(p)$ denote the number of edges in p. DyMem retains the highest-scoring explored path and assigns

$$p_i^* = \arg \max_{p \in \mathcal{P}_i^{\text{search}}(q)} S_i(p), \tag{10}$$

$$S_i(p) = \alpha s_i^{\text{node}} + \beta S_{\text{path}}(p \mid q) + \frac{\gamma}{d(p)+1}. \tag{11}$$

This is the best path within the bounded search space, rather than a global aggregation over all graph paths.

3.4 Evidence Selection and Source Grounding

The retrieved candidates are greedily pruned under a context budget. Given an already selected set A, candidate i has marginal utility

$$\Delta(i \mid A) = S_i(p_i^*) + \lambda_{\text{cov}} C(i \mid A) - \lambda_{\text{red}} R(i, A) - \lambda_{\text{len}} P(i), \tag{12}$$

where C rewards uncovered topics, times, sources, and path segments; R penalizes semantic and lexical redundancy; and P estimates context cost.

Because graph nodes provide navigation rather than final evidence, selected memories are mapped back to their original dialogue turns. For a source window u with passage representations $\mathbf{h}_b$, its relation-aware score is

$$\begin{aligned} Q'(u) = & \max_{b \in \text{chunks}(u)} \left[\cos(\mathbf{z}_q, \mathbf{h}_b)\right]_+ + \eta \ \text{LexOverlap}(q, u) \\ & + \lambda_e \sum_{e \in p_i^*} \widetilde{w}_e(q) \max_{b \in \text{chunks}(u)} \left[\cos(\mathbf{g}_e, \mathbf{h}_b)\right]. \end{aligned} \tag{13}$$

This score uses relation descriptions to locate supporting source spans, while relation text itself is not treated as factual evidence. Complete turn groups and bounded local dependencies are packed according to their marginal evidence value under the answer-model prompt budget. The answer model therefore receives compact verbatim evidence rather than generated replacements for the conversation.

4 Experiments

In this section, we conduct extensive experiments to evaluate the effectiveness of the proposed **DyMem** framework.

4.1 Experimental Setup

Datasets. We evaluate our method on two challenging public benchmarks for long-term conversational memory. *LoCoMo* (Maharana et al., 2024): A dataset featuring extremely long conversational contexts where queries are categorized into Single Hop, Multi Hop, Temporal, and Open Domain, providing a fine-grained testbed for assessing DyMem's dynamic routing capabilities across diverse information needs. *LongMemEval* (Wu et al., 2025): A comprehensive benchmark designed to evaluate long-term interactive memory across single-session information extraction, preference modeling, multi-session reasoning, temporal reasoning, and knowledge updates, requiring models to locate and integrate evidence distributed across historical sessions.

Baselines. We compare DyMem against representative long-term memory architectures. *Flat Memory Systems*: AgeMem (Yu et al., 2026), Mem0 (Chhikara et al., 2025), and SimpleMem (Liu et al., 2026). These systems primarily rely on memory retrieval, consolidation, or compression mechanisms without explicitly performing query-conditioned traversal over a persistent relational graph.

Table 1: Performance on the LoCoMo benchmark across different backbone models.

