Memory Matters More: Event-Centric Memory as a Logic Map for Agent Searching and Reasoning

Yuyang Hu, Jiongnan Liu, Jiejun Tan, Yutao Zhu, Zhicheng Dou*
Gaoling School of Artificial Intelligence, Renmin University of China
yuyang.hu@ruc.edu.cn, dou@ruc.edu.cn

Abstract

Large language models (LLMs) are increasingly deployed as intelligent agents that reason, plan, and interact with their environments. To effectively scale to long-horizon scenarios, a key capability for such agents is a memory mechanism that can retain, organize, and retrieve past experiences to support downstream decision-making. However, most existing approaches organize and store memories in a flat manner and rely on simple similarity-based retrieval techniques. Even when structured memory is introduced, existing methods often struggle to explicitly capture the logical relationships among experiences or memory units. Moreover, memory access is largely detached from the constructed structure and still depends on shallow semantic retrieval, preventing agents from reasoning logically over long-horizon dependencies. In this work, we propose `CompassMem`, an event-centric memory framework inspired by Event Segmentation Theory. `CompassMem` organizes memory as an Event Graph by incrementally segmenting experiences into events and linking them through explicit logical relations. This graph serves as a logic map, enabling agents to perform structured and goal-directed navigation over memory beyond superficial retrieval, progressively gathering valuable memories to support long-horizon reasoning. Experiments on LoCoMo and NarrativeQA demonstrate that `CompassMem` consistently improves both retrieval and reasoning performance across multiple backbone models. Our code is available at https://github.com/namespace-ERI/CompassMem.

1 Introduction

With the rapid development of large language models (LLMs), agents have evolved from simple interfaces into systems capable of complex reasoning and long-term interaction with environments (Zhang et al., 2025d; Wang et al., 2024). To support such behaviors, agents require memory mechanisms that go beyond simple text generation capabilities (Ouyang et al., 2025; Zhang et al., 2025d). Ideally, similar to human memory, agent memory should serve not only as a repository of knowledge, but also as a fundamental infrastructure that supports reasoning, planning, and decision-making (Wu et al., 2025a; Zhang et al., 2025c).

Within the broader field of agent memory research, a significant amount of attention has been directed toward *factual memory* (Zhang et al., 2025b; Hu et al., 2025). Factual memory refers to an agent's capacity to manage explicit information about past events, users, and the external environment. Such memory supports context awareness, personalization, and long-horizon tasks. Despite significant progress in this area, current approaches face two primary limitations. First, regarding memory structure, most methods rely on flat representations where information is stored as independent text segments, as shown in Figure 1 (a) (Hu et al., 2025). While some recent studies have explored structured organizations (Xu et al., 2025; Rasmussen et al., 2025; Rezazadeh et al., 2025; Sun and Zeng, 2025; Li et al., 2025a), they often fail to capture essential logical relations, such as causality and temporal sequences (Figure 1 (b)) (Yang et al., 2025). Second, regarding memory utilization, prior work primarily depends on simple semantic matching (Rasmussen et al., 2025). This reliance limits memory to functioning as a static storage system rather than an active component that guides the reasoning process.

In contrast, human memory is organized hierarchically and connected through rich logical associations rather than as a collection of isolated facts. Cognitive science offers theoretical support for this organization, particularly through Event Segmentation Theory (Baldassano et al., 2017; Zacks et al., 2007). According to this theory, humans naturally perceive continuous experience as a series of discrete and meaningful events. These events form the backbone of long-term memory and are encoded with rich temporal and semantic

*Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 22389–22407

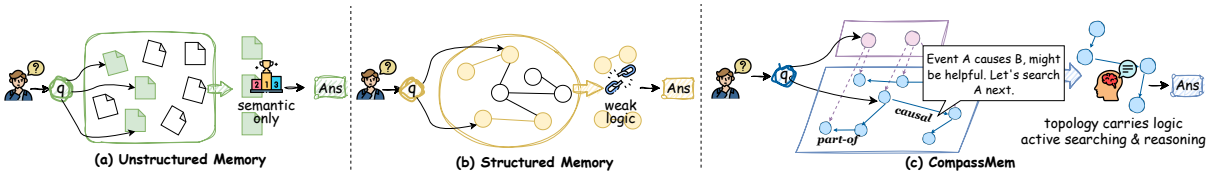

Figure 1: Comparison among CompassMem and the traditional agent memory framework.

information (Baldassano et al., 2017; Ezzyat and Davachi, 2011). This structured organization facilitates efficient retrieval. It enables the brain to selectively access relevant events by navigating a structured network, which helps guide reasoning and planning in new situations (Anderson, 1983). Unfortunately, these capabilities are largely absent in existing agent memory systems. This disparity leads to a critical research question: *Can we structure agent memory in a way that mimics human cognitive organization to support search and reasoning beyond isolated facts?*

Inspired by these cognitive principles, we propose CompassMem, an event-centric memory framework that explicitly models logical relations among memory units and leverages this structure to guide agent searching and reasoning. Unlike traditional approaches that store isolated text snippets, CompassMem incrementally constructs an *Event Graph* from experiences through **event segmentation**, **relation extraction**, and **topic evolution**. In this graph, nodes correspond to coherent event units, while edges encode logical dependencies such as causality and temporal order. During the inference phase, agents utilize the Event Graph as a **structured logic map** rather than a flat list. This structure provides directional cues that guide agent searching and reasoning. It allows agents to prioritize relevant information, follow meaningful logical connections, and avoid redundant retrieval. In this manner, memory goes beyond merely supplying content and actively guides the reasoning process to handle complex queries effectively. CompassMem achieves consistent and substantial improvements over strong baselines on LoCoMo and NarrativeQA, particularly on tasks requiring multi-hop and temporal reasoning. These results demonstrate that explicitly encoding logical structure into memory not only improves retrieval quality, but also enables memory to actively support reasoning, rather than serving as a passive knowledge store.

Our contributions are as follows:

(1) We propose CompassMem, an event-centric memory framework that organizes experiences into event units connected by explicit logical relations.

(2) In CompassMem, we design a graph-based memory retrieval mechanism, enabling agents to actively navigate the Event Graph for logic-aware evidence collection, rather than just relying on flat, similarity-based memory access.

(3) We evaluate CompassMem on dialogue and long-document benchmarks, observing consistent improvements and validating its effectiveness and generality.

2 Related Work

Memory has been widely regarded as a core capability of intelligent agents (Zhang et al., 2025d; Tan et al., 2026). Early systems such as MemGPT (Packer et al., 2023) manage long-term memory through paging and segmentation mechanisms, which inspired subsequent frameworks including MemOS (Li et al., 2025b) and MemoryOS (Kang et al., 2025). Methods such as Mem0 (Chhikara et al., 2025) and MemoryBank (Zhong et al., 2024) follow a RAG-style paradigm, placing greater emphasis on memory organization and lifecycle management.

Also, a growing line of work explores structured memory representations. Representative examples include tree-based designs such as MemTree (Rezazadeh et al., 2025), graph-based memories like A-Mem (Xu et al., 2025), and more general hierarchical or compositional memory systems (Rasmussen et al., 2025; Zhang et al., 2025a; Wu et al., 2025b; Li et al., 2025a; Fang et al., 2025). These approaches demonstrate the benefit of introducing structure into memory, particularly for improving organization. In parallel, other studies investigate automatic memory management and adaptation from different perspectives (Yan et al., 2025b; Wang et al., 2025).

While these methods enrich memory management, memory is still largely treated as a passive storage. Our work designs an event-centric memory that explicitly encodes logical structure and actively guides searching and reasoning.

3 Preliminary

In this section, we formalize the task setting and introduce the core concepts used in our approach.

We consider an agent operating over a stream of textual observations at time step t, denoted as $\mathcal{X}_t = (x_{t,1}, \ldots, x_{t,n})$, where each $x_{t,i}$ is a text unit such as a dialogue turn or a narrative sentence. Given the incoming observations and the previously stored memory $\mathcal{M}^{(t-1)}$, the agent updates its memory through a construction process Φ:

$$\mathcal{M}^{(t)} = \Phi(\mathcal{X}_t, \mathcal{M}^{(t-1)}). \tag{1}$$

Here, Φ first extracts a sub-memory $\mathcal{M}_t$ from the current input stream, and then integrates it with the existing memory $\mathcal{M}^{(t-1)}$, yielding the updated memory $\mathcal{M}^{(t)}$.

