AutoViewMem: Self-Configuring Orthogonal Views for Conversational Long-Term Memory

Anonymous ACL submission

Abstract

Long-term memory is essential for large language model (LLM) agents to maintain consistency and personalization over extended interactions. Existing memory systems typically rely on fixed granularities or static schemas, but these designs struggle when heterogeneous information—such as preferences, events, constraints, and temporal updates—is entangled in a single representation, leading to semantic interference and unstable top-K retrieval. We present AutoViewMem, a data-driven framework that organizes long-term conversational memory into self-configuring, low-overlap semantic views. AutoViewMem discovers candidate views from interaction traces, selects a compact complementary view set, and uses these views to guide write-time structured extraction of provenance-grounded memories. This representation-first design reduces semantic entanglement within each memory space, enabling simple top-K similarity search to retrieve precise evidence without explicit routing or iterative retrieval. We further apply offline consolidation to improve memory compactness and consistency. Experiments on the LoCoMo benchmark show that AutoViewMem achieves competitive or improved performance over strong baselines on long-horizon question answering and reasoning, while maintaining an efficient and deployable inference pipeline.

1 Introduction

Large language model (LLM)-driven conversational agents are increasingly being deployed in settings such as long-term companionship, personalized assistance, and enterprise support (Zhong et al., 2024; Maharana et al., 2024; Packer et al., 2023). In these scenarios, interactions often span weeks or months and quickly exceed any finite context window. Even with long-context models, naively supplying the full history is rarely effective: long-horizon conversations exhibit frequent topic drift and a long-tail distribution of salient facts, so large amounts of low-relevance context increase both cost and noise, while crucial preferences, constraints, and event details remain buried in distant tokens and are not reliably utilized (Liu et al., 2024; Maharana et al., 2024; Wang et al., 2023).

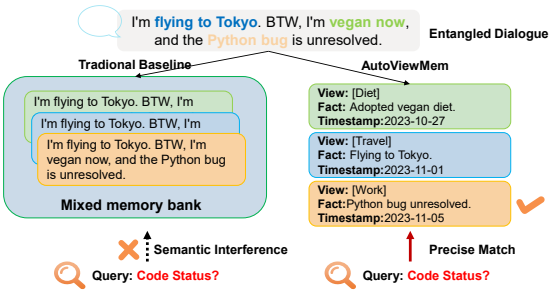

Figure 1: Case study comparing traditional baseline retrieval with AutoViewMem. While the baseline suffers from semantic interference by retrieving entangled irrelevant facts (diet, travel), AutoViewMem isolates the relevant [Work] view for the query "Code Status", enabling precise retrieval.

To address this limitation, recent systems commonly adopt external non-parametric memory under a write–retrieve–generate paradigm, retrieving historical information on demand (Lewis et al., 2020; Zhong et al., 2024; Packer et al., 2023; Wang et al., 2023; Chhikara et al., 2025). A common view is that the central design problem lies in memory granularity: overly fine-grained units fragment evidence and hurt recall, whereas overly coarse-grained units inject irrelevant content. This perspective has motivated methods based on multi-granularity association and adaptive retrieval, such as MemGAS (Xu et al., 2025a), as well as segmentation-and-compression pipelines such as SeCom (Pan et al., 2025).

However, granularity alone does not fully explain retrieval failures in long-term conversational memory. Long conversations are intrinsically het-

erogeneous: preferences, plans, facts, events, and temporal updates often coexist within the same dialogue span. When such mixed-content memories are embedded and indexed in a single representation space—whether as logs, summaries, or a single schema—semantically unrelated information can interfere during similarity search. This creates retrieval blind spots and forces systems into a persistent coverage–noise trade-off: increasing top-K improves coverage but amplifies noise, while reducing retrieval depth suppresses noise but misses critical evidence. Related observations have been reported in prior work on long-context utilization, conversational memory construction, and retrieval quality (Liu et al., 2024; Maharana et al., 2024; Pan et al., 2025; Xu et al., 2025a; Chhikara et al., 2025; Hu et al., 2026). These limitations suggest a more fundamental question: how should memory be organized before indexing so that heterogeneous semantics remain retrievable under diverse queries?

We propose **AutoViewMem** (Figure 2), a data-driven framework that organizes long-term memory into self-configuring, low-overlap semantic views. A view is conceptually orthogonal to granularity: granularity controls the size of a memory unit, whereas a view defines a semantic projection over the same interaction evidence. AutoViewMem discovers candidate views from interaction traces, selects a compact complementary view set, and uses these views to guide write-time structured extraction with timestamps and provenance. This representation-first design reduces semantic entanglement within each memory space and enables simple top-K similarity retrieval to construct precise contexts without explicit routing or iterative retrieval. We further apply an offline graph-based consolidation step to improve memory compactness and consistency.

We evaluate AutoViewMem on the LoCoMo benchmark (Maharana et al., 2024). Results show that AutoViewMem achieves competitive or improved performance over representative long-term memory baselines, including Mem0 (Chhikara et al., 2025), MemGAS (Xu et al., 2025a), LightMem (Fang et al., 2025), SimpleMem (Liu et al., 2026), and A-MEM (Xu et al., 2025b), across long-horizon question answering and reasoning tasks, while maintaining an efficient and deployable inference pipeline.

Our main contributions are:

- **A new perspective on long-term conversational memory.** We argue that retrieval failures in long-horizon conversations are not only a granularity problem, but also a memory organization problem caused by semantic interference in mixed-content representations.

- **A self-configuring multi-view memory framework.** We introduce AutoViewMem, which automatically discovers and selects complementary low-overlap semantic views from streaming interactions to organize memory before indexing.

- **Simple retrieval enabled by write-time organization.** We show that semantic view-based structuring makes standard top-K similarity retrieval effective in long-horizon settings, yielding competitive performance with a simple and deployable inference pipeline.

2 Related Work

2.1 Retrieval and Memory Access

Retrieval is a core mechanism for grounding LLM outputs in external evidence. Sparse retrievers such as BM25 remain strong lexical baselines (Robertson and Zaragoza, 2009), while dense retrieval and RAG-style pipelines make semantic matching a standard interface for external memory access (Karpukhin et al., 2020; Lewis et al., 2020). Contrastive retrievers and recent embedding models further strengthen semantic search in long-context settings (Izacard et al., 2022; **?**).