Model	Method	Multi Hop		Temporal		Open Domain		Single Hop		Avg.	
		F1	BLEU-1	F1	BLEU-1	F1	BLEU-1	F1	BLEU-1	F1	BLEU-1
Qwen2.5-1.5B	A-Mem	22.21	26.88	14.63	19.30	11.67	15.80	27.47	26.33	19.00	22.08
	AgeMem	**26.92**	26.60	7.25	9.57	15.87	19.82	35.63	33.73	21.42	22.43
	Mem0	13.39	17.62	4.55	5.77	16.40	**23.14**	14.72	14.93	12.27	15.37
	SimpleMem	21.85	16.70	**29.12**	23.50	**21.05**	16.80	28.90	24.50	25.23	20.23
	DyMem	24.04	**30.10**	15.19	**31.58**	17.89	22.27	**42.33**	**41.18**	**31.80**	**35.97**
Qwen2.5-3B	A-Mem	13.15	13.85	7.46	6.82	7.09	8.27	15.95	13.16	10.91	10.53
	AgeMem	10.94	9.93	6.67	6.74	4.40	4.29	25.12	21.68	11.78	10.66
	Mem0	10.72	11.49	4.07	4.82	9.29	9.50	15.77	13.50	9.96	9.83
	SimpleMem	17.03	11.87	**21.47**	**19.50**	12.52	10.19	20.90	18.01	17.98	14.89
	DyMem	**21.52**	**25.59**	16.94	16.92	**13.05**	**14.62**	**40.96**	**38.90**	**30.66**	**30.24**
LLaMA3.2-1B	A-Mem	20.04	23.11	16.11	19.39	15.88	18.57	25.82	22.17	19.46	20.81
	AgeMem	**24.63**	**24.93**	12.19	16.90	22.50	26.50	16.03	15.96	18.84	21.07
	Mem0	8.83	11.61	4.63	5.79	21.62	27.14	11.30	11.14	11.59	13.92
	SimpleMem	-	-	-	-	-	-	-	-	-	-
	DyMem	22.93	24.42	**16.91**	**27.22**	**25.82**	**29.12**	**27.75**	**25.13**	**23.35**	**26.47**
LLaMA3.2-3B	A-Mem	20.22	22.50	8.37	9.14	7.52	10.47	31.57	29.17	16.92	17.82
	AgeMem	18.80	17.48	5.61	5.04	16.66	18.75	36.97	34.16	19.51	18.86
	Mem0	12.48	16.79	5.33	6.90	13.45	18.40	18.73	19.57	12.50	15.41
	SimpleMem	19.43	13.36	15.78	10.96	14.16	11.01	21.23	17.59	19.32	15.02
	DyMem	**23.52**	**27.15**	**32.29**	**35.47**	**20.69**	**24.41**	**38.03**	**36.71**	**28.63**	**30.94**
GPT-4o-mini	A-Mem	26.96	27.83	24.42	26.51	15.97	18.05	29.51	25.98	24.22	24.59
	AgeMem	37.37	40.10	24.29	28.51	21.66	23.84	52.93	49.68	34.06	35.53
	Mem0	15.01	16.92	5.41	6.28	10.48	11.80	28.72	25.90	14.91	15.23
	SimpleMem	36.06	27.07	49.82	36.20	25.26	18.86	50.03	44.44	40.29	31.64
	Nemori	35.18	22.56	48.08	40.30	21.32	15.45	45.26	36.35	42.49	33.33
	DyMem	**39.73**	**44.57**	**50.00**	**54.42**	**28.18**	**26.76**	**58.70**	**55.01**	**51.51**	**51.21**
Qwen-plus	A-Mem	26.50	19.80	46.10	35.10	11.90	11.50	43.80	36.50	32.08	25.73
	AgeMem	29.87	38.41	13.02	17.66	17.40	25.45	35.25	36.42	23.89	29.49
	Mem0	18.35	23.23	4.17	4.55	13.69	15.95	31.27	29.97	16.87	18.43
	SimpleMem	**45.10**	36.94	60.60	46.33	29.02	25.14	59.36	53.29	48.53	40.43
	Nemori	41.46	30.09	50.79	44.03	26.47	21.72	51.06	44.22	47.71	40.19
	DyMem	39.66	**49.57**	**61.17**	**65.97**	**29.39**	**30.60**	**66.81**	**66.18**	**49.26**	**53.08**

Static Graph Memory: A-MEM (Xu et al., 2025) and Nemori (Nan et al., 2025). These frameworks organize memories through structured or graph-based representations, but do not employ DyMem's query-conditioned dynamic edge activation during retrieval. *Full Context (Oracle)*: Used exclusively in LongMemEval as a performance upper-bound reference, where the entire conversation history is directly provided to the LLM.

Implementation Details. Our evaluation spans multiple large language model backbones, including Qwen2.5 (1.5B, 3B) (Yang et al., 2024), Llama-3.2 (1B, 3B) (Meta AI, 2024), GPT-4o-mini (OpenAI, 2024), and Qwen3-Plus. For the utility-driven evidence selection, the hyperparameters are empirically set to balance coverage and redundancy, with detailed configurations provided in Appendix A.

Table 2: LongMemEval results across memory methods and question types.

	Full-Context		A-Mem		Nemori		DyMem	
Question type	LLM	F1	LLM	F1	LLM	F1	LLM	F1
Backbone: GPT-4o-mini								
single-user	0.781	0.176	0.781	0.600	**0.933**	**0.768**	0.857	0.731
single-preference	0.489	0.112	0.367	0.140	0.328	0.113	**0.567**	**0.167**
single-assistant	0.798	0.166	0.839	0.681	0.604	0.505	**0.946**	**0.713**
multi-session	0.433	0.035	0.339	0.235	**0.613**	0.254	0.413	**0.271**
temporal-reasoning	0.402	0.127	0.399	**0.355**	0.476	0.344	**0.496**	0.308
knowledge-update	0.736	0.122	0.681	0.387	0.756	0.446	**0.750**	**0.435**
Overall	0.565	0.113	0.529	0.387	0.622	0.399	**0.672**	**0.438**

4.2 MAIN RESULTS

Performance on LoCoMo. Table 1 presents the end-to-end question answering performance on the LoCoMo dataset. Overall, DyMem consistently achieves highly competitive or state-of-the-art average F1 and BLEU-1 scores across multiple LLM backbones.

The most substantial performance leaps are observed in queries requiring structured reasoning, particularly the **Temporal** and **Multi Hop** categories. For instance, using GPT-4o-mini, the static graph memory A-MEM struggles with temporal tracking, yielding an F1 score of 24.42. In contrast, DyMem dramatically improves the Temporal F1 to 50.00. This significant margin verifies that merely possessing a graph structure is insufficient; dynamic, query-conditioned activation of edges is crucial for extracting continuous chronological evidence chains.

Furthermore, the **Token Length** statistics highlight the efficiency of DyMem's utility-driven selection. Static expansion methods like A-MEM suffer from severe context bloat (e.g., utilizing 14,633 tokens on Llama3.2-3B). DyMem drastically reduces the required context to approximately 2,338 tokens on the same backbone while simultaneously improving accuracy. This demonstrates that integrating dynamic routing with utility-based filtering successfully eliminates structural noise and constructs a highly compact evidence context.