At inference time, given a query $q \in \mathcal{Q}$, the agent performs query-dependent memory search through a retrieval process Ψ:

$$\mathcal{M}^{(t)}|_q = \Psi(q, \mathcal{M}^{(t)}), \tag{2}$$

where $\mathcal{M}^{(t)}|_q \subseteq \mathcal{M}^{(t)}$ denotes the subset of memory selected for answering the query.

The final response is generated by a conditional generation function:

$$y = \mathcal{F}(q, \mathcal{M}^{(t)}|_q), \tag{3}$$

which produces an output $y \in \mathcal{Y}$ conditioned on the query and the retrieved memory. Our goal is to design more effective memory construction processes Φ and retrieval strategies Ψ to support higher-quality generation.

4 Method

4.1 Overview

As shown in Figure 2, `CompassMem` is an event-centric memory framework designed to make memory an active guide for agent searching and reasoning. The core idea is to organize memory as a structured hierarchical Event Graph, where experiences are stored as coherent event units connected by explicit logical relations.

Memory is constructed incrementally from input streams by segmenting continuous observations into events, extracting relations among them, and integrating the resulting structures into the existing memory over time. During inference, the agent performs logic-aware memory search by actively navigating the Event Graph. Rather than retrieving isolated memories by similarity, the agent follows meaningful logical paths between related events and progressively collects relevant evidence, with the memory structure guiding both where to search and how to reason for complex, long-horizon queries.

4.2 Incremental Hierarchical Memory Construction

We construct memory in an incremental manner. The system first segments the input into coherent events, then extracts explicit relations among these events, and finally integrates them into the memory through incremental graph updates.

4.2.1 Event Segmentation

Event Segmentation Theory (EST) (Baldassano et al., 2017) suggests that humans organize continuous experience into discrete and coherent events, which serve as fundamental units of long-term memory. An *event* is not an arbitrary text span, but a meaningful unit obtained by segmenting a continuous experience stream. Following this perspective, we prompt an LLM to identify events from the input stream and extract their attributes:

$$\mathcal{E}_t = \{e_{t_i}\}_{i=1}^{m} = \Phi_{\text{seg}}(\mathcal{X}_t), \tag{4}$$

where each event $e_{t_i} \in \mathcal{E}_t$ is represented as $e_{t_i} = \langle o_{t_i}, \tau_{t_i}, s_{t_i}, \pi_{t_i} \rangle$. Here, o_{t_i} denotes the span of observations belonging to the event, τ_{t_i} captures temporal information, s_{t_i} is a semantic summary, and π_{t_i} denotes the set of involved participants.

4.2.2 Relation Extraction

A memory composed of isolated events provides limited support for reasoning (Hu et al., 2025). In contrast, humans reason and form associations by following logical connections (Anderson, 1983). To enable structured retrieval and multi-step reasoning, we explicitly extract logical relations among event nodes using an LLM-based process:

$$\mathcal{R}_t = \{(e_{t_i}, e_{t_j}, \rho_{t_{ij}})\} = \Phi_{\text{rel}}(\mathcal{X}_t, \mathcal{E}_t), \tag{5}$$

where each relation $r_{ij} = (e_i, e_j, \rho_{ij})$ represents a logical dependency between two events. The relation label ρ_{ij} is drawn from an open-ended predicate set $\mathcal{P}$, covering relations such as *causal*, *temporal*, *motivation*, and *part-of*, and allowing new relation types to be introduced as needed. Together, the extracted events $\mathcal{E}_t$ and relations $\mathcal{R}_t$ form the current sub-memory $\mathcal{M}_t = (\mathcal{E}_t, \mathcal{R}_t)$.

4.2.3 Incremental Graph Update

As memory grows over time, new events must be integrated while preserving coherence, so that newly acquired information can be connected to existing

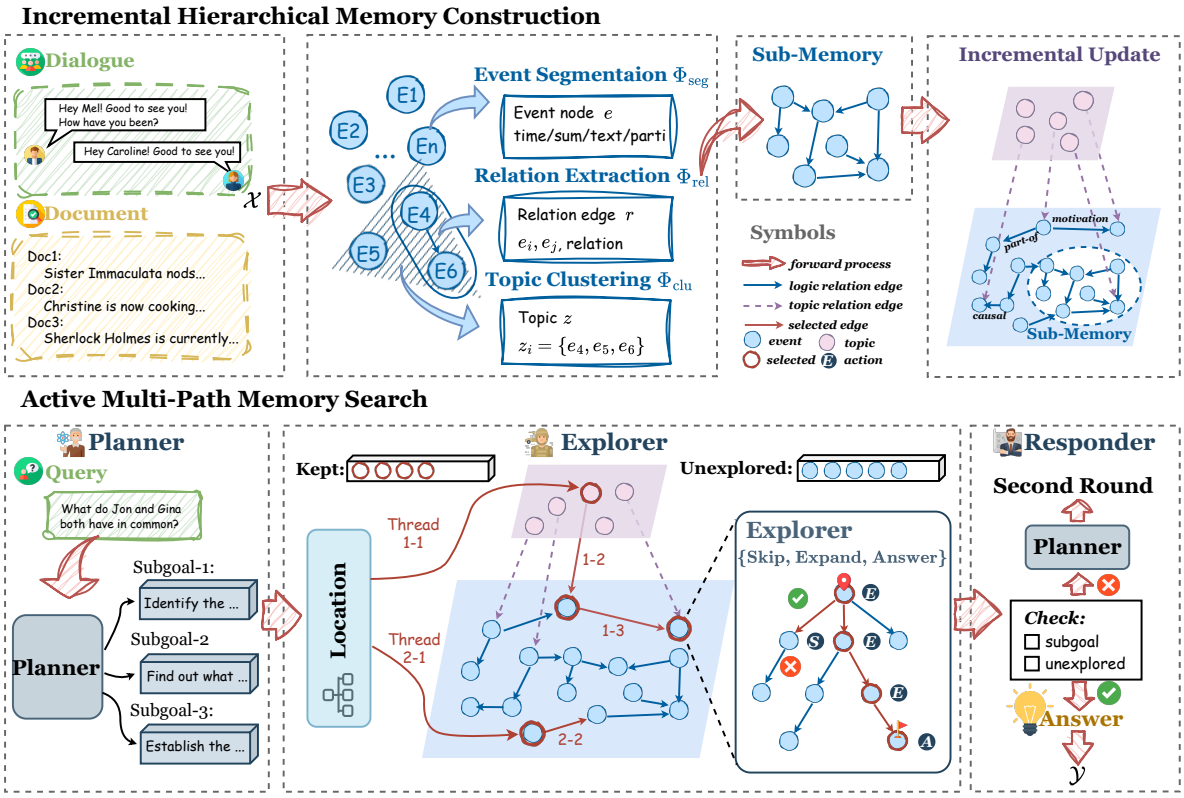

Figure 2: Overview of the proposed CompassMem framework, which contains mainly two part: Incremental Hierarchical Memory Construction and Active Multi-Path Memory Search

knowledge without introducing redundancy or semantic drift. We incrementally update memory $\mathcal{M}^{(t-1)}$ by incorporating new sub-memory $\mathcal{M}_t$ through the following two parts.

Node Fusion & Expansion Each new event $e_{\text{new}} \in \mathcal{E}_{t+1}$ is compared against existing events, where e^* denotes the most similar existing event. The integration follows three cases. If e_{new} is equivalent to e^*, the two events are merged. If a logical relation between e_{new} and e^* is identified, an edge is added to link them. Otherwise, e_{new} is inserted as a new node. This process integrates new information while avoiding redundancy.

Topic Evolution During memory search, exploration driven purely by local similarity may focus on a single semantic aspect of a query, which can be insufficient for complex questions involving multiple facets. To address this issue, we introduce a topic layer over the accumulated event set $\mathcal{E}^{(t)} = \bigcup_{i=1}^{t} \mathcal{E}_i$. Each topic $z_k \in \mathcal{Z}^{(t)}$ represents a semantic cluster of related events, and the topic–event associations are maintained in $\mathcal{A}^{(t)}$, indicating which events belong to each topic. This topic layer provides a coarse-grained semantic organization of events, which complements the fine-grained logical structure defined by event relations and facilitates efficient multi-path exploration during memory search.