Our work is complementary to retrieval advances: rather than designing a new retriever, we study how memory should be represented before indexing. In long-horizon dialogue, retrieval quality depends not only on the retriever itself, but also on whether stored items preserve semantically focused evidence instead of entangled mixed-content spans (Liu et al., 2024; Maharana et al., 2024; Pan et al., 2025; Xu et al., 2025a).

2.2 Long-Term Memory for LLM Agents

A growing body of work equips LLM agents with long-term memory to support personalization, coherence, and task continuity beyond the context window. Prior systems study memory paging, external memory modules, time-aware updating, hierarchical storage, and scalable consolidation (Packer et al., 2023; Wang et al., 2023; Zhong

et al., 2024; Chhikara et al., 2025; Kang et al., 2025; Xu et al., 2025b; Fang et al., 2025; Liu et al., 2026; Zhang et al., 2026; Li et al., 2026a).

These systems mainly focus on how memory is stored, consolidated, or accessed at scale. By contrast, AutoViewMem targets a representation-first question: how to organize heterogeneous conversational information before indexing so that standard retrieval remains stable under diverse long-horizon queries.

2.3 Structured Memory and Schema Induction

Our work is also related to schema induction and structured memory construction. Prior work on schema induction, information extraction, and knowledge base population studies how reusable structures can be induced from unstructured text (Chambers and Jurafsky, 2010; Li et al., 2023). Recent work further shows that slot structures can be induced directly from dialogue without manual schema design (Finch et al., 2024). In conversational memory, structured representations such as entity–attribute records and preference frames are often more reusable than raw logs under topic drift and long interaction horizons (Zhong et al., 2024; Pan et al., 2025).

Unlike approaches that rely on fixed templates or manually specified fields, AutoViewMem induces lightweight semantic views online from interaction traces and uses them to guide write-time structured extraction, enabling adaptive memory organization while keeping retrieval simple.

2.4 Benchmarks

Benchmarking long-horizon conversational memory has also advanced rapidly. LoCoMo provides a controlled testbed for very long-term memory in dyadic dialogue (Maharana et al., 2024), while newer benchmarks extend toward more realistic multi-party, cross-topic, and temporally evolving settings (Hu et al., 2026; Li et al., 2026b). These benchmarks highlight the need for memory systems that remain robust under heterogeneous evidence, topic interleaving, and long-horizon retrieval.

3 Methodology

We propose AutoViewMem (Figure 2), a two-stage memory architecture consisting of (i) an online view induction-and-writing pipeline and (ii) an offline consolidation mechanism for compactness and semantic consistency. Online, the system incrementally induces lightweight reusable schemas—which we call *views*—from a conversation stream and uses them to guide structured memory writing. Offline, it deduplicates and consolidates extracted memories through graph clustering and LLM-based semantic adjudication, rewriting redundant items into canonical records without adding an extra summary layer.

Views and memory items. A view v is a lightweight schema with (a) a name (e.g., *Diet Preferences*), (b) a set of slots or fields (e.g., *likes*, *dislikes*, *constraints*), and (c) an extraction template that instructs the LLM to populate those fields from dialogue. This design follows prior observations that structured representations improve the stability and reusability of conversational memory (Pan et al., 2025; Zhong et al., 2024). We use *orthogonal* in a pragmatic sense: views are intended to be complementary and low-overlap under semantic similarity, rather than strictly orthogonal in the linear-algebraic sense.

Formally, for each user u, we observe a conversation stream $\mathcal{S}_u = \{m_1, \ldots, m_T\}$. The system maintains (1) an active view set $\mathcal{V}_u^*$ and (2) a structured memory store $\mathcal{M}_u$, where each memory item is a fine-grained record grounded in specific dialogue evidence.

Why multi-view organization helps. Long-horizon dialogue often mixes heterogeneous semantics, such as preferences, events, and temporal updates, within the same interaction stream. Organizing memory under multiple focused views reduces semantic entanglement in stored items and makes standard top-K retrieval more stable under diverse queries.

3.1 Online Adaptive Memory Learning

The online learner follows a Divergence–Convergence strategy (Algorithm 1 in Appendix A) with three phases: (i) **View Discovery**, which accumulates weak signals from the stream without immediately writing persistent memory; (ii) **View Convergence**, which aggregates noisy candidates into a compact active view set; and (iii) **Schema-guided Extraction**, which writes structured memories conditioned on the selected views.

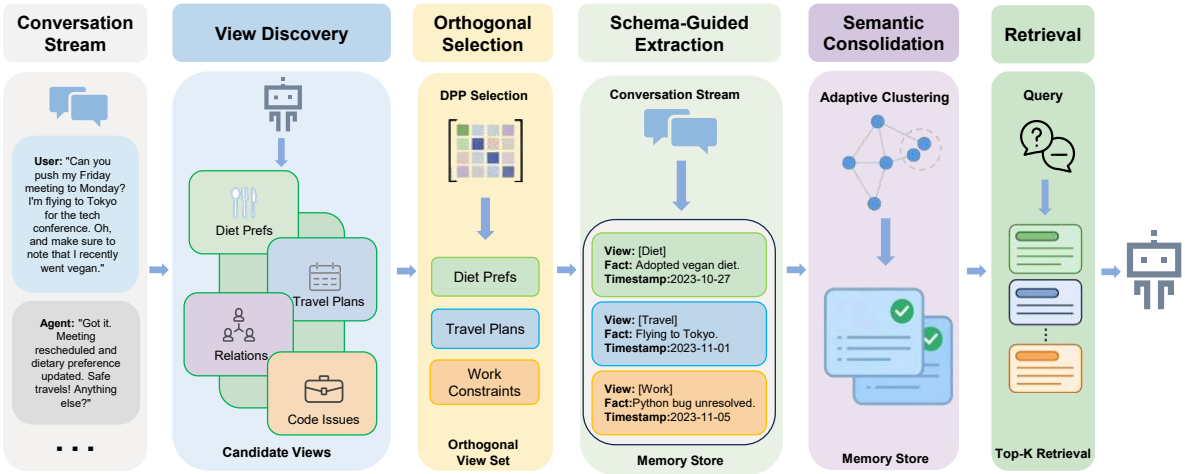

Figure 2: The AutoViewMem architecture. The system processes a conversation stream through online view discovery, selects an orthogonal view set via DPP, and performs schema-guided memory extraction and retrieval to provide context for the LLM agent.