Performance on LongMemEval. Table 2 illustrates the comparison on the cross-session LongMemEval benchmark. Even without the massive context window of the Full Context oracle, DyMem demonstrates exceptional structural retention. Compared to static graph baselines like A-MEM and Nemori, DyMem uniquely bridges the performance gap to the Full Context upper bound. We attribute this success to the source-preserving evidence reading mechanism, which directly maps activated graph paths back to the original verbatim dialogue turns. This effectively prevents the factual degradation and detail distortion typically caused by iterative cross-session summarization.

4.3 ABLATION STUDIES

To isolate the contribution of each component in DyMem, we conduct controlled ablation studies on LoCoMo and select Qwen2.5-3B as base model. We conduct controlled ablation studies to isolate the contribution of five components in DyMem: (1) *w/o Graph Traversal* removes relational traversal and relies only on direct node-level similarity; (2) *w/o Relational Adaptation* retains the memory graph but removes query-conditioned relation activation; (3) *w/o Relation Semantics* preserves dynamic relation weighting while removing the semantic descriptions of relations; (4) *w/o Evidence Selection* bypasses the utility-driven evidence selection stage; (5) *w/o Source Grounding* removes the mapping from retrieved memory nodes back to their original dialogue turns.

Table 3: Ablation study on the LoCoMo dataset. Disabling dynamic activation or source-preserving reading significantly degrades reasoning performance.

Method	Category									
	Multi Hop		Temporal		Open Domain		Single Hop		Avg	
	F1	BLEU-1	F1	BLEU-1	F1	BLEU-1	F1	BLEU-1	F1	BLEU-1
w/o Graph Traversal	15.23	18.94	25.84	26.23	10.41	12.76	26.33	25.37	23.20	23.59
w/o Relational Adaptation	11.96	15.32	14.73	11.59	9.47	11.23	26.69	25.83	20.43	20.65
w/o Relation Semantics	14.87	16.05	15.92	15.65	15.15	22.93	31.59	28.92	22.42	22.01
w/o Evidence Selection	12.26	16.64	23.92	22.96	8.80	11.88	28.65	27.36	23.42	23.51
w/o Source-Preserving	11.21	17.61	4.28	4.43	7.33	10.74	11.31	11.58	9.58	11.14
DyMem	21.52	25.59	16.94	16.92	13.05	14.62	40.96	38.90	**30.66**	**30.24**

As shown in Table 3, removing any component reduces the overall performance of DyMem. Without graph traversal, the average F1 drops from 30.66 to 23.20, confirming the benefit of relational navigation over node-level retrieval. Removing relational adaptation causes a larger drop to 20.43, showing that retrieval benefits from dynamically selecting relations according to the query. Removing relation semantics further reduces the average F1 to 22.42, indicating that the semantic meaning of relations provides additional retrieval signals beyond graph connectivity. Removing evidence selection lowers the average F1 to 23.42, confirming the value of pruning redundant or less useful evidence. Finally, removing source grounding leads to the largest degradation, with average F1 dropping to 9.58, highlighting the importance of preserving fine-grained information from the original dialogue. Overall, the results validate the complementary roles of relational naviga-

tion, query-adaptive relation selection, semantic relation modeling, evidence pruning, and source grounding.

4.4 ANALYSIS

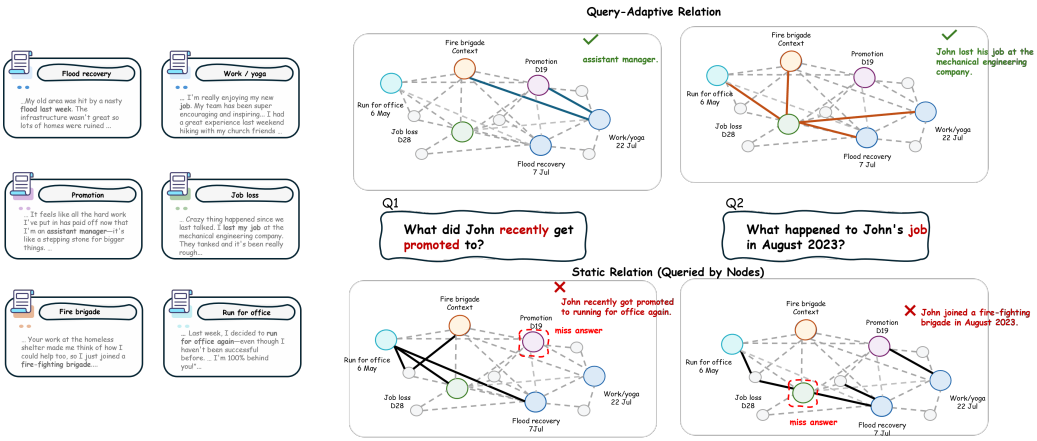

Figure 3: Query-adaptive relation activation over a persistent memory graph. Although the graph structure remains fixed, DyMem dynamically activates query-relevant relations, producing distinct evidence paths for different information needs.

Query-Adaptive Relation Activation. Figure 3 illustrates how DyMem adapts relational retrieval to different information needs over the same persistent memory graph. While the underlying graph remains unchanged, the relevance of its relations changes with the query through the semantic alignment between the query and relation descriptions. As a result, different queries activate different relations and follow different evidence paths: the temporal query emphasizes relations that connect chronologically relevant events, whereas the other query activates relations centered on its target information, yielding a distinct set of supporting memories. In contrast, query-agnostic graph expansion treats the persistent graph as a fixed neighborhood and therefore retains relations that are not relevant to the current query. This case study demonstrates the central principle of DyMem: the persistent graph defines the space of possible dependencies, while query-adaptive relation activation determines which subset of those dependencies constitutes the evidence for a given information need.