At the initial stage (e.g., $t = 1$), when no topic structure exists, we use K-means to perform topic clustering over the extracted events to initialize the topic set:

$$\mathcal{Z}^{(1)} = \{z_1, z_2, \ldots, z_k\} = \Phi_{\text{clu}}(\mathcal{E}^{(1)}; k). \quad (6)$$

As memory grows over time, we update topic–event associations in an online manner. For each newly integrated event $e_{\text{new}} \in \mathcal{E}_{t+1}$, we identify its most similar topic from the existing topic set $\mathcal{Z}^{(t)}$ based on semantic similarity. If the similarity exceeds a threshold δ, the event is assigned to that topic; otherwise, a new topic node is created to capture a previously unseen semantic direction. This process incrementally updates both the topic set $\mathcal{Z}^{(t)}$ and the topic–event associations $\mathcal{A}^{(t)}$.

To prevent semantic drift introduced by incremental updates, we periodically re-cluster all accumulated events:

$$\mathcal{Z}^{(t+1)} \leftarrow \Phi_{\text{clu}}(\mathcal{E}^{(t+1)}; k) \quad \text{when } t \bmod T = 0.$$

This strategy balances stability during online updates with global coherence over memory growth.

By treating temporally situated events as primary memory units, the resulting event graph preserves narrative structure and event-level semantics, which are often lost in triple-based representations (Yang et al., 2025). From this perspective, memory itself serves as an explicit *logic map* that guides subsequent search and reasoning. Prompts for all memory construction processes are provided in the Appendix F.1.

4.3 Active Multi-Path Memory Search

With the event graph constructed as a structured logic map, memory search proceeds through active navigation and reasoning. Given a query q, the goal is to retrieve a small set of event nodes that provide sufficient evidence.

`CompassMem` adopts a principle of guided active evidence construction. Reasoning is performed through traversal of the event graph, while only distilled evidence is passed to the final answer model. To support this process, we implement memory search Ψ using three LLM-based agents: a **Planner**, multiple **Explorers**, and an **Responder**. Prompts for all agents are provided in the Appendix F.2.

4.3.1 Planner

Given a query q, the **Planner** decomposes it into a small set of subgoals,

$$\mathcal{H}_q = \Psi_{\text{plan}}(q), \qquad |\mathcal{H}_q| \in [2, 5], \tag{7}$$

where each subgoal captures a distinct aspect that the search should cover. The Planner maintains a binary satisfaction vector $\mathbf{s} \in \{0, 1\}^K$ to indicate which subgoals have been supported by the currently collected evidence. This explicit progress signal provides a notion of the search stage and guides exploration toward unsatisfied subgoals.

If the search fails to terminate with sufficient evidence in the current round, the Planner performs gap-aware refinement. It generates a refined query by conditioning on the current query, the collected evidence, and the remaining unsatisfied subgoals,

$$q^{(r+1)} = \Psi_{\text{ref}}(q^{(r)}, \mathcal{H}_q, \mathbf{s}), \tag{8}$$

where refinement focuses on unfinished subgoals. This design yields a closed-loop process that alternates between exploration and query refinement.

4.3.2 Explorer

Active searching and reasoning over the memory is carried out by a set of **Explorer** agents. Guided by the memory topology structure, each Explorer operates directly on the event graph and decides which nodes to retain as evidence and how exploration should proceed.

Localization Before graph traversal, exploration is first localized to determine where to begin. Candidate event nodes are retrieved by ranking their embedding similarity to the query, and the top-k results are selected. Since these events are often highly similar and may focus on a single aspect, the Explorer further selects candidates from the first p distinct topic clusters appearing in the ranked list. From the resulting candidate set $\mathcal{C}_q$, the starting nodes are selected as:

$$\mathcal{S}_q = \Psi_{\text{start}}(q, \mathcal{C}_q),$$

where Ψ_{start} denotes an LLM-based selection operator. The selected starting nodes are then inserted into a globally maintained queue to initialize subsequent exploration.

Navigation Guided by the event-graph topology, exploration proceeds step by step. At each visited event node e, an Explorer conditions on the query, the current subgoal status, the retained evidence, and the local graph context, including neighboring nodes. Based on this information, the Explorer chooses an action from the action space $\{\textsc{Skip}, \textsc{Expand}, \textsc{Answer}\}$:

$$a = \Psi_{\text{cho}}(q, e, \hat{\mathcal{E}}, \mathcal{N}(e), \mathbf{s}), \tag{9}$$

where $\hat{\mathcal{E}}$ denotes the current evidence set and $\mathcal{N}(e)$ denotes neighboring events with typed relations. SKIP discards the current node, EXPAND retains it as evidence and continues exploration, and ANSWER terminates the current path when sufficient evidence has been collected. When EXPAND is selected, the evidence set is updated as:

$$\hat{\mathcal{E}}^{(t+1)} = \begin{cases} \hat{\mathcal{E}}^{(t)}, & \text{if } a = \text{SKIP}, \\ \hat{\mathcal{E}}^{(t)} \cup \{e\}, & \text{otherwise.} \end{cases} \tag{10}$$

Each retained node is annotated with the subgoals it supports, enabling explicit progress tracking.

This decision process operationalizes our key insight that *topology carries logic*: relations constrain exploration paths and guide reasoning over structured dependencies, rather than flat and isolated text.

Coordination Multiple Explorers run in parallel, each initialized from a different starting node. They share a global state that records visited nodes, retained evidence, and subgoal progress. All candidate nodes encountered during traversal are scheduled through a single global priority queue. The

Model	Method	Single-hop		Multi-hop		Open-domain		Temporal		Average	
		F1	BLEU	F1	BLEU	F1	BLEU	F1	BLEU	F1	BLEU
GPT-4o-mini	*Non-Graph-based*										
	RAG	52.19	46.80	32.17	23.59	23.21	18.88	30.77	25.99	42.25	36.47
	Mem0	47.65	38.72	38.72	27.13	**28.64**	21.58	48.93	40.51	45.10	35.90
	MemoryOS	48.62	42.99	35.27	25.22	20.02	15.52	41.15	30.76	42.84	35.47
	Graph-based										
	HippoRAG	54.84	48.84	33.59	25.46	28.59	**23.89**	48.17	39.32	47.92	41.02
	A-Mem	44.65	37.06	27.02	20.09	12.14	12.00	45.85	36.67	39.65	32.31
	CAM	50.58	44.36	33.55	24.18	18.23	12.77	44.14	38.28	44.10	37.43
	CompassMem	**57.36**	**49.79**	**38.84**	**27.98**	26.61	20.01	**57.96**	**50.51**	**52.18**	**44.09**
Qwen2.5-14B	*Non-Graph-based*										
	RAG	49.79	43.95	28.11	21.43	20.42	17.40	24.73	20.02	38.77	33.18
	Mem0	42.58	35.15	31.73	24.82	15.03	11.28	28.96	26.24	36.04	29.91
	MemoryOS	46.33	41.62	38.19	29.26	20.27	15.94	32.24	27.86	40.28	34.89
	Graph-based										
	HippoRAG	42.45	37.14	27.57	20.62	19.74	15.81	30.66	26.33	35.85	30.53
	A-Mem	33.75	30.04	22.09	15.28	13.49	10.74	27.19	22.05	28.98	24.47
	CAM	50.39	45.59	34.50	24.62	23.86	20.84	44.70	36.30	44.64	38.27
	CompassMem	**61.02**	**55.93**	**42.32**	**32.66**	**25.88**	**22.01**	**47.18**	**39.69**	**52.52**	**46.17**

Table 1: Performance comparison on the LoCoMo benchmark, covering single-hop, multi-hop, open-domain, and temporal settings. We report F1 and BLEU-1 scores (%). Best results are highlighted in **bold**, and second-best results are underlined.

Model	Method	F1	BLEU
GPT-4o-mini	RAG	28.99	25.68
	Mem0	29.98	23.34
	MemoryOS	25.58	21.74
	HippoRAG	28.77	23.04
	A-Mem	27.01	23.17
	CAM	33.55	29.74
	CompassMem	**39.04**	**35.23**
Qwen2.5-14B	RAG	25.82	20.65
	Mem0	26.94	22.01
	MemoryOS	22.17	19.32
	HippoRAG	22.10	17.77
	A-Mem	25.37	20.94
	CAM	27.87	23.47
	CompassMem	**35.90**	**28.66**

Table 2: Results on 298 questions belonging to 10 documents randomly sampled from NarrativeQA. We do the sample since the full test set contains over 10,000 questions and is too large for long-context evaluation.

priority of a candidate node u is defined by its embedding similarity to unsatisfied subgoals:

$$p(u) = \max_{j:s_j=0} \text{sim}\big(v(s_u), v(h_j)\big), \qquad (11)$$

where s_u denotes the summary of u and h_j denotes a subgoal. This subgoal-driven scheduling reduces redundant exploration and promotes complementary coverage across paths, enabling efficient multi-path reasoning over the event graph.