3.1.1 Phase 1: View Discovery

The conversation arrives as an ordered sequence of dialogue chunks. Instead of directly writing each chunk into persistent memory, the system first accumulates weak signals such as repeated entities, stable preferences, and recurring constraints. Periodically, every N chunks, an LLM proposes a candidate view set $\tilde{\mathcal{V}}_u = \{v_1, \ldots, v_K\}$ from the recent interaction trace (prompts and implementation details are deferred to the appendix). Each candidate view contains a name, slots, and an extraction template.

3.1.2 Phase 2: View Convergence

The candidate set $\tilde{\mathcal{V}}_u$ is noisy and potentially redundant. We therefore first cluster and deduplicate semantically similar candidates to form an aggregated pool $\mathcal{V}_u$. We then select a compact and diverse active set $\mathcal{V}_u^*$ using a Determinantal Point Process (DPP) (Kulesza and Taskar, 2012; Chen et al., 2018). For each view v_i, we compute an ℓ_2-normalized embedding z_i and define the L-ensemble kernel

$$L_{ij} = q_i \, (z_i^\top z_j) \, q_j, \tag{1}$$

where q_i is a uniform quality term. The DPP favors subsets with large $\det(L_S)$, encouraging complementary views with low semantic redundancy. The selected subset $\mathcal{V}_u^*$ serves as the schema prior for memory extraction.

3.1.3 Phase 3: Adaptive Extraction

Given $\mathcal{V}_u^*$, the system writes long-term memory in a schema-guided manner. For each dialogue chunk x, the LLM performs targeted extraction conditioned on the active views, producing view-tagged items with structured fields, timestamps, and provenance pointers.

At query time, given a user query q, we perform standard top-K dense retrieval over the memory items (Lewis et al., 2020; Karpukhin et al., 2020). Retrieved items are merged under a token budget with lightweight deduplication before being passed to the LLM. Because memories are organized into complementary views, simple top-K retrieval is often sufficient to return precise, low-redundancy context without explicit routing or iterative retrieval.

3.2 Offline Memory Consolidation

Online extraction prioritizes responsiveness and recall, while offline consolidation improves compactness and semantic consistency.

3.2.1 Exact deduplication and graph clustering

We first remove exact duplicates by hashing canonicalized text and/or structured representations. For duplicate items, we keep a single record and merge their metadata, including view tags, occurrence counts, and provenance pointers.

We then construct an undirected similarity graph $G = (V, E)$ over memory items, where an edge (i, j) exists if $\cos(e_i, e_j) \geq \tau$ (default

$\tau = 0.8$). Connected components define candidate consolidation batches. If a component exceeds a maximum batch size, we recursively increase τ (e.g., $0.80 \rightarrow 0.999$) to split it into tighter sub-clusters for finer-grained consolidation.

3.2.2 LLM-driven semantic consolidation

Embedding similarity alone is insufficient for safe merging. For each cluster, an LLM rewrites the grouped items into one or more canonical memory records according to three cases: (1) **Containment**, where one item subsumes another; (2) **Complementarity**, where non-overlapping facts can be merged; and (3) **Independence despite similarity**, where similar items should remain separate because they express different facts (Zhong et al., 2024). The resulting canonical records retain merged provenance and replace redundant variants in the memory store, preventing unbounded memory growth over long interaction horizons.

4 Experiments

We evaluate AutoViewMem on LoCoMo (Maharana et al., 2024) against representative long-term conversational memory systems. We focus on three questions: whether multi-view write-time organization improves long-horizon QA, which components contribute most to the gains, and whether such organization improves retrieval quality under a fixed budget.

4.1 Experimental Setup

Benchmark and metrics. We use LoCoMo (Maharana et al., 2024), a benchmark for long-term conversational memory with conversations of 200–400 turns and 1,540 evaluation questions spanning Multi-Hop, Temporal, Open Domain, and Single Hop reasoning. Following the benchmark protocol, we report BLEU-1 (B1), F1, and LLM Judge (J), and provide both category-wise and overall results (additional dataset statistics are given in Appendix E).

Retrieval evaluation. Because AutoViewMem may produce multiple view-specific memory items grounded in the same dialogue evidence, item-level retrieval can overestimate performance by rewarding redundant hits. We therefore evaluate retrieval at the evidence level using provenance pointers stored with each memory item. For each query, we retrieve top-K_0 items from each speaker-specific memory store ($K_0 = 30$), merge them, and deduplicate by evidence identity before computing Recall@K and NDCG@K. We emphasize NDCG in particular, since evidence-level deduplication can make Recall less informative in multi-view settings.

Backbones and baselines. We evaluate with two LLM backbones, Qwen3-8B and GPT-4o-mini, used for both memory processing and response generation. We compare against representative memory systems, including A-mem (Xu et al., 2025b), Mem0 (Chhikara et al., 2025), MemGAS (Xu et al., 2025a), LightMem (Fang et al., 2025), and SimpleMem. All methods use the same embedding backbone and the same LLM backend within each setting; additional implementation details are provided in Appendix D.

4.2 Main Results

Table 1 reports end-to-end QA results on LoCoMo. Under Qwen3-8B, AutoViewMem achieves the best overall F1 among the compared memory-augmented methods. Under GPT-4o-mini, it remains competitive overall and achieves the best LLM Judge score.

These results indicate that organizing memory at write time into complementary semantic views improves long-horizon QA while retaining a simple retrieval pipeline. Notably, AutoViewMem uses standard top-K dense retrieval rather than iterative retrieval or routing-based control.

4.3 Efficiency and Ablation Analysis

Performance–cost trade-off. Figure 3 compares answer quality and average token cost across methods. AutoViewMem achieves a favorable performance–cost trade-off: it improves answer quality over lower-cost baselines such as Mem0 and LightMem, while avoiding the substantially higher cost of more retrieval-intensive systems such as SimpleMem. This supports our design intuition that better write-time organization increases the usefulness of retrieved context without requiring expensive retrieval-time control.

Ablation settings. We conduct ablations on the Qwen3-8B backbone: (1) *Single-view Structured*, which removes multi-view organization; (2) *Multi-view Non-orthogonal*, which keeps multiple views but removes the low-overlap constraint; (3) *w/o DPP Selection*, which replaces DPP-based selection with random view selection; and (4) *w/o*

Table 1: Main Results Comparison. We report BLEU (B1), F1, and J (LLM Judge Score) across different categories and overall. The best results among memory-augmented methods are highlighted in **bold**.