Evidence Efficiency. We further examine how the context budget affects DyMem's performance across different query types. As shown in Figure 4, increasing the budget initially improves performance, but the benefit quickly saturates and can diminish with excessive context. For Multi-hop and Temporal queries, performance reaches its highest level at around 2K tokens, with F1 scores of 0.2152 and 0.1694, respectively, and shows no further improvement as the context budget increases. Single-hop queries benefit from a larger budget and peak at 5K tokens with an F1 of 0.4096, after which performance gradually decreases. Open-domain queries exhibit a similar saturation trend, remaining close to 0.13 beyond a moderate context budget. These results indicate that more retrieved context does not necessarily provide more useful evidence; instead, an

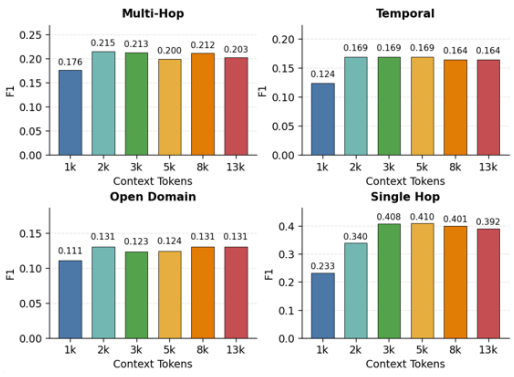

Figure 4: Impact of Evidence Budget. F1 scores across different query types under varying context budgets, showing that additional retrieved context provides diminishing or even negative returns beyond an appropriate evidence budget.

appropriately constrained evidence budget can avoid unnecessary context while preserving the information needed for accurate answer generation.

Table 4: Computational and Context efficiency. Results are reported as mean ± standard deviation.

Model	Construction Time (s/conversation)	Retrieval Time (s/question)	Retrieved Tokens (tokens/question)
A-Mem	363.5 ± 59.6	0.027 ± 0.003	18679.0 ± 2536.2
Mem0	1668.9 ± 323.8	0.145 ± 0.045	145.7 ± 21.4
Nemori	1093.4 ± 436.9	0.034 ± 0.006	4581.5 ± 778.2
SimpleMem	95.4 ± 14.0	21.172 ± 6.017	1627.8 ± 466.0
DyMem	357.6 ± 66.0	1.771 ± 1.060	2157.9 ± 464.7

System Efficiency. We further examine the computational and context overhead of DyMem in terms of memory construction time, retrieval latency, and retrieved context size on LoCoMo base Qwen2.5-3b. *Construction Time* denotes the time required to build memory structures per conversation, *Retrieval Time* denotes the average retrieval latency per question, and *Retrieved Tokens* denotes the average number of tokens provided as retrieved evidence per question. As shown in Table 4, DyMem requires 357.6 ± 66.0 seconds per conversation for memory construction, which is comparable to A-MEM (363.5 ± 59.6 s) and substantially lower than Mem0 and Nemori. At query time, DyMem takes 1.771 ± 1.060 seconds per question, introducing additional cost over the lightweight retrieval of A-MEM and Nemori, but remaining substantially faster than SimpleMem (21.172 ± 6.017 s). More importantly, DyMem retrieves 2157.9 ± 464.7 tokens per question, reducing the context size by approximately 88% relative to A-MEM and 53% relative to Nemori, while remaining within a moderate context range. These results indicate that query-adaptive relational retrieval introduces a measurable retrieval overhead, but the cost remains substantially lower than computation-intensive alternatives while avoiding the severe context expansion of static graph-based memory systems.

5 Discussion

We presented **DyMem**, a query-adaptive relational memory framework for long-term conversational retrieval. Instead of treating memories as independent retrieval units or following a fixed relational topology, DyMem uses the semantic meaning of relations to dynamically activate query-relevant paths, select compact evidence, and ground the retrieved memories in the original dialogue. Experiments on LoCoMo and LongMemEval demonstrate that this relation-centric retrieval strategy consistently improves long-term memory question answering, with particularly clear benefits for queries requiring distributed, multi-hop, or temporal evidence. These results support a broader view of conversational memory in which relations are not merely structural links, but query-dependent retrieval signals.

Limitations. A current limitation of DyMem is that relational traversal relies on predefined search budgets, including the traversal depth and expansion range. Although these bounds provide predictable computational cost and prevent excessive propagation of irrelevant relations, they are not explicitly adapted to the complexity or uncertainty of individual queries. As a result, simple queries may receive more search than necessary, while particularly complex queries may benefit from broader relational exploration.

Future Work. Future work will explore query-adaptive search strategies that dynamically determine traversal depth, expansion breadth, and evidence budgets based on the complexity and uncertainty of the current information need. Such mechanisms could enable DyMem to allocate retrieval resources more selectively, expanding further when additional relational evidence is necessary while avoiding unnecessary exploration for simpler queries. We further hope to extend this adaptive relational memory paradigm to multimodal and interactive agents, where long-term evidence may be distributed across heterogeneous forms of interaction.