4.3.3 Responder

The **Responder** is invoked when the global candidate queue becomes empty, and all subgoals are satisfied. If the queue becomes empty while some subgoals remain unsatisfied, the system returns to the Planner to start the second round search.

Upon termination, the search returns a concise evidence set $\mathcal{M}^{(t)}|_q = \hat{\mathcal{E}}$. If no evidence is retained, we fall back to the initial top-k retrieved candidates. The Responder then generates the final output, ensuring that generation conditions only on distilled evidence while reasoning is carried out through structured navigation on the logic map.

5 Experiment

5.1 Experimental Settings

Benchmarks We evaluate CompassMem on two long-context reasoning benchmarks, **LoCoMo** and **NarrativeQA**. LoCoMo focuses on conversational question answering, while NarrativeQA targets narrative understanding. Detailed dataset descriptions are provided in the Appendix B.1.

Backbone Models We use **GPT-4o-mini** as a closed-source model, and **Qwen2.5-14B-Instruct** as an open-source model. Qwen is deployed with vLLM, while GPT is accessed via API. All methods use **BGE-M3** for all mentioned embeddings.

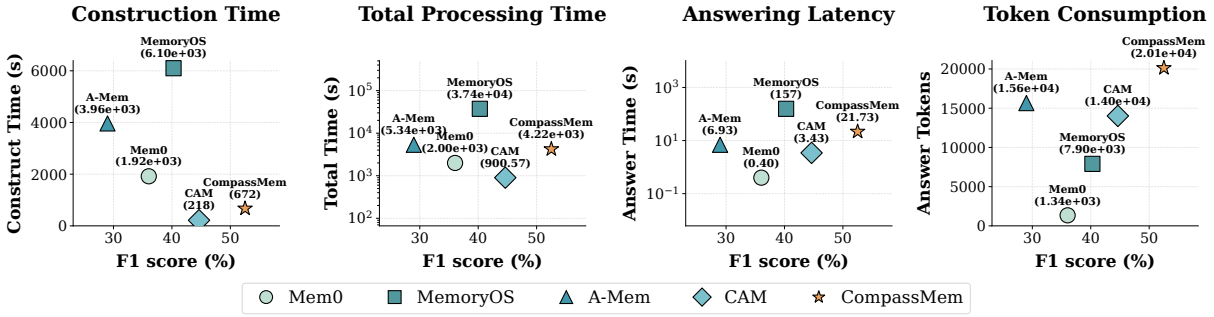

Figure 3: Efficiency–performance trade-off across memory frameworks. Scatter plots compare F1 with construction time, total processing time, per-question latency, and token consumption.

Baselines We compare CompassMem with non-graph baselines, including RAG, Mem0 (Chhikara et al., 2025), and MemoryOS (Kang et al., 2025), as well as graph-based baselines such as HippoRAG (Gutiérrez et al., 2025), A-Mem (Xu et al., 2025), and CAM (Li et al., 2025a). Official implementations or reported settings are used when available, with full implementation details provided in the Appendix B.3.

5.2 Main Results

We now present the main experimental results and several key observations. Addational results are provided in the Appendix C.

(1) Table 1 reports results on LoCoMo across question types. While most methods handle single-hop questions reasonably well, performance drops sharply on multi-hop and temporal QA. In contrast, CompassMem consistently achieves the strongest results. On GPT-4o-mini, it improves average F1 from 47.92% (HippoRAG) to 52.18%, with a large gain on temporal questions (57.96% vs. 48.93%). On Qwen2.5-14B, CompassMem further reaches 52.52% F1 and achieves the best performance on all subsets. These results demonstrate the benefit of event-graph memory with logic-aware retrieval for reasoning-intensive QA.

(2) Table 2 presents results on NarrativeQA, which requires long-range narrative understanding and evidence aggregation. CompassMem consistently outperforms all baselines, surpassing the strongest competitor CAM by over 5% F1 on GPT-4o-mini and more than 8% F1 on Qwen2.5-14B. This demonstrates the effectiveness of event-centric memory with explicit relations for retrieving globally relevant evidence in long narratives.

(3) Across both benchmarks, CompassMem shows consistent and robust improvements. Notably, the strongest baselines are generally graph-based, supporting the importance of structured memory.

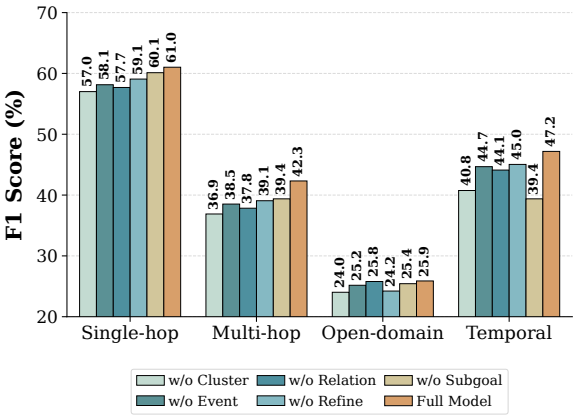

Figure 4: Ablation results on LoCoMo.

CompassMem further advances these methods by modeling memory at the event level with logic-aware relations, yielding the largest gains on tasks that require complex retrieval and reasoning.

(4) We further analyze efficiency on a representative LoCoMo conversation set, ramdomly selected from ten sessions with identical settings, as shown in Figure 3. CompassMem achieves low memory construction time cost, substantially lower than Mem0, A-Mem, and MemoryOS. Our total processing time and per-question latency are comparable to Mem0 and A-Mem, and markedly lower than MemoryOS. Although CompassMem uses more tokens, this cost is accompanied by substantial performance gains. Overall, CompassMem delivers strong reasoning improvements while maintaining practical computational efficiency.

5.3 Further Analysis

We further conduct in-depth analyses to better understand the behavior of CompassMem.

Ablation Study To examine the effectiveness of individual components in CompassMem, we conduct an ablation study by systematically removing key modules. Specifically, we evaluate variants that (i) remove topic clustering, (ii) replace event units

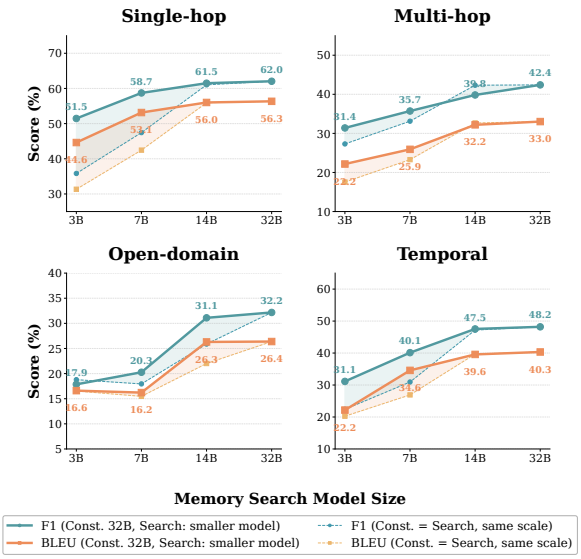

Figure 5: Scaling results comparing fixed high-capacity and scale-consistent memory construction. Shaded areas show gains from stronger construction models.

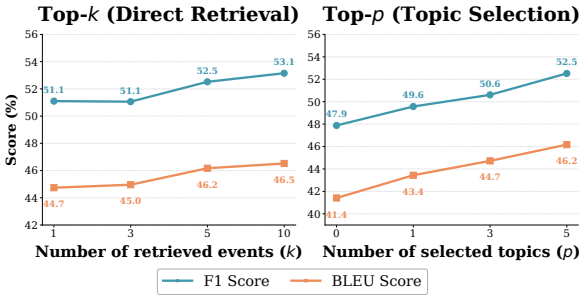

Figure 6: Sensitivity analysis of CompassMem with respect to localization hyperparameters

with fixed-length chunks to eliminate event modeling, where the chunk length is set to the average size of extracted events (approximately 100 tokens), (iii) remove edges to discard explicit relations, (iv) disable query refinement to prevent second-round exploration, and (v) remove subgoal generation. Figure 4 reports the ablation results across question categories. Removing any component leads to consistent performance drops, confirming the contribution of each module. In particular, multi-hop and temporal questions are most affected, while single-hop and open-domain questions show smaller degradation due to lower reasoning complexity.