Method	MultiHop			Temporal			OpenDomain			SingleHop			Overall		
	B1	F1	J	B1	F1	J	B1	F1	J	B1	F1	J	B1	F1	J
Qwen3-8B															
Full History	0.2124	0.2646	0.8333	0.2875	0.3841	0.6199	0.1381	0.1918	0.6146	0.2759	0.3699	0.8811	0.2581	0.3425	0.8013
A-mem	0.2067	0.2738	0.4929	0.2200	0.3062	0.4143	0.0958	0.1272	0.3646	0.3700	0.4197	0.6278	0.2917	0.3511	0.5422
MemGAS	0.0672	0.0995	0.4078	0.0614	0.0923	0.2835	0.0437	0.0714	0.3750	0.0816	0.1239	0.4293	0.0724	0.1096	0.3916
LightMem	0.1875	0.2713	0.6241	0.2054	0.2437	0.4268	0.0954	0.1396	**0.5312**	0.2028	0.2432	0.4780	0.1939	0.2420	0.4974
Mem0	0.2131	0.3164	0.5426	0.3765	0.5074	0.5670	0.1544	0.1859	0.4375	0.3377	0.3983	0.6492	0.3115	0.3928	0.5994
SimpleMem	**0.2708**	0.3459	**0.7553**	0.3653	0.4398	0.4206	0.1697	0.2063	0.5000	0.3714	0.4211	**0.7539**	0.3392	0.3978	0.6688
AutoViewMem	0.2516	**0.3560**	0.6135	**0.4295**	**0.5492**	**0.6231**	**0.1741**	**0.2161**	0.4375	**0.4092**	**0.4697**	0.7360	**0.3699**	**0.4496**	**0.6714**
GPT-4o-mini															
Full History	0.2444	0.3259	0.6454	0.2605	0.3466	0.5296	0.1766	0.2286	0.6146	0.3443	0.4880	0.8549	0.2981	0.4126	0.7338
A-mem	0.1794	0.2363	0.4504	0.3252	0.3802	0.5514	0.0802	0.1020	0.2917	0.3522	0.4080	0.6611	0.2980	0.3517	0.5766
MemGAS	0.0958	0.1337	0.4574	0.0715	0.1035	0.2928	0.0812	0.1162	0.4688	0.1050	0.1573	0.5351	0.0949	0.1392	0.4662
LightMem	0.1548	0.2305	0.3865	**0.4249**	0.4814	0.6075	0.1793	0.2286	0.4062	0.3643	0.4201	0.6147	0.3270	0.3862	0.5584
Mem0	0.2073	0.3040	0.5000	0.3842	0.4696	0.5296	0.1786	0.2410	0.5312	0.3466	0.4142	0.6504	0.3115	0.3948	0.5903
SimpleMem	**0.2589**	**0.3508**	**0.7005**	0.3574	**0.4990**	0.5681	0.1202	0.1723	0.4908	**0.4276**	**0.4821**	0.8035	**0.3629**	**0.4423**	0.7161
AutoViewMem	0.2191	0.2934	0.6667	0.3322	0.4618	**0.6480**	**0.1868**	0.2316	**0.6771**	0.3922	0.4525	**0.8430**	0.3352	0.4116	**0.7597**

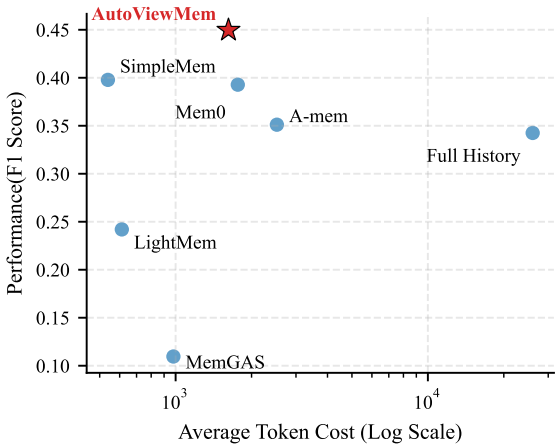

Figure 3: Performance–cost trade-off on LoCoMo. The x-axis shows average token cost (log scale), and the y-axis shows overall F1.

Graph Consolidation, which removes offline consolidation.

Ablation results. Table 2 shows that removing multi-view organization causes the largest drop in end-to-end performance, indicating that structured extraction alone is insufficient when heterogeneous conversational information is forced into a single semantic space. The gap between the full model and *Multi-view Non-orthogonal* further shows that merely increasing the number of views is not enough; low-overlap view selection is important for keeping retrieved context focused. Removing DPP selection or graph consolidation also degrades answer quality, though more moderately.

Some ablations achieve higher Recall@K than the full model while still producing worse final answers. This suggests that downstream quality depends not only on evidence coverage, but also on whether relevant evidence is ranked cleanly and presented with limited redundancy.

4.4 Retrieval Analysis

Retrieval stability and ranking quality. We next examine retrieval quality under a fixed budget. Our hypothesis is that multi-view write-time organization reduces semantic interference, making standard top-K dense retrieval more stable. Because evidence-level deduplication can suppress apparent recall gains in a multi-view system, we emphasize ranking-sensitive metrics such as NDCG and similarity-based separation diagnostics.

Similarity margin. To quantify ranking stability, we compute a per-query similarity margin

$$\Delta(q) = \max_{e \in G(q)} s(q, e) - \max_{e \notin G(q)} s(q, e), \quad (2)$$

where $s(\cdot, \cdot)$ is the retrieval similarity score and $G(q)$ is the set of relevant evidence units for query q. Larger margins indicate better separation between the best positive and the hardest negative. As shown in Figure 4, the full model shifts the mean margin from negative to positive relative to *Single-view Structured* and also lowers the no-positive rate (Full: 14.16% vs. Single-view:

Table 2: Ablation results on Qwen3-8B. We report generation metrics (overall B1 and F1) and evidence-level retrieval diagnostics under a fixed retrieval budget.