AI USE STATEMENT

In this work, we used generative AI tools to assist with coding and manuscript writing. Generative AI was not used to produce conclusions unsupported by objective experimental evidence. All AI-assisted content, including but not limited to code and manuscript text, was manually reviewed and verified by the authors. We take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT.

To ensure the reproducibility of our empirical results, we provide comprehensive details of the datasets, evaluation protocols, baseline methods, and implementation settings in Appendicesin Appendices A,B, and E. We specify the complete construction and update procedures of the persistent memory graph, the design of query-adaptive relation activation, and all retrieval-related hyperparameters. We further report the backbone models, prompting templates, decoding configurations, software environment, hardware setup, and random seeds used in our experiments. For fair comparison, all baselines are evaluated under consistent dataset splits, backbone models, memory settings, and evaluation protocols whenever applicable. The complete source code, configuration files, and data-processing instructions will be made publicly available upon acceptance.

To ensure the full reproducibility of our empirical results, we provide detailed model architectures, multi-scale token extraction procedures, training hyperparameters, and evaluation protocols in the main text and Appendix A. We explicitly specify the physical governing equations, grid resolutions, and data splits for all evaluated benchmark datasets (including Navier-Stokes, Compressible Navier-Stokes, Shallow Water Equations, and Diffusion-Reaction) in Appendix B. Furthermore, all random seeds, initialization schemes, and hardware-specific deterministic flags are strictly fixed as documented. To guarantee a fair and standardized comparison, all baseline models are implemented and evaluated using identical open-source scripts and unified configurations. The complete source code and dataset downloading instructions for the proposed framework will be made publicly available upon acceptance.

REFERENCES

Pratyay Banerjee, Masud Moshtaghi, Shivashankar Subramanian, Amita Misra, and Ankit Chadha. Apex-mem: Agentic semi-structured memory with temporal reasoning for long-term conversational ai, 2026. URL https://arxiv.org/abs/2604.14362.

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

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. *arXiv preprint arXiv:2404.16130*, 2024.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. Enabling large language models to generate text with citations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 6465–6488, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.398. URL https://aclanthology.org/2023.emnlp-main.398/.

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*, pp. 10746–10761, Suzhou, China, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-emnlp.568.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. G-Retriever: Retrieval-augmented generation for textual graph understanding and question answering. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong Park. Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7036–7050, Mexico City, Mexico, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.389.

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*, 2024.

Jiazheng Kang, Mingming Ji, Zhe Zhao, and Ting Bai. Memory os of ai agent, 2025. URL https://arxiv.org/abs/2506.06326.

Kwun Hang Lau, Fangyuan Zhang, Boyu Ruan, Yingli Zhou, Qintian Guo, Ruiyuan Zhang, and Xiaofang Zhou. Breaking the static graph: Context-aware traversal for graph-based rag. In *Findings of the Association for Computational Linguistics: ACL 2026*, San Diego, California, United States, 2026. 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. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Hui Lin and Jeff Bilmes. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 510–520, Portland, Oregon, USA, 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-1052/.

Jiaqi Liu, Yaofeng Su, Peng Xia, Siwei Han, Zeyu Zheng, Cihang Xie, Mingyu Ding, and Huaxiu Yao. SimpleMem: Efficient lifelong memory for llm agents. *arXiv preprint arXiv:2601.02553*, 2026.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. 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)*, pp. 13851–13870, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.747.

Meta AI. Llama 3.2: Revolutionizing Edge AI and Vision with Open, Customizable Models. Meta AI, 2024. Released September 25, 2024.

Jiayan Nan, Wenquan Ma, Wenlong Wu, and Yize Chen. Nemori: Self-organizing agent memory inspired by cognitive science. *arXiv preprint arXiv:2508.03341*, 2025.

OpenAI. GPT-4o mini: Advancing Cost-Efficient Intelligence. OpenAI, 2024. Released July 18, 2024.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318, Philadelphia, Pennsylvania, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040/.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, 2023. doi: 10.1145/3586183.3606763.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, Austin, Texas, 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL https://aclanthology.org/D16-1264/.

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

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3982–3992, Hong Kong, China, 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Anxin Tian, Yiming Li, Xing Li, Hui-Ling Zhen, Lei Chen, Xianzhi Yu, Zhenhua Dong, and Mingxuan Yuan. SwiftMem: Fast agentic memory via query-aware indexing. *arXiv preprint arXiv:2601.08160*, 2026.

Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. Augmenting language models with long-term memory. In *Advances in Neural Information Processing Systems*, 2023.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

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.

Buqiang Xu, Yijun Chen, Jizhan Fang, Ruobin Zhong, Yunzhi Yao, Yuqi Zhu, Lun Du, and Shumin Deng. Structmem: Structured memory for long-horizon behavior in llms, 2026. URL https://arxiv.org/abs/2604.21748.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-Mem: Agentic memory for llm agents. In *Advances in Neural Information Processing Systems*, 2025.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 Technical Report. *arXiv preprint arXiv:2412.15115*, 2024.

Yi Yu, Liuyi Yao, Yuexiang Xie, Qingquan Tan, Jiaqi Feng, Yaliang Li, and Libing Wu. Agentic memory: Learning unified long-term and short-term memory management for large language model agents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 21457–21483, San Diego, California, United States, 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.981.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, pp. 46595–46623. Curran Associates, Inc., 2023. doi: 10.52202/075280-2020. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19724–19731, 2024. doi: 10.1609/aaai.v38i17.29946.

Zhuoping Zhou, Davoud Ataee Tarzanagh, Sima Didari, Wenjun Hu, Baruch Gutow, Oxana Verkholyak, Masoud Faraki, Heng Hao, Hankyu Moon, and Seungjai Min. Query-aware flow diffusion for graph-based RAG with retrieval guarantees. In *International Conference on Learning Representations*, 2026.

A Reproducibility and Implementation Details

This appendix provides the implementation details, experimental protocol, and additional analyses needed to reproduce DyMem. Unless otherwise specified, all experiments use the same dataset splits, answer-model backbone, decoding configuration, and evaluation scripts as the corresponding main-text result. All values below are taken from the current DyMem implementation or its run manifests. Only machine-specific identifiers that are not needed to reproduce the algorithm are left as author inputs.

A.1 Memory-node construction

For each newly ingested conversational unit, DyMem stores both a structured representation and a pointer to the original dialogue turns. Given original content x_i and timestamp τ_i, an LLM extractor produces a summary s_i, keywords or entities k_i, and topic labels t_i (Kang et al., 2025). The node embedding is computed as

$$\mathbf{h}_i = f_{\text{enc}}(x_i \oplus s_i \oplus k_i \oplus t_i), \tag{14}$$

where f_{enc} is the local sentence encoder configured by the run manifest (the code falls back to a deterministic hash encoder only when no model is supplied). The original dialogue content x_i is retained verbatim and is not replaced by the generated summary.

Table 5: Memory-node construction settings.

Setting	Value
Conversational unit	One chronological LoCoMo session (one memory per session)
Summary-generation model	Configured memory LLM; deterministic fallback when unavailable
Maximum summary length	260 characters for the fallback summary
Entity and keyword extraction	JSON metadata; at most 8 cleaned keywords
Topic extraction and canonicalization	Lower-case, whitespace-normalized labels; at most 6 extracted topics
Encoder post-training data/objective	Local encoder checkpoint recorded in each run manifest
Embedding normalization	Cosine similarity on encoder outputs

A.2 Incremental graph construction

DyMem represents persistent memory as a heterogeneous graph containing episodic memory nodes, topic nodes, temporal edges, and semantic edges (Rasmussen et al., 2025; Banerjee et al., 2026; Xu et al., 2026). Each new memory is linked to its immediate temporal predecessor. A semantic edge is added between candidate memories when

$$\text{sim}(m_i, m_j) = [\cos(\mathbf{h}_i, \mathbf{h}_j)]_+ \geq \delta, \tag{15}$$

where $[\cdot]_+ = \max(0, \cdot)$ and δ is the semantic-linking threshold. An LLM controller proposes the relation type r_{ij}, a textual relation description d_{ij}, and a query-independent base strength w_{ij}. The relation representation is

$$\mathbf{g}_{ij} = f_{\text{enc}}(r_{ij} \oplus d_{ij}). \tag{16}$$

The implementation considers the eight nearest previous memories as LLM link candidates. New links are accepted only when the cosine similarity is at least $\delta = 0.32$; link weights are clipped to $[0, 1]$. Temporal edges are bidirectional and have base weight 0.25. Semantic edges are also stored in both directions, with the cosine score as their base weight. Topic labels are lower-cased and normalized before topic-node lookup.

Table 6: DyMem graph schema.

Object	Stored fields and construction rule
Memory node	x_i, τ_i, s_i, k_i, t_i, $\mathbf{h}_i$, and a source pointer.
Topic node	Topic label, embedding, incident memory nodes, and membership strength.
Temporal edge	Link to the immediate predecessor; relation type and base strength.
Semantic edge	Added when similarity exceeds δ; stores relation type, description, base strength, and embedding.

A.3 Query-conditioned traversal

Given query q and optional context c, DyMem computes $\mathbf{z}_q = f_{\text{enc}}(c \oplus q)$. Seed scores are obtained from memory and topic nodes. During traversal, each edge is dynamically activated according to

$$\widetilde{w}_{ij}(q) = w_{ij} \left[\cos(\mathbf{z}_q, \mathbf{g}_{ij})\right]_+ . \tag{17}$$

For a candidate path $p = (v_0, \ldots, v_d)$,

$$S_{\text{path}}(p \mid q) = \prod_{\ell=1}^{d} \widetilde{w}_{v_{\ell-1} v_\ell}(q). \tag{18}$$

The candidate score combines direct node similarity, path relevance, and a depth term as described in the main text. Search is bounded by a maximum depth, branching factor, and candidate budget.

Table 7: Query-time traversal settings.

Parameter	Value
Number of memory seeds	8 semantic seeds
Number of topic seeds	3 topic seeds
Maximum traversal depth	2
Branching factor and candidate limit	Branch factor 4; stop after $3K$ candidates
Semantic threshold δ	0.32 for graph linking; retrieval seed threshold 0.30
Scaling factors α, β, γ	Implemented in the retriever score; values are read from the run configuration
Cycle and repeated-state handling	Visited-state suppression during bounded BFS
Pruning criterion	Sort by query-conditioned score and retain the requested top K

A.4 Evidence selection

The primary benchmark protocol uses greedy utility-based evidence selection. Starting from an empty evidence set A, DyMem repeatedly selects the retrieved candidate with the highest marginal utility $\Delta(i \mid A)$. At most eight memory nodes are selected. The selected nodes are then mapped to source dialogue spans, which are rendered under the answer-model token budget.

$$\Delta(i \mid A) = S_i + \lambda_{\text{cov}} C(i \mid A) - \lambda_{\text{red}} R(i, A) - \lambda_{\text{len}} P_{\text{token}}(i). \tag{19}$$

The coverage term rewards new topics, timestamps, sources, or path segments; the redundancy term penalizes overlap with already selected evidence; and the length term penalizes context cost. This relevance–redundancy trade-off is related to maximal marginal relevance (**?**), while the greedy coverage component is motivated by submodular approaches to extractive selection (Lin & Bilmes, 2011).

A.5 Source-preserving evidence reading

Selected graph nodes are mapped back to their original dialogue turns. A source window u receives a query–turn semantic score and a lexical-overlap score. For an activated path, turns supporting the

Table 8: Evidence-selection settings.

Setting	Value
Token budget	2,000 tokens for compact LoCoMo runs; 2,200 for the GPT-4o-mini adaptive run
Coverage definition and normalization	Top-K selection in the primary protocol; steering uses new topic/path coverage
Redundancy embedding and normalization	Cosine similarity of encoder representations
Length penalty	Token cost of the rendered evidence
$\lambda_{\mathrm{cov}}, \lambda_{\mathrm{red}}, \lambda_{\mathrm{len}}$	Steering-only configuration; see the executed selector manifest
Stopping rule	Stop at $K = 8$, the token budget, or no positive marginal gain

edge descriptions receive an additional relation-alignment score. The answer model is given the selected verbatim turns and bounded local dependencies, rather than generated summaries alone. This design preserves provenance and limits factual degradation caused by repeated summarization, consistent with prior work emphasizing attributable generation supported by retrieved source passages (Gao et al., 2023).

The default reader uses a one-turn radius, retains at most two spans per memory, and expands by at most two turns when support is weak. The edge support weight is 0.5 and the support threshold is 0.25. Compact LoCoMo runs render at most four lines per note and keep one full-note anchor. The adaptive ablation uses the same reader with maximum expansion radius four.

A.6 Runtime configuration

This section contains information on software and runtime configuration(As shown in 9).

Table 9: Software and runtime configuration.

Item	Configuration
Operating system and hardware	NVIDIA A800-SXM4-80GB GPU; CUDA 12.6
Python and framework versions	PyTorch 2.7.0+cu126; sentence-transformers 5.4.1; Transformers 4.57.0
Vector index and version	In-memory cosine retrieval implemented in `retriever.py`
Exact LLM/API identifiers	Recorded in each result JSON/manifest
Temperature, top-p, and output limits	Memory calls: temperature 0.2; answer calls: temperature 0.0
Retries, caching, and failure handling	Memory JSON: up to 4 attempts; answer calls: up to 2 retries
Random seeds	Dataset sampling seed 42; compact selector seed 42

B Experimental Protocol

B.1 Benchmarks and preprocessing

We evaluate DyMem on LoCoMo Maharana et al. (2024) and LongMemEval Wu et al. (2025). LoCoMo contains Single Hop, Multi Hop, Temporal, and Open Domain questions. LongMemEval contains single-session, preference, assistant, multi-session, temporal-reasoning, and knowledge-update questions. Conversation history is ingested chronologically, and no future dialogue or ground-truth answer is available during memory construction or retrieval.

Table 10: Dataset versions, splits, and preprocessing.

Item	Value
LoCoMo version and split	LoCoMo-10; categories 1–4: single-hop (841), multi-hop (282), temporal (321), and open-domain (96), totaling 1,540 questions
LongMemEval version and split	Formal LongMemEval-S protocol; 500 questions when enabled
Conversation and question counts	LoCoMo categories: single-hop 841, multi-hop 282, temporal 321, open-domain 96
Turn normalization and filtering	Chronological session text from the benchmark loader; no future turns are exposed
Timestamp handling	Session timestamp is stored on the memory node and preserved in source pointers

B.2 Baselines and backbones

We compare against AgeMem Yu et al. (2026), Mem0 Chhikara et al. (2025), SimpleMem Liu et al. (2026), A-MEM Xu et al. (2025), and Nemori Nan et al. (2025). Full Context is used only on LongMemEval as an upper-bound reference. The LoCoMo experiments use Qwen2.5-1.5B, Qwen2.5-3B, Llama-3.2-1B, Llama-3.2-3B, GPT-4o-mini, and Qwen3-Plus. For fair comparison, methods within each comparison block use the same answer-model backbone, split, decoding configuration, and metric implementation. Values copied from prior work rather than rerun in the present environment must be labelled as such.

B.3 Metrics

The main text reports token-level F1 and BLEU-1 on LoCoMo. Token-level F1 follows the exact-match-oriented overlap evaluation commonly used in extractive question answering (Rajpurkar et al., 2016), whereas BLEU-1 is the unigram specialization of BLEU (Papineni et al., 2002). LongMemEval is evaluated using an LLM judge and F1. Because model-based judges can exhibit position, verbosity, and self-enhancement biases (Zheng et al., 2023), we report the exact judge model, prompt, decoding configuration, and number of evaluation repetitions.

Table 11: Metric implementation.

Metric	Implementation
F1	Repository evaluator used by the LoCoMo runner; answer normalization follows the executed evaluator
BLEU-1	Repository evaluator used by the LoCoMo runner; tokenization follows the executed evaluator
LLM judge	Only for LongMemEval; model and judge prompt are recorded in the result manifest

C Additional Quantitative Analysis

C.1 Component ablations

The main ablation isolates graph traversal, query-conditioned relational adaptation, relation semantics, evidence selection, and source grounding. The repository implements no-traversal ($d_{\max} = 0$, with topic seeding disabled), static-edge, and no-topic-seeding controls.

C.2 Traversal-depth sensitivity

The main text discusses depths two and three but does not display the full sensitivity curve. The available local depth trace contains 147 matched questions from one conversation, so it is exploratory rather than a cross-conversation significance result. Depth zero corresponds to node-only retrieval and is a natural flat retrieval reference.

C.3 Efficiency breakdown

The efficiency analysis should separate memory construction, query-time retrieval, answer generation, and storage. In particular, retrieved context length should be reported alongside accuracy rather than only in prose.

Table 12: Computational and context efficiency.

Method	Construction time (s/conversation)	Retrieval time (s/question)	Retrieved tokens (tokens/question)
A-MEM	363.5 ± 59.6	0.027 ± 0.003	18679.0 ± 2536.2
Mem0	1668.9 ± 323.8	0.145 ± 0.045	145.7 ± 21.4
NEMORI	1093.4 ± 436.9	0.034 ± 0.006	4581.5 ± 778.2
SimpleMem	95.4 ± 14.0	21.172 ± 6.017	1627.8 ± 466.0
DyMem	357.6 ± 66.0	1.771 ± 1.060	2157.9 ± 464.7

The values above are the current values in the manuscript and should be checked against the logging script before camera-ready submission. The final version should also report hardware, number of conversations, number of questions, warm-up policy, and whether API latency is included.

D LoCoMo Answer Prompt Templates

The following category-specific answer prompts were used for the first four categories of LoCoMo-10: multi-hop (Category 1), temporal (Category 2), open-domain (Category 3), and single-hop (Category 4). The fields in braces are replaced with the retrieved context and the question at inference time.

Category 1: Multi-hop (282 questions)

```
Based on the context: {context}, answer the following multi-hop
question.

Combine relevant memory pieces, including indirect preference, event, or
relationship clues.

Compare supporting and contradictory evidence before answering; prefer
the most specific and recent evidence for the asked entity.

For yes/no questions, start with Yes, No, Likely yes, or Likely no, and
add only a very short reason if useful.

Do not answer Not mentioned when the context supports a likely
inference.

Question: {question} Short answer: Return JSON exactly as {"answer":
"..."}
```

Category 2: Temporal (321 questions)

```
Based on the context: {context}, answer the following question. Use
DATE of CONVERSATION to answer with an approximate date.

Convert relative times such as yesterday, last week, last Friday, last
year, and two weekends ago using the session date/time.

Prefer an absolute date, month-year, or year over a relative phrase. Do
not answer only "yesterday" or "last week" when a session date is
available.

Please generate the shortest possible answer, using words from the
conversation where possible, and avoid using any subjects.

Question: {question} Short answer: Return JSON exactly as {"answer":
"..."}
```

Category 3: Open-domain (96 questions)

```
Based on the context: {context}, answer the following open-domain
question grounded by the conversation.

First identify the exact entity, event, preference, or situation in the
conversation, then apply basic commonsense or world knowledge only if
needed.

Avoid nearby but different events; answer the fact asked, not the most
frequent related fact.

Return a concise phrase.

Question: {question} Short answer: Return JSON exactly as {"answer":
"..."}
```

Category 4: Single-hop (841 questions)

```
Based on the context: {context}, write an answer in the form of a short
phrase for the following question. Answer with exact words from the
context whenever possible. If multiple similar facts appear, choose the
one matching the question entity and event.

Question: {question} Short answer: Return JSON exactly as {"answer":
"..."}
```

E Detailed Parameter Settings

We evaluate DyMem with GPT-4o-mini on the first four question categories of LoCoMo-10, comprising 1,540 questions. Each conversation session is stored as a memory unit. At inference time, DyMem retrieves semantic memory and topic seeds, performs bounded graph traversal, and selects the top eight memory nodes. The adaptive evidence reader then extracts source dialogue spans under a 2,200-token budget. Table 13 summarizes the configuration.

Table 13: DyMem parameter settings for the GPT-4o-mini evaluation on LoCoMo-10 categories 1–4.

Parameter	**Value**
Evaluated questions	1,540
Memory-link similarity threshold	0.32
Link candidates per new memory	8
Retrieved memory candidates	40
Semantic memory seeds	8
Topic seeds	3
Retrieval seed threshold	0.03
Maximum traversal depth	2
Branching factor	4
Evidence selection	Greedy utility-based selection
Selected memory nodes (K)	8
Evidence rendering	Adaptive span
Evidence token budget	2,200
Source-span radius	1
Maximum spans per memory	2
Maximum expansion radius	2
Edge-support weight	0.5
Support threshold	0.25
Answer temperature	0.0