Impact of Model Size We examine the scalability of CompassMem. Figure 5 shows that CompassMem continues to improve as model scale increases when the same backbone is used for both memory construction and search. We further evaluate a decoupled setting where memory is constructed with a high-capacity model (Qwen2.5-32B) while search and response generation use

Method	Single-hop		Multi-hop		Open-domain		Temporal	
	F1	BLEU	F1	BLEU	F1	BLEU	F1	BLEU
RAG	44.26	37.98	23.84	15.46	11.39	8.33	17.88	13.09
MemO	37.50	31.76	23.58	15.17	14.37	11.48	41.29	30.37
MemoryOS	40.87	35.84	24.71	19.28	16.09	14.50	39.41	28.71
HippoRAG	46.12	40.58	31.62	24.52	22.04	17.47	40.39	32.94
A-Mem	44.57	39.37	28.53	20.16	18.35	15.23	31.60	23.49
CAM	45.79	38.48	34.07	26.01	19.96	16.36	43.82	36.11
Ours	**50.04**	**43.40**	**35.33**	**27.86**	**28.02**	**23.25**	**49.35**	**38.91**

Table 3: Results of LoCoMo on Qwen3-8B.

smaller models. This configuration yields clear improvements. These results suggest that high-quality memory structures built offline can effectively support downstream reasoning, even when paired with lightweight search models.

Impact of Location Hyperparameters Figure 6 analyzes the sensitivity of CompassMem to two localization hyperparameters: the direct retrieval size k, and the topic selection size p. Overall, introducing topic-based selection ($p > 0$) consistently improves performance compared to the no-clustering setting, and larger values of p lead to steadily better results. This suggests that selecting starting nodes from multiple semantic topics helps diversify exploration and reduces bias toward a single semantic view. Similarly, increasing retrieval size k provides a broader pool of candidate events and yields monotonic performance gains, indicating that richer initial retrieval better supports downstream search.

Impact of Model Thinking Ability Table 3 reports LoCoMo results on Qwen3-8B, a backbone equipped with explicit thinking capability. All methods benefit from the stronger reasoning capacity, with noticeable improvements on multi-hop and temporal questions compared to non-thinking models. Nevertheless, CompassMem consistently achieves the best performance across all task categories. The gains indicate that explicit reasoning alone is insufficient. Effective memory organization and logic-aware retrieval remain critical for fully exploiting the backbone's thinking ability.

6 Conclusion

We presented CompassMem, an event-centric memory framework that rethinks agent memory as a structured logic map rather than a flat storage. By organizing experiences into coherent events and explicitly modeling their logical relations, CompassMem enables memory to actively guide searching and reasoning. Experiments on dialogue and long-document demonstrate that this design provides strong and consistent benefits, particularly for reasoning-intensive tasks. We hope this work

encourages future research on memory structures that more directly support long-horizon reasoning and decision-making in intelligent agents.

Limitations

While CompassMem shows consistent gains, it has several limitations. First, the quality of the Event Graph depends on event segmentation and relation extraction. We adopt a naive LLM-based pipeline and more fine-grained and robust segmentation may further improve memory quality. We leave this direction for future work. Second, our evaluation focuses on a set of representative benchmarks. Demonstrating the effectiveness of CompassMem across a broader range of tasks and agent settings would further strengthen its applicability.

Ethical considerations

This work studies agent memory architectures for long-context reasoning and does not introduce new datasets. All experiments are conducted on publicly available benchmarks, LoCoMo and NarrativeQA, which do not contain sensitive personal information. We do not intentionally collect, or generate content that identifies specific individuals.

Acknowledgement

This work was supported by National Natural Science Foundation of China No. 62272467. The work was partially done at the Engineering Research Center of Next-Generation Intelligent Search and Recommendation, MOE.

References

John R Anderson. 1983. A spreading activation theory of memory. *Journal of verbal learning and verbal behavior*, 22(3):261–295.

Christopher Baldassano, Janice Chen, Asieh Zadbood, Jonathan W Pillow, Uri Hasson, and Kenneth A Norman. 2017. Discovering event structure in continuous narrative perception and memory. *Neuron*, 95(3):709–721.

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

Sarah DuBrow, MJ Kahana, and AD Wagner. 2024. Event and boundaries. *Oxford handbook of human memory*, 1.

Youssef Ezzyat and Lila Davachi. 2011. What constitutes an episode in episodic memory? *Psychological science*, 22(2):243–252.

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. 2025. Lightmem: Lightweight and efficient memory-augmented generation. *CoRR*, abs/2510.18866.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From RAG to memory: Non-parametric continual learning for large language models. In *ICML*. OpenReview.net.

Yuyang Hu, Shichun Liu, Yanwei Yue, Guibin Zhang, Boyang Liu, Fangyi Zhu, Jiahang Lin, Honglin Guo, Shihan Dou, Zhiheng Xi, Senjie Jin, Jiejun Tan, Yanbin Yin, Jiongnan Liu, Zeyu Zhang, Zhongxiang Sun, Yutao Zhu, Hao Sun, Boci Peng, and 28 others. 2025. Memory in the age of ai agents. *Preprint*, arXiv:2512.13564.

Jiazheng Kang, Mingming Ji, Zhe Zhao, and Ting Bai. 2025. Memory OS of AI agent. *CoRR*, abs/2506.06326.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2017. The narrativeqa reading comprehension challenge. *Preprint*, arXiv:1712.07040.

Rui Li, Zeyu Zhang, Xiaohe Bo, Zihang Tian, Xu Chen, Quanyu Dai, Zhenhua Dong, and Ruiming Tang. 2025a. CAM: A constructivist view of agentic memory for llm-based reading comprehension. *CoRR*, abs/2510.05520.

Zhiyu Li, Shichao Song, Chenyang Xi, Hanyu Wang, Chen Tang, Simin Niu, Ding Chen, Jiawei Yang, Chunyu Li, Qingchen Yu, Jihao Zhao, Yezhaohui Wang, Peng Liu, Zehao Lin, Pengyuan Wang, Jiahao Huo, Tianyi Chen, Kai Chen, Kehang Li, and 20 others. 2025b. Memos: A memory OS for AI system. *CoRR*, abs/2507.03724.

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

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

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

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

Alireza Rezazadeh, Zichao Li, Wei Wei, and Yujia Bao. 2025. From isolated conversations to hierarchical schemas: Dynamic tree memory representation for llms. In *ICLR*. OpenReview.net.

Haoran Sun and Shaoning Zeng. 2025. Hierarchical memory for high-efficiency long-term reasoning in llm agents. *Preprint*, arXiv:2507.22925.

Jiejun Tan, Zhicheng Dou, Liancheng Zhang, Yuyang Hu, Yiruo Cheng, and Ji-Rong Wen. 2026. Memsifter: Offloading llm memory retrieval via outcome-driven proxy reasoning. *Preprint*, arXiv:2603.03379.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Jirong Wen. 2024. A survey on large language model based autonomous agents. *Frontiers Comput. Sci.*, 18(6):186345.

Yu Wang, Ryuichi Takanobu, Zhiqi Liang, Yuzhen Mao, Yuanzhe Hu, Julian J. McAuley, and Xiaojian Wu. 2025. Mem-α: Learning memory construction via reinforcement learning. *CoRR*, abs/2509.25911.

Yaxiong Wu, Sheng Liang, Chen Zhang, Yichao Wang, Yongyue Zhang, Huifeng Guo, Ruiming Tang, and Yong Liu. 2025a. From human memory to AI memory: A survey on memory mechanisms in the era of llms. *CoRR*, abs/2504.15965.

Yaxiong Wu, Yongyue Zhang, Sheng Liang, and Yong Liu. 2025b. Sgmem: Sentence graph memory for long-term conversational agents. *CoRR*, abs/2509.21212.

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

B. Y. Yan, Chaofan Li, Hongjin Qian, Shuqi Lu, and Zheng Liu. 2025a. General agentic memory via deep research. *Preprint*, arXiv:2511.18423.

Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Hinrich Schütze, Volker Tresp, and Yunpu Ma. 2025b. Memory-r1: Enhancing large language model agents to manage and utilize memories via reinforcement learning. *CoRR*, abs/2508.19828.

Zairun Yang, Yilin Wang, Zhengyan Shi, Yuan Yao, Lei Liang, Keyan Ding, Emine Yilmaz, Huajun Chen, and Qiang Zhang. 2025. EventRAG: Enhancing LLM generation with event knowledge graphs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16967–16979, Vienna, Austria. Association for Computational Linguistics.

Jeffrey M Zacks, Nicole K Speer, Khena M Swallow, Todd S Braver, and Jeremy R Reynolds. 2007. Event perception: a mind-brain perspective. *Psychological bulletin*, 133(2):273.