Method	Generation		Retrieval Diagnostics (Evidence-level)					
	B1	F1	Recall@3	Recall@5	Recall@10	NDCG@3	NDCG@5	NDCG@10
AutoViewMem (Full)	**0.3699**	**0.4496**	**0.4873**	**0.5549**	**0.6410**	**0.4500**	**0.4779**	**0.5083**
Single-view Structured	0.2850	0.3688	0.4391	0.5074	0.5908	0.4024	0.4304	0.4594
Multi-view Non-orthogonal	0.3340	0.4263	0.4438	0.5017	0.5890	0.4214	0.4440	0.4744
w/o DPP Selection	0.3285	0.4187	0.4650	0.5260	0.6042	0.4382	0.4632	0.4907
w/o Graph Consolidation	0.3345	0.4248	0.4660	0.5055	0.5667	0.4468	0.4617	0.4829

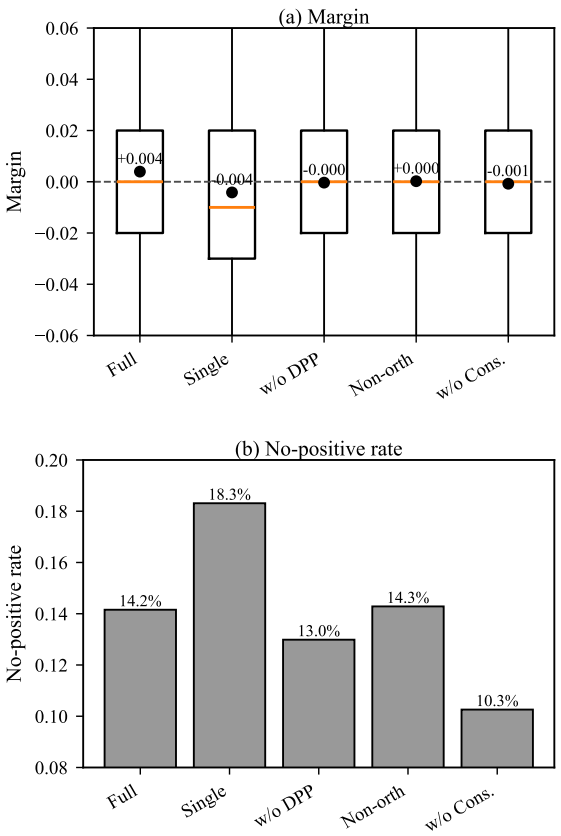

Figure 4: Similarity margin and no-positive rate across retrieval variants. Larger margins and lower no-positive rates indicate more stable evidence separation under a fixed retrieval budget.

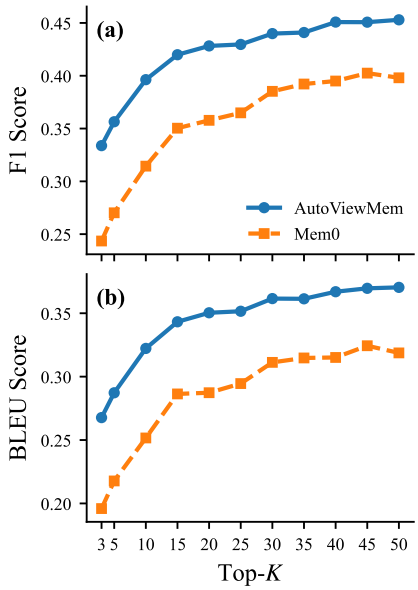

Figure 5: Effect of retrieval budget on downstream performance. We vary the top-K limit and report mean F1 and BLEU on LoCoMo, comparing AutoViewMem with Mem0.

18.31%). The *Multi-view Non-orthogonal* variant remains near zero, indicating that low-overlap view selection is important for stable positive–negative separation.

These findings are consistent with Table 2: higher Recall@K alone does not necessarily translate into better downstream answers, whereas cleaner ranking and lower redundancy produce a more useful top-K context.

Sensitivity to retrieval budget. We further vary the top-K limit used for context construction and compare AutoViewMem with Mem0. Figure 5 shows that both methods improve as K increases and then gradually saturate, while AutoViewMem remains consistently better, especially under smaller budgets. This suggests that semantically focused views improve retrieval efficiency by reducing noise in the retrieved context.

Case study. Figure 1 shows a qualitative example. A single-space baseline retrieves multiple loosely related memories from semantically entangled topics, whereas AutoViewMem retrieves the view aligned with the query intent and provides a cleaner context. Additional examples are deferred to Appendix B.

5 Conclusion

We presented AutoViewMem, a self-configuring multi-view framework for long-term conversational memory. Beyond memory granularity, our work highlights memory organization as a key

factor in long-horizon retrieval: storing heterogeneous dialogue content in a single representation space can introduce semantic interference and destabilize retrieval.

By organizing the same interaction stream into complementary low-overlap semantic views, AutoViewMem supports provenance-grounded write-time structuring and effective top-K retrieval with a simple inference pipeline. An additional offline consolidation stage further improves compactness and consistency.

Results on LoCoMo show that this representation-first design achieves competitive or improved performance against strong baselines. Our analyses further suggest that low-overlap multi-view organization is the main source of improvement, enabling more stable retrieval and stronger long-horizon reasoning without complex retrieval-time control.

6 Limitations

AutoViewMem has several limitations that should be addressed in future work.

Dependence on the underlying LLM. AutoViewMem relies on the underlying LLM in multiple stages, including view discovery, schema-guided extraction, and offline semantic consolidation. As a result, memory quality may vary across backbone models, prompting strategies, and decoding settings.

Periodic rather than fully online adaptation. View convergence is performed periodically from buffered interaction traces rather than updated fully online. While this design keeps the system practical, it may slow adaptation when user behavior or topic distributions shift rapidly.

Risk in semantic consolidation. Offline semantic consolidation introduces a risk of over-merging or imperfect rewriting when similar memories should remain distinct, especially in cases involving subtle temporal changes, fine-grained factual differences, or partially conflicting evidence.

Benchmark scope. Our evaluation is conducted mainly on LoCoMo. Although this benchmark is suitable for studying long-horizon conversational memory, broader validation on more diverse settings, such as multi-party, highly dynamic, or safety-critical environments, is still needed.

Privacy and safety. Persistent storage of conversational memory raises privacy and safety concerns. Although we retain provenance and apply PII filtering, stronger mechanisms for deletion, access control, and user-governed retention would be necessary in real-world deployments.

References

Nathanael Chambers and Dan Jurafsky. A database of narrative schemas. In *LREC*, 2010.

Laming Chen, Guoxin Zhang, and Hanning Zhou. Fast greedy map inference for determinantal point process to improve recommendation diversity. In *NeurIPS*, 2018.

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.

Jizhan Fang, Xinle Deng, Haoming Xu, Ziyan Jiang, Yuqi Tang, Ziwen Xu, Shumin Deng, Yunzhi Yao, Mengru Wang, Shuofei Qiao, Huajun Chen, and Ningyu Zhang. Lightmem: Lightweight and efficient memory-augmented generation. *arXiv preprint arXiv:2510.18866*, 2025.