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

Guibin Zhang, Muxin Fu, and Shuicheng Yan. 2025b. Memgen: Weaving generative latent memory for self-evolving agents. *CoRR*, abs/2509.24704.

Guibin Zhang, Haotian Ren, Chong Zhan, Zhenhong Zhou, Junhao Wang, He Zhu, Wangchunshu Zhou, and Shuicheng Yan. 2025c. Memevolve: Meta-evolution of agent memory systems. *Preprint*, arXiv:2512.18746.

Zeyu Zhang, Quanyu Dai, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. 2025d. A survey on the memory mechanism of large language model-based agents. *ACM Trans. Inf. Syst.*, 43(6):155:1–155:47.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2024. Memorybank: Enhancing large language models with long-term memory. In *AAAI*, pages 19724–19731. AAAI Press.

Appendix

A Event Segmentation Theory

Event Segmentation Theory (EST) (Baldassano et al., 2017; Zacks et al., 2007; Ezzyat and Davachi, 2011) is a framework in cognitive science and neuroscience that explains how humans parse continuous streams of perceptual experience into meaningful units, or events. According to this theory, when perceiving a dynamic environment, humans do not process information as an undifferentiated continuous flow. Instead, experience is automatically segmented into a sequence of relatively stable event episodes. Within each event, representations remain coherent and stable; when a salient change occurs, such as a shift in scene, action goals, or environmental state, an event boundary is triggered, prompting the construction of a new event representation model.

This segmentation process operates not only at the perceptual level but also plays a critical role in the encoding of event memories and their subsequent retrieval. Event Segmentation Theory emphasizes that human experience is not a continuous whole, but rather is composed of a series of identifiable event units. Such segmentation enhances perceptual efficiency and provides a fundamental basis for memory structuring and information retrieval (DuBrow et al., 2024).

B Experiment Details

B.1 Dataset Descriptions

LoCoMo LoCoMo (Maharana et al., 2024) is a benchmark of very long-term conversational dialogues designed to evaluate long-range memory and reasoning capabilities in agent systems. The dataset consists of 10 extended conversations, each spanning dozens of sessions and hundreds of dialogue turns, with an average of around 600 turns and roughly 16K tokens per conversation. Questions in the LoCoMo QA evaluation are annotated with answer locations and categorized into types such as single-hop, multi-hop, open-domain, temporal reasoning, and adversarial, targeting different memory and inference challenges. In our experiments on LoCoMo QA, we follow standard practice in related work and do not use adversarial question data, which aligns with previous evaluations (Chhikara et al., 2025; Yan et al., 2025a; Kang et al., 2025).

NarrativeQA NarrativeQA (Kočiský et al., 2017) is a large-scale reading comprehension benchmark that assesses models' ability to understand and reason over long narrative text such as books and movie scripts. The full NarrativeQA dataset contains on the order of tens of thousands of human-written question–answer pairs associated with over a thousand story documents, where questions require synthesis across global document structure rather than shallow pattern matching. Questions are constructed based on human-generated abstractive summaries, encouraging deep narrative understanding and integrative reasoning beyond local context overlaps. In our evaluation, we randomly sampled 10 long documents from the NarrativeQA corpus and used their associated 298 QA pairs to measure performance on long-range narrative question answering. This sampling strategy is adopted because the full NarrativeQA test set contains 10,557 questions, making exhaustive evaluation computationally prohibitive. The selected documents have an average length of around 60,000 tokens, which still poses a substantial challenge for long-context understanding and coherent evidence aggregation.

B.2 CompassMem

In `CompassMem`, we adopt a fixed set of hyperparameters across all main experiments. During memory construction, newly extracted events are merged with existing ones when their semantic similarity exceeds a threshold of 0.9, which helps reduce redundancy while preserving coherent event structure. In the topic evolution stage, we apply the same similarity threshold (0.9) when merging events into existing topics, and perform periodic re-clustering every 4 construction steps to maintain semantic coherence over time. For the LoCoMo benchmark, memory localization retrieves the top-k=5 candidate events based on embedding similarity. To encourage multi-perspective exploration, candidates are selected from the top-p=5 distinct topic clusters. Topic clustering is performed using k-means, where the number of clusters is automatically determined by the current memory size as $n_{\text{clusters}} = \max(2, \min(\lfloor n_{\text{samples}}/5 \rfloor, 50))$ During memory search, we employ three parallel Explorer agents to conduct multi-path traversal over the Event Graph. Query refinement is limited to a single additional round to control search complexity. Our choice of hyperparameters is motivated by the analysis in Section 5.3.

For NarrativeQA, where documents are substantially longer, we increase the retrieval scope to top-k=10 while keeping all other settings unchanged. This adjustment allows broader initial coverage without altering the overall search strategy.

B.3 Baseline

Description

- Mem0 (Chhikara et al., 2025): A scalable long-term memory system that dynamically extracts, consolidates, and retrieves salient facts from ongoing dialogues or streams. It maintains a compact set of memory entries by continually updating and merging similar facts, avoiding redundancy, and retrieves only the most relevant facts rather than reprocessing the full context.

- MemoryOS (Kang et al., 2025): A hierarchical memory architecture designed specifically for AI agents in long conversational interactions. It organizes memory into multiple tiers (short-, mid-, and long-term stores) and coordinates four core modules—memory storage, dynamic update, adaptive retrieval, and response generation—to maintain continuity, context coherence, and personalization over long dialogues.

- HippoRAG (Gutiérrez et al., 2025): A graph-based retrieval-augmented generation framework inspired by the hippocampal indexing theory of human long-term memory. It transforms documents into a knowledge graph and uses Personalized PageRank over concept seeds to integrate information across disparate contexts, enabling efficient single-step multi-hop retrieval. This structure allows deeper integration of new experiences and improved retrieval for reasoning-intensive tasks compared to standard RAG.

- A-Mem (Xu et al., 2025): An agentic memory system for LLM agents that dynamically organizes memory entries into an interconnected network using principles from human note-taking methods. When new memories are added, it generates structured notes with multiple attributes and connects them to related historical memories, enabling continuous memory evolution and contextual organization beyond fixed schemas.

- CAM (Li et al., 2025a): A structured memory framework grounded in constructivist theory, which organizes memory hierarchically and supports flexible integration and dynamic adaptation. It maintains overlapping clusters and hierarchical summaries and explores memory structure during retrieval in a way reminiscent of human associative processes, improving both performance and efficiency on long-text reading tasks.

Implementation For baselines that rely on chunk-based retrieval, we apply a unified preprocessing strategy by segmenting documents into fixed-length chunks of 512 tokens. For all such methods, we retrieve the top-5 most relevant chunks based on embedding similarity and use them as context for downstream reasoning or answer generation. This ensures a consistent retrieval budget across chunk-based baselines.

For memory-based baselines, we follow their original experimental settings and implementations as described in the corresponding papers or official codebases, without additional modification. This design ensures a fair comparison while preserving the intended behavior of each baseline.

C Detailed Search and Reasoning Statistics

This section provides a detailed analysis of the search and reasoning behavior of `CompassMem` on the LoCoMo benchmark. We report aggregated statistics to characterize efficiency, exploration dynamics, and the role of planning and refinement during memory search.

C.1 Overall Statistics

Table 4 summarizes the overall runtime, retrieval, and reasoning statistics across all 1,540 questions. On average, each query is processed within a moderate and stable time budget, indicating that active navigation over the Event Graph does not lead to excessive overhead. The median runtime is close to the mean, suggesting consistent behavior across different queries.

The Planner generates approximately three subgoals per question, providing structured guidance for exploration. While not all subgoals are fully satisfied, partial satisfaction is common, reflecting the varying availability of supporting evidence in memory. The high refinement rate indicates that

iterative query adjustment plays an important role in addressing uncovered aspects during search.

Total Questions	1540
Time Metrics	
Total Time	32136.5 s
Avg. Time per Question	20.87 s
Median Time	19.32 s
Max Time	65.38 s
Min Time	4.84 s
Subgoal Metrics	
Avg. Subgoals	3.04
Avg. Subgoal Satisfaction	68.3%
Fully Satisfied	594 (38.6%)
Retrieval Metrics	
Avg. Retrieved Nodes	50.0
Avg. Initial Nodes	3.7
Avg. Similarity	0.7374
Traversal Metrics	
Avg. Paths	2.5
Avg. Total Steps	7.5
Avg. Path Length	2.84
Max Path Length	11
Avg. Max Rounds	2.4
Action Distribution	
Total Actions	11595
EXPAND	7348 (63.4%)
SKIP	4230 (36.5%)
ANSWER	17 (0.1%)
Queue Metrics	
Avg. Initial Queue Size	3.7
Avg. Max Queue Size	3.7
Refinement Metrics	
Refinement Count	1176
Refinement Rate	76.4%
Kept Nodes	
Avg. Kept Nodes	3.15
Max Kept Nodes	14
No Kept Nodes	102

Table 4: Overall search and reasoning statistics on LoCoMo.

Path Length	**Count**	**Percentage**
1	172	11.2%
2	410	26.6%
3	336	21.8%
4	217	14.1%
5	155	10.1%
6	169	11.0%
7	21	1.4%
8	32	2.1%
9	18	1.2%
10	6	0.4%
11	4	0.3%

Table 5: Distribution of exploration path lengths in memory search.

C.2 Per-Item Aggregated Statistics

Table 6 reports statistics aggregated by item groups in LoCoMo. Across different items, the average runtime and exploration depth remain relatively stable, suggesting that the proposed search mechanism adapts robustly to different dialogue structures and content distributions. Variations in refinement rate and retained evidence reflect differences in reasoning complexity across items, rather than instability in the search process.

C.3 Statistics by Question Category

Table 7 summarizes the search and reasoning behavior of `CompassMem` across different question categories. Reasoning-intensive questions, particularly multi-hop, require longer search trajectories, as reflected by higher average steps and longer processing time. Temporal questions, while involving fewer steps on average, exhibit the highest refinement rate, indicating frequent use of query refinement to resolve temporal dependencies. Single-hop questions are generally easier, requiring fewer steps and refinements while maintaining a high subgoal satisfaction rate. Overall, these patterns align well with the inherent complexity of each category and suggest that `CompassMem` adapts its search behavior according to task demands.

C.4 Path Length Distribution

Table 5 shows the distribution of exploration path lengths. Most paths are short, with the majority falling between two and four steps, indicating that useful evidence is typically reached through localized reasoning over event relations. Longer paths are rare and correspond to more complex queries requiring extended exploration, demonstrating that deep traversal is selectively invoked rather than pervasive.

D Case Study: Multi-hop Reasoning over the Event Graph

We present a representative multi-hop question from the LoCoMo benchmark to qualitatively illustrate how `CompassMem` performs logic-aware memory search and reasoning over the Event Graph. This example highlights how evidence is incrementally constructed through structured traversal rather than flat retrieval. The case query is:

> *"What kinds of artworks did the speaker mention creating after moving to the new city?"*

Answering this question requires linking events about relocation with later creative activities that are mentioned in separate dialogue segments.

Item	#Q	Avg. Time (s)	Avg. Steps	Refine %	Avg. Kept
locomo_item1	152	21.73	7.9	82.9	3.3
locomo_item2	81	21.21	7.6	80.2	3.1
locomo_item3	152	20.02	6.9	79.6	2.8
locomo_item4	199	22.28	8.5	71.9	3.7
locomo_item5	178	19.57	6.9	77.5	2.3
locomo_item6	123	21.25	7.8	77.2	3.4
locomo_item7	150	17.89	5.7	80.7	2.0
locomo_item8	191	20.05	7.1	70.7	3.1
locomo_item9	156	21.96	8.0	76.9	3.6
locomo_item10	158	22.80	8.8	70.9	4.0

Table 6: Aggregated search statistics per item group.

Category	#Q	Avg. Time (s)	Avg. Steps	Subgoal Sat. %	Refine %
Multi-hop	282	24.61	10.1	71.5	78.7
Temporal	321	18.64	5.9	61.3	83.8
Open-domain	96	24.49	9.2	57.9	85.4
Single-hop	841	20.05	7.1	71.0	71.7

Table 7: Search and reasoning statistics by question category.

Planner: Subgoal Decomposition Given the query, the Planner decomposes it into three subgoals:

- h_1: Identify the event describing the speaker's move to a new city.
- h_2: Find events mentioning artistic or creative activities after the move.
- h_3: Extract the specific types of artworks mentioned.

The Planner initializes the subgoal satisfaction vector as $\mathbf{s} = [0, 0, 0]$, which is updated as evidence is collected.

Localization: Selecting Starting Events Using embedding similarity, the system retrieves the top-5 candidate events and selects starting nodes from 5 distinct topic clusters. Example starting events include:

> *"Moved to Chicago last summer for a new job."*
> *"I have been spending weekends exploring art museums."*

A total of 3 starting nodes are inserted into the global exploration queue.

Explorer: Multi-path Navigation and Evidence Collection Three Explorer agents traverse the Event Graph in parallel. At each visited event, the Explorer conditions on the query, current subgoals, retained evidence, and local graph relations. The retained evidence set is updated accordingly. Across all paths, the agent explores 10 candidate nodes, retains 7 as evidence, and reaches an average path length of 2.84 steps.

Query Refinement After the first exploration round, the Planner observes that h_3 is only partially supported. It triggers a single refinement step to focus on missing details:

> *"What specific forms of art did the speaker create after moving to the new city?"*

This refined query guides a second round of targeted exploration, a mechanism triggered in 76.4% of LoCoMo questions overall.

Evidence Aggregation After refinement, the retained evidence set consists of the following key events:

> *"Moved to Chicago last summer for a new job."*
> *"I started painting landscapes in my apartment."*
> *"I also experimented with stained glass designs."*

These events jointly satisfy all subgoals, yielding $\mathbf{s} = [1, 1, 1]$.

Answer Generation The Answerer generates the final response conditioned only on the distilled evidence:

> *"The speaker created paintings and stained glass artworks after moving."*

Discussion. This case illustrates how CompassMem constructs answers through guided traversal over logically connected events. Rather than retrieving a single text chunk, the agent incrementally accumulates evidence across multiple paths, refines its search when gaps are detected, and reasons over event dependencies. This process mirrors human multi-step recall and demonstrates the advantage of event-centric memory for complex multi-hop reasoning.

E Use of AI Assistants

We use ChatGPT to improve the presentations of this paper.[1]

[1] https://chatgpt.com/

F Prompt Templates

F.1 Memory Construction

Event Extraction Prompt

You are an expert information extraction system. Given a multi-turn dialog, extract meaningful events and output ONE strict JSON object.

Goals:
• Extract logically coherent events (E1, E2, ...) in chronological order. Each event represents a complete logical unit.
• **AGGRESSIVELY COMBINE** related micro-events into comprehensive summaries to avoid fragmentation. Merge events that:
  – Involve same participants discussing the same topic.
  – Form a logical sequence (decision + action + completion).
  – Are temporally close and thematically related (within 3-5 utterances).
  – Represent different aspects of the same situation/problem.
  – Include follow-up questions, clarifications, or elaborations.
• **PRESERVE ALL** important details within each merged event summary. Include:
  – Complete context and all key outcomes, results, and conclusions.
  – Specific facts, numbers, dates, locations, and concrete details.
  – Emotional states, reactions, and interpersonal dynamics.
  – Technical details, requirements, and specifications.
  – Any conditions, constraints, or limitations discussed.
  – **IMPORTANT**: Include visual content descriptions for shared images.
• **Constraints**: List people involved as an array `people` (max 3). Do not output other entity types or attributes.
• **Event Count**: Extract 6-10 comprehensive events. Prioritize fewer, more detailed events over many fragmented ones.

RESPONSE FORMAT:
Output JSON only, no additional commentary.

Event Relation Extraction Prompt

You are an expert information extraction system. Given a list of extracted events from a dialog, identify meaningful pairwise relations between them and output ONE strict JSON object.

Goals:
• Consider **ALL** unordered pairs of events within the same session (not only adjacent events).
• Extract pairwise event relations with a SHORT, free-form label in `type` that best characterizes the link.
• Relation types can include: *causal, motivation, enablement, follow_up, temporal_before, temporal_after, contrast, part_of, parallel, elaboration*. These are examples, not a closed set.
• Add relations only when meaningful. Prefer specific semantic links over trivial temporal ordering.
• It is acceptable to have no temporal edges if they add no insight.

CRITICAL GUIDELINES:
• **IMPORTANT**: For temporal relations (*follow_up, temporal_before, temporal_after*), base them on the **ACTUAL TIME** when events occurred in the real world, NOT on when they are described in the dialog. Focus on the chronological sequence of reality.
• For each relation, cite minimal `evidence` utterance ids that support the linkage between the two events.

RESPONSE FORMAT:
Output JSON only, no additional commentary.

Event Coreference & Overlap Prompt

You are an expert at analyzing events and determining if they refer to the same real-world occurrence or have significant overlap.