James D. Finch, Boxin Zhao, and Jinho D. Choi. Transforming slot schema induction with generative dialogue state inference. In *SIGDIAL*, 2024.

Chuanrui Hu, Tong Li, Xingze Gao, Hongda Chen, Yi Bai, Dannong Xu, Tianwei Lin, Xiaohong Li, Yunyun Han, Jian Pei, and Yafeng Deng. Evaluating long-horizon memory for multi-party collaborative dialogues. *arXiv preprint arXiv:2602.01313*, 2026.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*, 2022.

Jiazheng Kang, Mingming Ji, Zhe Zhao, and Ting Bai. Memory os of ai agent. *arXiv preprint arXiv:2506.06326*, 2025.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *EMNLP*, 2020.

Alex Kulesza and Ben Taskar. Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*, 2012.

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 *NeurIPS*, 2020.

Kai Li, Xuanqing Yu, Ziyi Ni, Yi Zeng, Yao Xu, Zheqing Zhang, Xin Li, Jitao Sang, Xiaogang Duan, Xuelei Wang, Chengbao Liu, and Jie Tan. Timem: Temporal-hierarchical memory consolidation for long-horizon conversational agents. *arXiv preprint arXiv:2601.02845*, 2026a.

Sha Li, Heng Ji, and Jiawei Han. Open-domain hierarchical event schema induction by incremental prompting and verification. In *ACL*, 2023.

Yifei Li, Weidong Guo, Lingling Zhang, Rongman Xu, Muye Huang, Hui Liu, Lijiao Xu, Yu Xu, and Jun Liu. Locomo-plus: Beyond-factual cognitive memory evaluation framework for llm agents. *arXiv preprint arXiv:2602.10715*, 2026b.

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.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 2024.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of llm agents. In *ACL*, 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.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Xufang Luo, Hao Cheng, Dongsheng Li, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Jianfeng Gao. On memory construction and retrieval for personalized conversational agents. *ICLR*, 2025.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 2009.

Weizhi Wang, Li Dong, Haibo Bao, Shaohan Huang, Li Zhang, Xiaoye Ma, and Furu Wei. Augmenting language models with long-term memory. In *NeurIPS*, 2023.

Derong Xu, Yi Wen, Pengyue Jia, Yingyi Zhang, Wenlin Zhang, Yichao Wang, Huifeng Guo, Ruiming Tang, Xiangyu Zhao, Enhong Chen, and Tong Xu. Towards multi-granularity memory association and selection for long-term conversational agents. *arXiv preprint arXiv:2505.19549*, 2025a.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*, 2025b.

Ningning Zhang, Xingxing Yang, Zhizhong Tan, Weiping Deng, and Wenyong Wang. Himem: Hierarchical long-term memory for llm long-horizon agents. *arXiv preprint arXiv:2601.06377*, 2026.

Wanjun Zhong et al. Memorybank: Enhancing large language models with long-term memory. *AAAI*, 2024.

A Additional Experimental Details

Details regarding hyperparameters (e.g., DPP kernel parameters, graph clustering thresholds) and model configurations are provided here.

A.1 Algorithm Implementation Details

We provide the pseudo-code for the core algorithms used in AutoViewMem: Online View Learning (Algorithm 1) and Recursive Graph Consolidation (Algorithm 2).

Algorithm 1 Online View Learning Strategy

Require: Conversation Stream $\mathcal{S}_u$, Buffer Threshold N, View Count K
Initialize active view set $\mathcal{V}_u^* \leftarrow \emptyset$, candidate pool $\mathcal{P} \leftarrow \emptyset$, buffer $\mathcal{B} \leftarrow \emptyset$
for each incoming chunk x_t in $\mathcal{S}_u$ **do**
  Append x_t to $\mathcal{B}$
  if $|\mathcal{B}| \geq N$ **then**
    Phase 1: Discovery
    Generate candidates $C \leftarrow \text{LLM}(\mathcal{B}, \text{Prompt}_{\text{discovery}})$
    $\mathcal{P} \leftarrow \mathcal{P} \cup C$
    Clear $\mathcal{B}$
    Phase 2: Convergence (DPP Selection)
    Embed candidates: $Z \leftarrow \text{Encoder}(\mathcal{P})$
    Normalize Z: $z_i \leftarrow z_i / \|z_i\|$
    Compute Kernel: $L \leftarrow ZZ^\top$
    $S \leftarrow \emptyset$
    for $j = 1$ to K **do**
      $i^* \leftarrow \arg\max_{i \in \mathcal{P} \setminus S} \det(L_{S \cup \{i\}})$
      $S \leftarrow S \cup \{i^*\}$
    end for
    $\mathcal{V}_u^* \leftarrow \emptyset$
    for each $s \in S$ **do**
      Find neighbors $\mathcal{N}_s$ of s in $\mathcal{P}$
      $v_{new} \leftarrow \text{LLM}(\mathcal{N}_s, \text{Prompt}_{\text{summary}})$
      $\mathcal{V}_u^* \leftarrow \mathcal{V}_u^* \cup \{v_{new}\}$
    end for
  end if
  Phase 3: Extraction
  Extract memories $M_t \leftarrow \text{LLM}(x_t, \mathcal{V}_u^*, \text{Prompt}_{\text{extract}})$
  Store M_t in vector database
end for

A.2 Hyperparameter Settings

Table 3 lists the key hyperparameters used in our experiments.

A.3 Model Configuration

We evaluate AutoViewMem under two backbone LLMs: Qwen3-8B and GPT-4o-mini.

For memory retrieval, we encode memory items and queries using a dense embedding model and store them in a vector index supporting cosine similarity search.

Unless otherwise specified, the decoding temperature is set to 0.0 and the maximum generation length is 8192 tokens.

Algorithm 2 Recursive Graph Consolidation

Require: Memory Set $\mathcal{M}$, Initial Threshold τ_0, Max Batch Size B
Embed memories: $E \leftarrow \text{Encoder}(\mathcal{M})$
Build Graph $G = (V, E)$ where $(i, j) \in E$ if $\cos(e_i, e_j) \geq \tau_0$
Find connected components $\mathcal{C} \leftarrow \text{ConnectedComponents}(G)$
for each component $C \in \mathcal{C}$ **do**
  if $|C| > B$ **then**
    Recursive Refinement
    Increase threshold: $\tau' \leftarrow \tau_0 + \Delta\tau$ (e.g., $\Delta\tau = 0.05$)
    Apply Recursive Graph Consolidation on C with τ'
  else
    Consolidation
    Generate canonical memory $m_{cons} \leftarrow \text{LLM}(C, \text{Prompt}_{\text{dedup}})$
    Replace C with m_{cons} in $\mathcal{M}$
  end if
end for

Table 3: Hyperparameter Settings

Parameter	**Value**
Embedding Model	
Model Name	e5-base-v2
Dimension	768
View Learning	
Buffer Threshold (N)	50 turns
Candidate Views per Step	10
Target Views (K)	10
DPP Kernel	Linear ($z_i^\top z_j$)
Graph Consolidation	
Initial Similarity Threshold (τ)	0.8
Max Batch Size (B)	30

A.4 View Discovery Settings

View discovery is triggered based on a fixed-size interaction trace buffer. The system accumulates recent dialogue chunks and initiates the discovery process when the buffer reaches 50 entries.

At each discovery step, the LLM proposes K = 10 candidate views representing potential semantic dimensions of user information. These candidate views are subsequently passed to the view convergence stage for clustering, deduplication, and DPP-based selection.

B Qualitative Case Analysis of Learned Views

To better understand what the proposed view induction module learns in practice, we present two representative qualitative cases from LoCoMo. We focus on whether AutoViewMem separates mixed conversational evidence into a small set of complementary and low-overlap semantic views.

B.1 Case 1: Jon

Jon's conversations mix business development, interpersonal support, and event progression. AutoViewMem separates these signals into the following views:

- **View 1: Career Trajectory.** This view stores long-term professional direction, such as major ventures, setbacks, and growth-related decisions.

- **View 2: Relationship Dynamics.** This view captures how Jon interacts with close collaborators, including support, coordination, and shared responsibility.
- **View 3: Project Timeline.** This view isolates concrete milestones and temporally anchored events, such as launches, competitions, and other project-related progress markers.

These views are intentionally low-overlap: the first focuses on long-range professional development, the second on interpersonal structure, and the third on time-specific operational events.

B.2 Case 2: Caroline

Caroline's dialogue combines advocacy, everyday personal activity, and emotional reflection. AutoViewMem organizes them into the following views:

- **View 1: Community Engagement.** This view captures advocacy, volunteering, and participation in broader social or community-facing activities.
- **View 2: Personal Activity Profile.** This view stores recurring individual activities and stable personal attributes, such as creative work, routines, and ongoing commitments.
- **View 3: Emotional Themes.** This view summarizes higher-level affective patterns, such as resilience, self-acceptance, and other recurring emotional signals.

Compared with Jon's case, the semantic axes are different, but the same principle holds: each view occupies a distinct role, reducing overlap between public/community information, individual activity, and emotional interpretation.

B.3 Discussion

Across both cases, the learned views are more compact and more orthogonal than broad topic-based summaries. Instead of mixing facts, relationships, events, and emotions in the same memory space, AutoViewMem separates them into cleaner semantic dimensions. This supports more stable retrieval, since different query intents are more likely to align with a single focused view rather than compete within an entangled representation.

These examples illustrate that AutoViewMem successfully disentangles heterogeneous information into distinct semantic dimensions (e.g., professional milestones vs. emotional dynamics vs. advocacy themes), verifying the "orthogonality" objective.

C Prompt Templates

We list the prompt templates used for View Discovery, Extraction, Consolidation, and Evaluation in our experiments.

Table 4: Prompt used for View Discovery (Divergence Phase).

Meta-Prompt – Adaptive Layer
You are an expert in information extraction for a comprehensive long-term memory system. Your goal is to extract ALL valuable information from the conversation, covering different granularities and dimensions. {context_instruction} Read the conversation snippet below. Generate 5-10 specific extraction instructions. The instructions should cover multiple layers of granularity: 1. **Fact-level details**: Specific entities, dates, events, numbers, names (e.g., 'Extract the user's specific dietary restrictions mentioned'). 2. **Concept-level attributes**: User traits, preferences, habits, style (e.g., 'Extract the user's general communication style'). 3. **Relational information**: Connections between people, projects, or concepts.

4. **Temporal/Process information**: Current state of tasks, plans, or timelines.
5. **Implicit patterns**: Underlying intents, emotional states, or recurring themes.
Each instruction must be an imperative sentence starting with 'Extract...'. Ensure the instructions are diverse and cover both fine-grained details and high-level summaries.
Conversation Snippet:
{text_chunk}
{output_format_instruction}

Table 5: Prompts used for View Clustering and Adaptive View Extraction.

Clustering Summary Prompt – Adaptive Layer

Here is a list of semantically similar memory extraction instructions (views) from a clustering algorithm:
{prompt_text}
Please summarize these into a single, comprehensive, and representative extraction instruction (view). The summary should cover the common core intent of these views while being concise and actionable.
Return ONLY the summarized instruction text.

Adaptive View Extraction Prompt

Perspective: {prompt}
Extract stable facts and schemas from the following conversation matching the perspective.
Each schema should include:
- type: The category of information (e.g., persona, preference, fact, relationship, etc. relevant to the perspective)
- content: The extracted information
- supporting_spans: List of original message IDs or text snippets that support this schema
Output Format:
Return a JSON object with a single key "facts". The value of "facts" must be a list of STRINGS, where each string is a valid JSON representation of the schema.
Example format:
{{
"facts": [
"{{\"type\": \"preference\", \"content\": \"Alice prefers sci-fi movies\", \"supporting_spans\": [\"msg_123\"]}}"
]
}}
Conversation:
{text}
Extract the relevant information concisely. If no relevant information is found, return nothing.

Table 6: Prompt used for generating the final answer based on retrieved memories.

Answer Generation Prompt

You are an intelligent memory assistant tasked with retrieving accurate information from conversation memories.
 # CONTEXT:
You have access to memories from two speakers in a conversation. These memories contain timestamped information that may be relevant to answering the question.
 # INSTRUCTIONS:
1. Carefully analyze all provided memories from both speakers
2. Pay special attention to the timestamps to determine the answer
3. If the question asks about a specific event or fact, look for direct evidence in the memories

4. If the memories contain contradictory information, prioritize the most recent memory
5. If there is a question about time references (like "last year", "two months ago", etc.), calculate the actual date based on the memory timestamp. For example, if a memory from 4 May 2022 mentions "went to India last year," then the trip occurred in 2021.
6. Always convert relative time references to specific dates, months, or years. For example, convert "last year" to "2022" or "two months ago" to "March 2023" based on the memory timestamp. Ignore the reference while answering the question.
7. Focus only on the content of the memories from both speakers. Do not confuse character names mentioned in memories with the actual users who created those memories.
8. The answer should be less than 5-6 words.
 # APPROACH (Think step by step):
1. First, examine all memories that contain information related to the question
2. Examine the timestamps and content of these memories carefully
3. Look for explicit mentions of dates, times, locations, or events that answer the question
4. If the answer requires calculation (e.g., converting relative time references), show your work
5. Formulate a precise, concise answer based solely on the evidence in the memories
6. Double-check that your answer directly addresses the question asked
7. Ensure your final answer is specific and avoids vague time references
Memories for user {{speaker_1_user_id}}:
{{speaker_1_memories}}
Memories for user {{speaker_2_user_id}}:
{{speaker_2_memories}}
Question: {{question}}
Answer:

Table 7: Prompt used for offline memory deduplication and consolidation.

Memory Deduplication Prompt

You are a memory consolidation assistant. You are given a list of similar memories from a user's memory bank. Your task is to deduplicate them.
Rules:
1. If memories are identical or one is a subset of another, keep the most complete/detailed one.
2. If memories are complementary, merge them into a single comprehensive memory.
3. If memories are actually distinct despite being similar (e.g., different dates or specific details that shouldn't be merged), keep them separate.
4. Output a JSON list of the final memories. Do not include markdown code blocks (like "'json) in your response.
5. For each final memory, specify the 'content' and the list of 'source_ids' (the numeric IDs provided below) that contributed to it.
Memories:
{json.dumps(memory_texts, ensure_ascii=False, indent=2)}
Output format (JSON only):

```
{{
    "resolved_memories": [
        {{ "content": "Merged or selected content...", "source_ids": [0, 2] }},
        {{ "content": "Another distinct memory...", "source_ids": [1] }}
    ]
}}
```

Table 8: Prompt used for LLM Judge Evaluation (GPT-4o-mini).

LLM Judge Evaluation Prompt
You are an impartial judge evaluating the correctness of an AI assistant's answer to a user's question. You will be provided with: 1. The Question. 2. The Ground Truth Answer. 3. The AI's Predicted Answer. Task: Determine if the Predicted Answer is semantically equivalent to or correctly entails the Ground Truth Answer. - Respond with "1" if it is correct (matches the core intent/facts of the ground truth). - Respond with "0" if it is incorrect, contradicts the ground truth, or misses the core facts. Question: {question} Ground Truth: {ground_truth} Predicted Answer: {predicted_answer} Output only "1" or "0".

D Baseline Implementation Details

To ensure fair comparison and reproducibility, we describe the implementation details of the baselines evaluated in our experiments:

- **Full History**: Feeds the full conversation history into the LLM context window. Due to context length limitations (e.g., 8k tokens for Qwen3-8B), the history is truncated to retain the most recent turns that fit within the window.
- **SimpleMem**: A robust baseline that utilizes an adaptive and iterative retrieval strategy. When a query is complex, it dynamically expands the retrieval window or performs multi-hop retrieval to gather sufficient context.
- **Mem0** (Chhikara et al., 2025): An open-source long-term memory system. We deploy Mem0 using its default semantic deduplication and chunk-based vector search. It relies on standard memory extraction prompts without our multi-view orthogonal constraints.
- **MemGAS** (Xu et al., 2025a): A memory system focusing on multi-granularity association and adaptive retrieval. It attempts to balance the coverage-noise trade-off by dynamically selecting the granularity of retrieved memory units.
- **LightMem** (Fang et al., 2025): A lightweight memory management framework designed for efficiency. It uses minimal overhead during the write phase but may suffer from semantic interference under complex queries.
- **A-mem** (Xu et al., 2025b): An agentic memory control system that relies on an active LLM controller to manage memory reads and writes, focusing on inference-time routing.

For all baselines, we maintain the same underlying embedding model (`e5-base-v2`) and LLM backend (either Qwen3-8B or GPT-4o-mini) to isolate the impact of the memory organization architecture.

E Dataset Statistics

The LoCoMo (Long-Term Conversational Memory) benchmark (Maharana et al., 2024) evaluates LLMs on extended interactions. Key statistics of our evaluation setup include:

- **Conversation Length**: Each session ranges from 200 to 400 turns, introducing significant topic drift and temporal updates.
- **Evaluation Set**: The benchmark contains 1,540 questions designed to test various reasoning capabilities over long histories.
- **Question Categories**:
  - *Multi-Hop*: Requires synthesizing facts from multiple distinct memory spans.
  - *Temporal*: Requires reasoning about the order of events or tracking changing preferences over time.
  - *Open Domain*: Broad queries requiring general retrieval of user context.
  - *Single Hop*: Direct factual lookups of static preferences or explicit details.

F Retrieval Metrics Definition

Due to the multi-view nature of AutoViewMem, multiple generated memory records may originate from the same dialogue evidence (i.e., the same turn or span). Evaluating retrieval solely on memory item IDs would artificially inflate the recall score if redundant items from different views are retrieved. Therefore, we evaluate retrieval at the *evidence level*.

Let $\mathcal{E}^*(q)$ be the set of ground-truth evidence units (provenance IDs) required to answer query q. Let $\mathcal{M}_K(q)$ be the top-K memory items retrieved by the system. We map each retrieved item $m \in \mathcal{M}_K(q)$ to its underlying evidence unit $e(m)$. The deduplicated retrieved evidence set is $\mathcal{E}_K(q) = \{e(m) \mid m \in \mathcal{M}_K(q)\}$.

Evidence Recall@K is defined as:

$$\text{Evidence Recall@}K = \frac{|\mathcal{E}_K(q) \cap \mathcal{E}^*(q)|}{|\mathcal{E}^*(q)|} \tag{3}$$

Evidence NDCG@K is calculated by assigning a relevance score of 1 to the first occurrence of any evidence unit $e \in \mathcal{E}^*(q)$ in the ranked list, and 0 to duplicates or irrelevant units. Standard NDCG formulation is then applied to this deduplicated relevance vector, ensuring the system is rewarded for ranking diverse, correct evidence highly, rather than filling the top-K slots with redundant views of the same fact.