Given two event descriptions extracted from different dialog sessions, determine:
• 1. Whether they describe the **SAME** event (same occurrence at the same time).
• 2. Whether they have **SIGNIFICANT OVERLAP** (mention or relate to the same real-world situation/topic).

Consider these factors:
• Do they involve the same people/participants?
• Do they describe the same actions, situations, or topics?
• Do they have compatible time references?
• Would merging their information create a more complete picture of ONE event?

Output a JSON object with these exact keys:

```
{
```

```
"same_event": boolean,                                  // true if they are the same event
"has_overlap": boolean,                                 // true if they refer to the same situation
"relation_type": string | null,                         // suggest relation type if overlap
"reasoning": string                                     // brief explanation
}
```

F.2 Memory Search

Action Decision Prompt

You are an expert information evaluator. Your task is to decide which action to take for the current node based on how relevant and sufficient it is for answering the given question. You have **THREE** possible actions:

1. SKIP: The current node is NOT helpful for answering the question or satisfying any sub-goals.
• Use SKIP when the current node contains completely irrelevant information.
• The current node will be DISCARDED, not used in final answer.
• You should specify which neighbor node(s) to explore next, OR specify NONE if ALL neighbors are irrelevant.
• **Multi-node selection rules**: Maximum 3 nodes, only select HIGHLY relevant ones.

2. EXPAND: The current node IS helpful and helps satisfy some sub-goals, but NOT all sub-goals are satisfied yet.
• Use EXPAND when the current node contains useful information for one or more sub-goals.
• The current node will be KEPT and used in the final answer.
• Specify neighbor node(s) to explore next to satisfy remaining sub-goals, OR specify NONE if no neighbors are relevant.
• **CRITICAL**: You MUST indicate which sub-goals are now satisfied by this node + previously kept information. Only mark a sub-goal as satisfied if you have DIRECT evidence.

3. ANSWER: Use ONLY when ALL sub-goals are SATISFIED (or nearly all).
• Use ANSWER when the previously kept information + current node together satisfy ALL sub-goals.
• The current node will be KEPT and exploration will STOP.
• **CRITICAL**: You MUST list ALL satisfied sub-goals to confirm completeness.
• Be conservative: If ANY sub-goal remains unsatisfied, use EXPAND instead.

CRITICAL GUIDELINES:
– **Check sub-goals systematically**: For each action, explicitly evaluate which sub-goals are satisfied.
– **ANSWER only when complete**: Use ANSWER only when ALL (or all critical) sub-goals are satisfied.
– **Navigate strategically**: Choose next nodes that are likely to help satisfy remaining unsatisfied sub-goals.
– **Be explicit about progress**: Always indicate which sub-goals your current decision addresses.

RESPONSE FORMAT (follow strictly):
ACTION: [SKIP/EXPAND/ANSWER]
NEXT_NODES: [NODE_ID1, NODE_ID2, ...] (or NONE)
SATISFIED_SUBGOALS: [1, 3, 4] (REQUIRED for EXPAND/ANSWER; [] for SKIP)
REASONING: [Brief explanation: (1) info provided, (2) sub-goals satisfied, (3) sub-goals remaining, (4) why chosen next nodes target remaining sub-goals]

IMPORTANT: (1) For SKIP, SATISFIED_SUBGOALS must be []; (2) For EXPAND/ANSWER: provide list even if empty; (3) Only include sub-goals with DIRECT evidence; (4) Do NOT speculate.

QUESTION: {question}
{subgoals_text}

PREVIOUSLY KEPT INFORMATION:
{kept_nodes_info if kept_nodes_info else "(No information kept yet)"}

CURRENT NODE INFORMATION:
{current_info}

NEIGHBOR NODES (available for exploration):
{neighbor_info}

Now, make your decision:

Response Generation Prompt

Your task is to answer the QUESTION based on the provided CONTEXT.

Requirements:
• **Be concise and direct**: Provide ONLY the answer in the form of **a short phrase**, not a sentence. No explanations or additional commentary.
• **Original wording**: If the context contains direct statements that answer the question, use the original wording from the context.
• **Inference**: If the context doesn't have direct statements, you may summarize and infer the answer from the relevant information.

• **Time Reference Calculation**: If there is a question about time references (like "last year", "two months ago", etc.), calculate the actual date based on the memory timestamp.
Example: If a memory from 4 May 2022 mentions "went to India last year," then the trip occurred in 2021.
• **Specific Dates**: 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.
• **Reasonable Justification**: If you are uncertain or lack sufficient information, do not state that the information is insufficient. Instead, provide a reasonable and well-justified answer based on general knowledge.
• **Keep it brief**: Keep your answer brief and to the point.

CONTEXT:
{context}

QUESTION:
{question}

ANSWER:

Refinement Query Prompt

You are an assistant whose role is to generate a refined search query to find missing information.

ORIGINAL QUESTION:
{original_question}

SUB-GOALS STATUS:
Satisfied sub-goals:
{satisfied_text}
Unsatisfied sub-goals:
{unsatisfied_text}

INFORMATION COLLECTED SO FAR:
{context_so_far}

TASK:
Generate a NEW search query that specifically targets the UNSATISFIED sub-goals.

Your new query should:
• 1. Focus on the specific unsatisfied sub-goals.
• 2. Be clear and specific.
• 3. Use different keywords or phrases than the original question.
• 4. Target information that would help satisfy the remaining sub-goals.
• 5. NOT repeat the original question.

RESPONSE FORMAT:
New Query: [Your refined search query - single clear question or search phrase targeting unsatisfied sub-goals]
Target Sub-goals: [List which sub-goal numbers this query aims to satisfy]

Generate your response:

Memory Node Selection Prompt

You are selecting the most promising memory nodes to explore for answering a question.

QUESTION: {question}
{subgoals_text}

CANDIDATE NODES (retrieved by semantic similarity):
{nodes_text}

INSTRUCTIONS:
Select the nodes that are HIGHLY LIKELY to contain information relevant to one or more sub-goals.
• **Be selective**: Only choose nodes whose summaries clearly indicate relevance to specific sub-goals.
• **Maximum 5 nodes**: Select at most 5 nodes to explore.
• **Diversity**: Try to select nodes that address different sub-goals if possible.
• **Quality over quantity**: It's better to select 2 highly relevant nodes than 5 marginally relevant ones.
• If a node's summary is vague or doesn't clearly relate to any sub-goal, DON'T select it.
• Consider both the summary content and the similarity score.

RESPONSE FORMAT:
Selected Nodes: [NODE_ID1, NODE_ID2, ...]
Reasoning: [Brief explanation of why each selected node is likely relevant to specific sub-goals]

Now make your selection:

Cluster-based Node Selection Prompt

You are selecting the most relevant memory node(s) to answer a question.

QUESTION: {question}

AVAILABLE NODES:
{nodes_text}

INSTRUCTIONS:
Select the node(s) that are HIGHLY relevant to answering the question.
• **Be selective**: Only choose nodes that are HIGHLY relevant to the question.
• **Maximum 3 nodes**: Select at most 3 nodes per cluster.
• If ONLY ONE node is clearly the most relevant, select just that one.
• Select multiple nodes (2-3) ONLY when they are ALL highly relevant AND provide complementary information:
  * Information is distributed across multiple memories about the SAME topic.
  * The question has multiple specific aspects that DIFFERENT nodes address.
  * Multiple nodes provide different pieces of the SAME answer.
• Consider the summary content, people involved, and time information.
• **Do NOT select nodes that are only tangentially related or vaguely relevant.**

RESPONSE FORMAT:
Selected Nodes: [NODE_ID1, NODE_ID2, ...]
Reason: [Brief explanation of why these specific nodes are HIGHLY relevant]

Strategic Planning Prompt

You are a strategic planning assistant. Your task is to analyze a question and break it down into 2-5 specific sub-goals that need to be satisfied to fully answer the question.

QUESTION: {question}

INSTRUCTIONS:
• 1. Analyze what information components are needed to fully answer this question.
• 2. Break down the question into 2-5 specific, concrete sub-goals.
• 3. Each sub-goal should represent a distinct piece of information needed.
• 4. Sub-goals should be:
  – Specific and clear (not vague)
  – Independently verifiable (can determine if it's satisfied)
  – Collectively sufficient (together they fully answer the question)
  – Atomic (each sub-goal addresses ONE aspect)

RESPONSE FORMAT (follow strictly):
Sub-goal 1: [First specific information need]
Sub-goal 2: [Second specific information need]
Sub-goal 3: [Third specific information need]
...

Now analyze the question and generate sub-goals: