MAGMA: A Multi-Graph based Agentic Memory Architecture for AI Agents

Dongming Jiang[α], **Yi Li**[α], **Guanpeng Li**[β], **Bingzhe Li**[α*]
[α]Department of Computer Science, The University of Texas at Dallas
[β]Department of Electrical and Computer Engineering, University of Florida
{dongming.jiang, yi.li3, bingzhe.li}@utdallas.edu
liguanpeng@ufl.edu

Abstract

Memory-Augmented Generation (MAG) extends Large Language Models with external memory to support long-context reasoning, but existing approaches largely rely on semantic similarity over monolithic memory stores, entangling temporal, causal, and entity information. This design limits interpretability and alignment between query intent and retrieved evidence, leading to suboptimal reasoning accuracy. In this paper, we propose MAGMA, a multi-graph agentic memory architecture that represents each memory item across orthogonal semantic, temporal, causal, and entity graphs. MAGMA formulates retrieval as policy-guided traversal over these relational views, enabling query-adaptive selection and structured context construction. By decoupling memory representation from retrieval logic, MAGMA provides transparent reasoning paths and fine-grained control over retrieval. Experiments on LoCoMo and LongMemEval demonstrate that MAGMA consistently outperforms state-of-the-art agentic memory systems in long-horizon reasoning tasks. The open-source code is available at: https://github.com/FredJiang0324/MAGMA.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities across a wide range of tasks (Brown et al., 2020; Achiam et al., 2023; Wei et al., 2022), yet they remain limited in their ability to maintain and reason over long-term context. These models process information within a finite attention window, and their internal representations do not persist across interactions, causing earlier details to be forgotten once they fall outside the active context (Brown et al., 2020; Beltagy et al., 2020a). Even within a single long sequence, attention effectiveness degrades with distance due to *attention dilution*, positional encoding limitations, and token interference, leading to the well-known "lost-in-the-middle" and context-decay phenomena (Liu et al., 2024; Press et al., 2021a). Moreover, LLMs lack native mechanisms for stable and structured memory, resulting in inconsistent recall, degraded long-horizon reasoning, and limited support for tasks requiring persistent and organized memory (Khandelwal et al., 2018; Maharana et al., 2024).

To address these inherent limitations, Memory-Augmented Generation (MAG) systems have emerged as a promising direction for enabling LLMs to operate beyond the boundaries of their fixed context windows. MAG equips an agent with an external memory continuously recording interaction histories and allowing the agents to retrieve and reintegrate past experiences when generating new responses. By offloading long-term context to an explicit memory module, MAG systems provide a means for agents to accumulate knowledge over time, support multi-session coherence, and adapt to evolving conversational or task contexts. In this paradigm, memory is no longer implicit in internal activations but becomes a persistent, queryable resource that substantially enhances long-horizon reasoning, personalized behavior, and stable agent identity.

Despite their promise, current MAG systems exhibit structural and operational limitations that constrain their effectiveness in long-term reasoning (Li et al., 2025; Chhikara et al., 2025; Xu et al., 2025; Packer et al., 2023; Rasmussen et al., 2025; Wang and Chen, 2025; Kang et al., 2025a). Most existing approaches store past interactions in monolithic repositories or minimally structured memory buffers, relying primarily on semantic similarity, recency, or heuristic scoring to retrieve relevant content. For example, A-Mem (Xu et al., 2025) organizes past interactions into Zettelkasten-like memory units that are incrementally linked and refined, yet their retrieval pipelines rely primarily on se-

*Corresponding author

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 36848–36865

mantic embedding similarity, missing the relations such as temporal or causal relationships. Cognitive-inspired frameworks like Nemori (Nan et al., 2025) introduce principled episodic segmentation and representation alignment, enabling agents to detect event boundaries and construct higher-level semantic summaries. However, their memory structures are still narrative and undifferentiated, with no explicit modeling of distinct relational dimensions.

To address the structural limitations of existing MAG systems, we propose MAGMA, a multi-graph agentic memory architecture that explicitly models heterogeneous relational structure in an agent's experience. MAGMA represents each memory item across four orthogonal relational graphs (i.e., semantic, temporal, causal, and entity), yielding a disentangled representation of how events, concepts, and participants are related.

Built on this unified multi-graph substrate, MAGMA introduces a hierarchical, intent-aware query mechanism that selects relevant relational views, traverses them independently, and fuses the resulting subgraphs into a compact, type-aligned context for generation. By decoupling memory representation from retrieval logic, MAGMA enables transparent reasoning paths, fine-grained control over memory selection, and improved alignment between query intent and retrieved evidence. This relational formulation provides a principled and extensible foundation for agentic memory, improving both long-term coherence and interpretability.

Our contributions are summarized as follows:

1. We propose **MAGMA**, a multi-graph agentic memory architecture that explicitly models semantic, temporal, causal, and entity relations essential for long-horizon reasoning.
2. We introduce an Adaptive Traversal Policy that routes retrieval based on query intent, enabling efficient pruning of irrelevant graph regions and achieving lower latency and reduced token usage.
3. We design a dual-stream memory evolution mechanism that decouples latency-sensitive event ingestion from asynchronous structural consolidation, preserving responsiveness while refining relational structure.
4. We demonstrate that MAGMA consistently outperforms state-of-the-art agentic memory systems on long-context benchmarks including LoCoMo and LongMemEval, while reducing retrieval latency and token consumption relative to prior systems. The code is open-sourced[1].

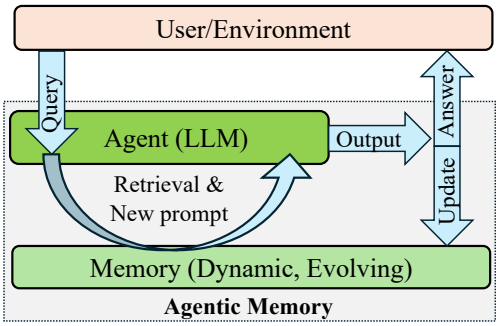

Figure 1: High-Level Architecture of Memory-Augmented Generation (MAG).

2 Background

Existing Large Language Models (LLMs) face fundamental challenges in handling long-term agentic interactions. These challenges stem from the inherent limitations of fixed-length contexts, which result in fragmented memory and an inability to maintain narrative coherence over time. The evolution of long-term consistency in LLMs is shifted from *Context-Window Extension* (Beltagy et al., 2020a; Press et al., 2021a; Kang et al., 2025c; Qian et al., 2025), *Retrieval-Augmented Generation (RAG)* (Lewis et al., 2020; Jiang et al., 2025; Wang et al., 2024; Jiang et al., 2024; Gutiérrez et al., 2025; Lin et al., 2025) to *Memory-Augmented Generation (MAG)*.

Retrieval-oriented approaches enrich the model with an external, dynamic memory library, giving rise to the paradigm of Memory-Augmented Generation (MAG) (Zhong et al., 2024; Park et al., 2023; Huang et al., 2024). Formally, unlike static RAG, MAG maintains a time-variant memory $\mathcal{M}_t$ that evolves via a feedback loop:

$$o_t = \text{LLM}(q_t, \text{Retrieve}(q_t, \mathcal{M}_t)) \quad (1)$$

$$\mathcal{M}_{t+1} = \text{Update}(\mathcal{M}_t, q_t, o_t) \quad (2)$$

As shown in Figure 1, this feedback loop enables the memory module to evolve over time: the user query is combined with retrieved information to form an augmented prompt, and the model's output is subsequently written back to refine $\mathcal{M}_t$.

Some prior schemes focused on structuring the intermediate states or relationships of memory to enable better reasoning. Think-in-Memory (TiM) (Liu et al., 2023) stores evolving chains-of-thought

[1]https://github.com/FredJiang0324/MAGMA

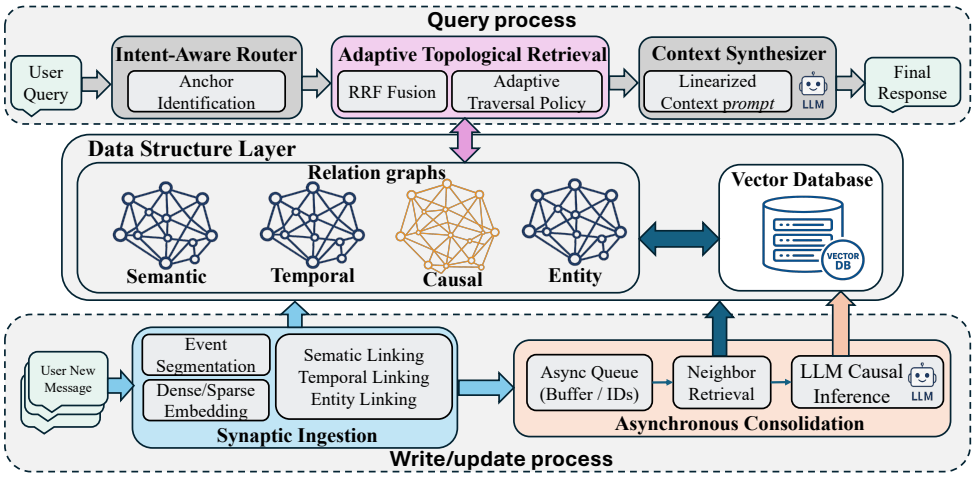

Figure 2: Architectural Overview of MAGMA. The system is composed of three layers: (1) A Query Process that routes and synthesizes context; (2) A Data Structure Layer organizing memory into Relation Graphs and a Vector Database; and (3) A Write/Update Process utilizing a dual-stream mechanism for fast ingestion and asynchronous consolidation.

to maintain consistency. A-MEM (Xu et al., 2025) draws inspiration from the Zettelkasten method, organizing knowledge into an interconnected note network. More recently, graph-based approaches like GraphRAG (Edge et al., 2024a) and Zep (Rasmussen et al., 2025) structure memory into knowledge graphs to capture cross-document dependencies. We provide a detailed discussion of related work in Appendix A.

However, prior work typically organizes memory around associative proximity (e.g., semantic similarity) rather than mechanistic dependency (Kiciman et al., 2023). As a result, such methods can retrieve *what* occurred but struggle to reason about *why*, since they lack explicit representations of causal structure, leading to reduced accuracy in complex reasoning tasks (Jin et al., 2023; Zhang et al., 2025).

3 MAGMA Design

In this section, we introduce the proposed Multi-Graph based Agentic Memory (MAGMA) design and its components in detail.

3.1 Architectural Overview

MAGMA architecture is organized into the following three logical layers, orchestrating the interaction between control logic and the memory substrate as illustrated in Figure 2.

Query Process: The inference engine responsible for retrieving and synthesizing information. It comprises the *Intent-Aware Router* for dispatching tasks, the *Adaptive Topological Retrieval* module for executing graph traversals, and the *Context Synthesizer* for generating the final narrative response.

Data Structure ($\mathcal{G}$): The unified storage substrate that fuses disparate modalities. As shown in the center of Figure 2, it maintains a *Vector Database* for semantic search alongside four distinct *Relation Graphs* (i.e., Semantic, Temporal, Causal and Entity). This layer provides the topological foundation for cross-view reasoning.

Write/Update Process: A dual-stream pipeline manages memory evolution. It decouples latency-sensitive operations via *Synaptic Ingestion* (Fast Path) from compute-intensive reasoning via *Asynchronous Consolidation* (Slow Path), ensuring the system remains responsive while continuously deepening its memory structure.

Functionally, the Query Layer interacts with the Data Structure Layer to execute the synchronous Query Process (Section 3.3), while the Write/Update Layer manages the continuous Memory Evolution (Section 3.4).

3.2 Data Structure Layer

As the core component of Memory-Augmented Generation (MAG), the data structure layer is responsible for storing, organizing, and evolving past information to support future retrieval and updates. In MAGMA, we formalize this layer as a time-variant directed multigraph $\mathcal{G}_t = (\mathcal{N}_t, \mathcal{E}_t)$, where nodes represent events and edges encode heterogeneous relational structures. This unified manifold

enables structured reasoning across multiple logical dimensions(i.e., semantic, temporal, causal, and entity) while preserving their orthogonality.

Unified node representation: The node set $\mathcal{N}$ is hierarchically organized to represent experience at multiple granularities, ranging from fine-grained atomic events to higher-level episodic groupings. Each Event-Node $n_i \in \mathcal{N}_{\text{event}}$ is defined as:

$$n_i = \langle c_i, \tau_i, \mathbf{v}_i, \mathcal{A}_i \rangle \tag{3}$$

where c_i denotes the event content (e.g., observations, actions, or state changes), τ_i is a discrete timestamp anchoring the event in time, and $\mathbf{v}_i \in \mathbb{R}^d$ is a dense representation indexed in the vector database (Johnson et al., 2019). The attribute set $\mathcal{A}_i$ captures structured metadata such as entity references, temporal cues, or contextual descriptors, enabling hybrid retrieval that integrates semantic similarity with symbolic and structural constraints.

Relation graphs (edge space): The edge set $\mathcal{E}$ is partitioned into four semantic subspaces, corresponding to the relation graphs:

- **Temporal Graph ($\mathcal{E}_{temp}$):** Defined as strictly ordered pairs (n_i, n_j) where $\tau_i < \tau_j$. This immutable chain provides the ground truth for chronological reasoning.
- **Causal Graph ($\mathcal{E}_{causal}$):** Directed edges representing logical entailment. An edge $e_{ij} \in \mathcal{E}_{causal}$ exists if $S(n_j|n_i, q) > \delta$, explicitly inferred by the consolidation module to support "Why" queries.
- **Semantic Graph ($\mathcal{E}_{sem}$):** Undirected edges connecting conceptually similar events, formally defined by $\cos(\mathbf{v}_i, \mathbf{v}_j) > \theta_{sim}$.
- **Entity Graph ($\mathcal{E}_{ent}$):** Edges connecting events to abstract entity nodes, solving the object permanence problem across disjoint timeline segments.

3.3 Query Process: Adaptive Hierarchical Retrieval

As illustrated in Figure 3, retrieval in MAGMA is formulated as a policy-guided graph traversal rather than a static lookup operation. The query process is orchestrated by a Router $\mathcal{R}$, which decomposes the user query into structured control signals and executes a multi-stage retrieval pipeline (Algorithm 1) that dynamically selects, traverses, and fuses relevant relational views. Four main stages in the query process is introduced below:

Stage 1 - Query Analysis & Decomposition: The process begins by decomposing the raw user query q into structured control signals, including semantic, lexical, and temporal cues. MAGMA then extracts three complementary representations to guide the retrieval process:

- **Intent Classification (T_q):** A lightweight classifier maps q to a specific intent type $T_q \in \{\text{WHY}, \text{WHEN}, \text{ENTITY}\}$. This acts as the "steering wheel," determining which graph edges will later be prioritized (e.g., "Why" queries trigger a bias for Causal edges).
- **Temporal Parsing ($[\tau_s, \tau_e]$):** A temporal tagger resolves relative expressions (e.g., "last Friday") into absolute timestamps, defining a hard time window for filtering.
- **Representation Extraction:** The system simultaneously generates a dense embedding $\vec{q}$ for semantic search and extracts sparse keywords q_{key} for exact lexical matching.

Stage 2 - Multi-Signal Anchor Identification: Before initiating graph traversal, the system first identifies a set of anchor nodes that serve as entry points into the memory graph. To ensure robustness across query modalities, we fuse signals from dense semantic retrieval, lexical keyword matching, and temporal filtering using Reciprocal Rank Fusion (RRF) (Cormack et al., 2009):

$$S_{anchor} = \text{Top}_K \left(\sum_{m \in \{vec, key, time\}} \frac{1}{k + r_m(n)} \right) \tag{4}$$

This ensures robust starting points regardless of query modality.

Stage 3 - Adaptive Traversal Policy: Starting from the anchor set $\mathcal{S}_{anchor}$, the system expands the context using a Heuristic Beam Search. Unlike rigid rule-based traversals, MAGMA calculates a dynamic transition score $S(n_j|n_i, q)$ for moving from node n_i to neighbor n_j via edge e_{ij}. This score fuses structural alignment with semantic relevance:

$$S(n_j|n_i, q) = \exp\Bigg(\lambda_1 \cdot \underbrace{\phi(type(e_{ij}), T_q)}_{\substack{\text{Structural} \\ \text{Alignment}}} + \lambda_2 \cdot \underbrace{\text{sim}(\vec{n}_j, \vec{q})}_{\substack{\text{Semantic} \\ \text{Affinity}}} \Bigg) \tag{5}$$

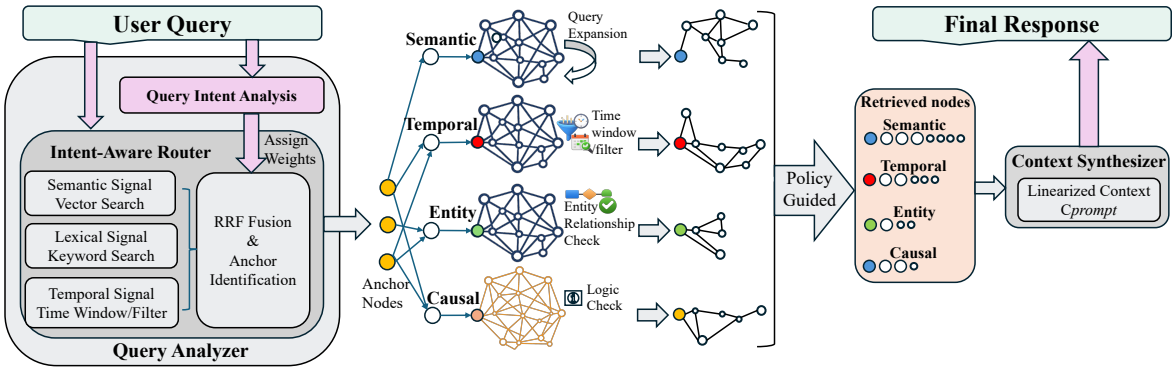

Figure 3: Query process with adaptive hybrid retrieval pipeline. (1) Query Analysis detects intent and fuses signals to find Anchors. (2) Adaptive Traversal navigates specific graph views (Causal, Temporal) based on the policy weights.

Here, $\text{sim}(\cdot)$ denotes the cosine similarity between the neighbor's embedding and the query embedding. The structural alignment function ϕ dynamically rewards edge types based on the detected query intent T_q:

$$\phi(r, T_q) = \mathbf{w}_{T_q}^{\top} \cdot \mathbf{1}_r \tag{6}$$

where $\mathbf{w}_{T_q}$ is an adaptive weight vector specific to intent T_q (e.g., assigning high weights to CAUSAL edges for "Why" queries), and $\mathbf{1}_r$ is the one-hot encoding of the edge relation.

At each step, the algorithm retains the top-k nodes with the highest cumulative scores. This ensures the traversal is guided by a dual signal: strictly following the logical structure (via ϕ) while maintaining contextual focus (via sim).

Stage 4: Narrative Synthesis via Graph Linearization: The final phase transforms the retrieved subgraph $\mathcal{G}_{sub}$ into a coherent narrative context. MAGMA employs a structure-aware linearization protocol that preserves the relational dependencies encoded in the graph with the following three phases.

1. Topological Ordering: Raw nodes are reorganized to reflect the logic of the query. For temporal queries ($T_q = \text{WHEN}$), nodes are sorted by timestamp τ_i. For causal queries ($T_q = \text{WHY}$), we apply a topological sort on the causal edges $\mathcal{E}_{causal}$ to ensure causes precede effects in the prompt context.

2. Context Scaffolding with Provenance: To mitigate hallucination, each node is serialized into a structured block containing its timestamp, content, and explicit reference ID. We define the linearized context C_{prompt} as:

Algorithm 1 Adaptive Hybrid Retrieval (Heuristic Beam Search)

Input: Query q, Graph G, VectorDB V, Intent T_q
Output: Narrative Context C_{out}

// *Phase 1: Initialization*
$S_{anchor} \leftarrow \text{RRF}(V.\text{SEARCH}(\vec{q}), K.\text{SEARCH}(q_{key}))$ // *Hybrid Retrieval*
$CurrentFrontier, Visited \leftarrow S_{anchor}$
$\mathbf{w}_{T_q} \leftarrow \text{GETATTENTIONWEIGHTS}(T_q)$
for $d \leftarrow 1$ **to** $MaxDepth$ **do**
  $Candidates \leftarrow \text{PRIORITYQUEUE}()$
  for $u \in CurrentFrontier$ **do**
    for $v \in G.\text{NEIGHBORS}(u)$ **do**
      if $v \notin Visited$ **then**
        // *Calculate transition score via Eq. 5*
        $s_{uv} \leftarrow \exp(\lambda_1(\mathbf{w}_{T_q}^{\top} \cdot \mathbf{1}_{e_{uv}}) + \lambda_2 \text{sim}(\vec{v}, \vec{q}))$
        $score_v \leftarrow \text{score}_u \cdot \gamma + s_{uv}$ // *Apply Decay* γ
        $Candidates.\text{PUSH}(v, score_v)$
      end if
    end for
  end for
  $CurrentFrontier \leftarrow Candidates.\text{TOPK}(BeamWidth)$
  $Visited.\text{ADDALL}(CurrentFrontier)$
  if $Visited.\text{SIZE}() \geq Budget$ **then break**
  end if
end for
$C_{sorted} \leftarrow \text{TOPOLOGICALSORT}(Visited, T_q)$
return $\text{SERIALIZE}(C_{sorted})$

$$C_{prompt} = \bigoplus_{n_i \in \text{Sort}(\mathcal{G}_{sub})} [\texttt{<t:}\tau_i\texttt{>}\ n_i.content\ \texttt{<ref:}n_i.id\texttt{>}] \tag{7}$$

where $\bigoplus$ denotes string concatenation.

3. Salience-Based Token Budgeting: Given a fixed LLM context window, we cannot include all retrieved nodes. We utilize the relevance scores $S(n_j|n_i, q)$ computed in Eq. (5) to enforce a dynamic budget. Low-probability nodes are summarized into brevity codes (e.g., "...3 intermediate

Algorithm 2 Fast Path: Synaptic Ingestion

Input: User Interaction I, Current Graph $\mathcal{G}_t$
Output: Updated Graph $\mathcal{G}_{t+1}$

$n_t \leftarrow \text{SEGMENTEVENT}(I)$
$n_{prev} \leftarrow \text{GETLASTNODE}(\mathcal{G}_t)$
// Update Temporal Backbone
$\mathcal{G}.\text{ADDEDGE}(n_{prev}, n_t, \text{type} = \text{TEMP})$
// Indexing
$\mathbf{v}_t \leftarrow \text{ENCODER}(n_t.c)$
$VDB.\text{ADD}(\mathbf{v}_t, n_t.id)$
$Queue.\text{ENQUEUE}(n_t.id)$ *// Trigger Slow Path*
return n_t

Algorithm 3 Slow Path: Structural Consolidation

Worker Process:
loop
  $id \leftarrow Queue.\text{DEQUEUE}()$
  if id is **null then continue**
  end if
  $n_t \leftarrow \mathcal{G}.\text{GETNODE}(id)$
  $\mathcal{N}_{local} \leftarrow \mathcal{G}.\text{GETNEIGHBORHOOD}(n_t, \text{hops} = 2)$
  // Infer latent Causal and Entity structures
  $Prompt \leftarrow \text{FORMAT}(\mathcal{N}_{local})$
  $\mathcal{E}_{new} \leftarrow \Phi_{LLM}(Prompt)$
  $\mathcal{G}.\text{ADDEDGES}(\mathcal{E}_{new})$
end loop

events..."), while high-salience nodes retain full semantic detail.

This structured scaffold forces the LLM to act as an interpreter of evidence rather than a creative writer, significantly reducing grounding errors.

3.4 Memory Evolution (Write and Update)

Long-term reasoning requires not only effective retrieval, but also a memory substrate that can adapt and reorganize as experience accumulates. MAGMA addresses this requirement through a structured memory evolution scheme that incrementally refines its multi-relational graph over time. Specifically, the transition from $\mathcal{G}_t$ to $\mathcal{G}_{t+1}$ is governed by a dual-stream process that decouples latency-sensitive ingestion from compute-intensive consolidation (Kumaran et al., 2016), balancing short-term responsiveness with long-term reasoning fidelity.

Fast path (synaptic ingestion): The Fast Path operates on the critical path of interaction, constrained by strict latency requirements. It performs non-blocking operations: event segmentation, vector indexing, and updating the immutable temporal backbone ($n_{t-1} \rightarrow n_t$). As detailed in Algorithm 2, no blocking LLM reasoning occurs here, ensuring the agent remains responsive regardless of memory size.

Slow path (structural consolidation): Asynchronously, the slow path performs Memory Consolidation (Algorithm 3). It functions as a background worker that dequeues events and densifies the graph structure. By analyzing the local neighborhood $\mathcal{N}(n_t)$ of recent events, the system employs an LLM Φ to infer latent connections:

$$\mathcal{E}_{new} = \Phi_{reason}(\mathcal{N}(n_t), \mathcal{H}_{history}) \quad (8)$$

This process constructs high-value $\mathcal{E}_{causal}$ and $\mathcal{E}_{ent}$ links, effectively trading off compute time for relational depth.

3.5 Implementation

We implement MAGMA as a modular three-layer architecture designed for extensibility, scalability, and deployment flexibility. The storage layer abstracts over heterogeneous physical backends, providing unified interfaces for managing the typed memory graph, dense vector indices, and sparse keyword indices. This abstraction cleanly separates the logical memory model from its physical realization, enabling seamless substitution of storage backends (e.g., in-memory data structures versus production-grade graph or vector databases) with minimal engineering effort.

The retrieval layer coordinates the core algorithmic components, including memory construction, multi-stage ranking, and policy-guided graph traversal. It is supported by specialized utility modules for episodic segmentation and temporal normalization, which provide structured signals to downstream retrieval and traversal policies. The application layer manages the interaction loop, evaluation harnesses, and prompt construction, serving as the interface between the agent and the underlying memory system.

4 Experiments

We conduct comprehensive experiments to evaluate both the reasoning effectiveness and systems properties of the proposed MAGMA architecture over state-of-the-art baselines.

4.1 Experimental Setup

Datasets. We evaluate long-term conversational capability using two widely adopted benchmarks: **(1) LoCoMo** (Maharana et al., 2024): which contains ultra-long conversations (average length of 9K tokens) designed to assess long-range temporal and causal retrieval. **(2) LongMemEval** (Wu et al., 2024): a large-scale stress-test benchmark with

Table 1: Performance on the LoCoMo benchmark evaluated using the LLM-as-a-Judge metric. Higher scores indicate better performance. LLM model is based on gpt-4o-mini.

Method	Multi-Hop	Temporal	Open-Domain	Single-Hop	Adversarial	Overall
Full Context	0.468	0.562	0.486	0.630	0.205	0.481
A-MEM	0.495	0.474	0.385	0.653	0.616	0.580
MemoryOS	0.552	0.422	0.504	0.674	0.428	0.553
Nemori	**0.569**	0.649	0.485	0.764	0.325	0.590
MAGMA (ours)	0.528	**0.650**	**0.517**	**0.776**	**0.742**	**0.700**

an average context length exceeding 100K tokens, used to evaluate scalability and memory retention stability over extended interaction horizons..

Baselines. We compare MAGMA against four state-of-the-art memory architectures. For fair comparison, all methods employ the same backbone LLMs.

- **Full Context**: Feeds the entire conversation history into the LLM.
- **A-MEM** (Xu et al., 2025): A biological-inspired, self-evolving memory system that dynamically organizes agent experiences.
- **Nemori** (Nan et al., 2025): A graph-based memory utilizing a "predict-calibrate" mechanism for episodic segmentation.
- **MemoryOS**(Kang et al., 2025a) : A semantic-focused memory operating system employing a hierarchical storage strategy.

Metrics. Following standard evaluation protocols, we primarily use the LLM-as-a-Judge score (Zheng et al., 2023) to assess the accuracy of different methods. The detailed evaluation prompt used for the judge model is provided in the appendix. For completeness, we also report token-level F1 and BLEU-1 (Papineni et al., 2002).

4.2 Overall Comparison

This section introduces the accuracy performance comparison between all methods on the LoCoMo benchmark based on LLM-as-a-judge. As shown in Table 1, MAGMA achieves the highest overall judge score of 0.7, substantially outperforming the other baselines: Full Context (0.481), A-MEM (0.58), MemoryOS (0.553) and Nemori (0.59) by relative margins of 18.6% to 45.5%. This result demonstrates that explicitly modeling multi-relational structure enables more accurate long-horizon reasoning than flat or purely semantic memory architectures.

A closer analysis reveals that MAGMA's advantage is particularly pronounced in reasoning-intensive settings. In the Temporal category, MAGMA slightly but consistently outperforms others (Judge: 0.650 for MAGMA vs. 0.422 - 0.649 for others), validating the effectiveness of our Temporal Inference Engine in resolving relative temporal expressions into grounded chronological representations. The performance gap further widens under adversarial conditions, where MAGMA attains a judge score of 0.742. This robustness stems from the Adaptive Traversal Policy, which prioritizes causal and entity-consistent paths and avoids semantically similar yet structurally irrelevant distractors that often mislead vector-based retrieval systems. Additional results and analyzes, including case studies and evaluations under alternative metrics, are provided in the appendix.

4.3 Generalization Study

To evaluate generalization under extreme context lengths, we compare MAGMA against prior methods on the LongMemEval benchmark. LongMemEval poses a substantial scalability challenge, with an average context length exceeding 100k tokens, and therefore serves as a rigorous stress test for long-term memory retention and retrieval under strict computational constraints.

As summarized in Table 2, MAGMA achieves the highest average accuracy (61.2%), outperforming both the Full-context baseline (55.0%) and the Nemori system (56.2%). These results indicate that MAGMA generalizes effectively to ultra-long interaction histories while maintaining strong retrieval precision.

At the same time, the results highlight a favorable efficiency–granularity trade-off. Although the Full-context baseline performs strongly on *single-session-assistant* tasks (89.3%), this performance comes at a prohibitive computational cost, requiring over 100k tokens per query. MAGMA achieves competitive accuracy (83.9%) while using only 0.7k–4.2k tokens per query, representing a reduction of more than 95%. This demonstrates that

Table 2: Performance comparison on LongMemEval dataset across different question types. We compare our MAGMA method against the Full-context baseline and the Nemori system.

	Question Type	Full-context (101K tokens)	Nemori (3.7–4.8K tokens)	MAGMA (0.7–4.2K tokens)
gpt-4o-mini	single-session-preference	6.7%	62.7%	**73.3%**
	single-session-assistant	**89.3%**	73.2%	83.9%
	temporal-reasoning	42.1%	43.0%	**45.1%**
	multi-session	38.3%	**51.4%**	50.4%
	knowledge-update	78.2%	52.6%	**66.7%**
	single-session-user	**78.6%**	77.7%	72.9%
	Average	55.0%	56.2%	**61.2%**

Table 3: System efficiency comparison with total memory build time (in hours), average token consumption per query (in k tokens), and average query latency (in seconds).

Method	Build Time (h)	Tokens/Query (k)	Latency (s)
Full Context	N/A	8.53	1.74
A-MEM	1.01	**2.62**	2.26
MemoryOS	0.91	4.76	32.68
Nemori	**0.29**	3.46	2.59
MAGMA	0.39	3.37	**1.47**

Table 4: Breakdown analysis on the performance impact of different schemes in MAGMA.

MAGMA schemes	Judge	F1	BLEU-1
w/o Adaptive Policy	0.637	0.413	0.357
w/o Causal Links	0.644	0.439	0.354
w/o Temporal Backbone	0.647	0.438	0.349
w/o Entity Links	0.666	0.451	0.363
MAGMA (Full)	**0.700**	**0.467**	**0.378**

MAGMA effectively compresses long interaction histories into compact, reasoning-dense subgraphs, preserving essential information while substantially reducing inference-time overhead.

4.4 System Efficiency Analysis

To evaluate the system efficiency of MAGMA, two metrics are focused: (1) memory build time (the time required to construct the memory graph) and (2) token cost (the average tokens processed per query).

Table 3 reports the comparative results. While A-MEM achieves the lowest token consumption (2.62k) due to its aggressive summarization, it sacrifices reasoning depth (see Table 1). In contrast, MAGMA achieves the lowest query latency (1.47s) about 40% faster than the next best retrieval baseline (A-MEM) while maintaining a competitive token cost (3.37k). This efficiency stems from our Adaptive Traversal Policy, which prunes irrelevant subgraphs early, and the dual-stream architecture that offloads complex indexing to the background.

4.5 Ablation Study

In this subsection, we conduct a systematic ablation study to assess the contribution of individual components in MAGMA. By selectively disabling edge types and traversal mechanisms, we isolate the sources of its reasoning capability. The results in Table 4 reveal three main findings.

First, removing the Adaptive Policy results in the largest performance drop, with the Judge score decreasing from 0.700 to 0.637. This confirms that intent-aware routing is critical: without it, retrieval degenerates into a static graph walk that introduces structurally irrelevant information and degrades reasoning quality. Second, removing either Causal Links or the Temporal Backbone leads to comparable and substantial performance losses (0.644 and 0.647, respectively), indicating that causal structure and temporal ordering provide complementary, non-substitutable axes of reasoning. Finally, removing Entity Links causes a smaller but consistent decline (0.700 to 0.666), highlighting their role in maintaining entity permanence and reducing hallucinations in entity-centric queries.

To further isolate the contribution of each relation type, we additionally conducted a single-graph-only ablation on LoCoMo, shown in Table 5. The results are consistent with the leave-one-out findings in Table 4. Among single-graph variants, *Causal Only* achieves the highest overall score (0.590), suggesting that causal relations provide strong logical filtering and robustness against distractor noise. *Temporal Only* performs best on temporal questions (0.620), confirming that explicit temporal structure is particularly important for sequential reasoning. In contrast, *Entity Only* obtains the lowest overall score (0.531): while it remains

Table 5: Single-graph ablation study on LoCoMo.

Graph Configuration	Multi-Hop	Temporal	Open-Domain	Single-Hop	Adversarial	Overall
Causal Only	0.470	0.460	0.430	0.650	0.680	0.590
Temporal Only	0.440	0.620	0.450	0.650	0.520	0.577
Entity Only	0.485	0.420	0.460	0.640	0.450	0.531
Full MAGMA	**0.528**	**0.650**	**0.517**	**0.776**	**0.742**	**0.700**

helpful for concept bridging in multi-hop reasoning, it lacks both timeline awareness and logical filtering, which explains why removing entity links causes the smallest drop in the full-system ablation.

Overall, these results show that no single relation type is sufficient to recover MAGMA's full reasoning capability, as all single-graph variants remain below 0.60 overall. By explicitly decoupling causal, temporal, and entity relations and combining them with adaptive traversal, MAGMA leverages their complementary strengths to achieve the best overall performance.

5 Conclusion

We introduced MAGMA, a multi-graph agentic memory architecture that models semantic, temporal, causal, and entity relations within a unified yet disentangled memory substrate. By formulating retrieval as a policy-guided graph traversal and decoupling memory ingestion from asynchronous structural consolidation, MAGMA enables effective long-horizon reasoning while maintaining low inference-time latency. Empirical results on LoCoMo and LongMemEval demonstrate that MAGMA consistently outperforms state-of-the-art memory systems while achieving substantial efficiency gains under ultra-long contexts.

6 Limitations

While MAGMA demonstrates strong empirical performance, it has several limitations. First, the quality of the constructed memory graph depends on the reasoning fidelity of the underlying Large Language Models used during asynchronous consolidation. This dependency is a shared limitation of agentic memory systems that rely on LLM-based structural inference, as they are susceptible to extraction errors and hallucinations (Pan et al., 2024; Xi et al., 2025; Wadhwa et al., 2023). Although MAGMA employs structured prompts and conservative inference thresholds to reduce spurious links, erroneous or missing relations may still arise and propagate to downstream retrieval. Nevertheless, our experimental results indicate that, even under these constraints, agentic memory systems such as MAGMA substantially outperform traditional baselines, including full-context approaches, in long-horizon reasoning tasks.

Second, multi-graph substrate may introduce additional storage and engineering complexity compared to flat, vector-only memory systems. Maintaining multiple relational views and dual-stream processing incurs a little higher implementation and memory overhead, which may limit applicability in highly resource-constrained environments.

Finally, most existing agentic memory systems, including MAGMA, are primarily evaluated on long-context conversational and agentic benchmarks such as LoCoMo and LongMemEval. While these benchmarks effectively stress temporal and causal reasoning, they do not cover the full range of settings in which agentic memory may be required(Hu et al., 2025; Jiang et al., 2026). Extending MAGMA to other scenarios, such as multimodal agents or environments with heterogeneous observation streams, may require additional adaptation and calibration. Addressing these broader evaluation settings remains an important research direction for future work.

7 Acknowledgment

This work was partially supported by NSF 2343863, 2413520, 2417747 and 2440611. Any opinions, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020a. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020b. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

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

Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. 2009. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, page 758–759, New York, NY, USA. Association for Computing Machinery.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024a. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024b. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

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. *arXiv preprint arXiv:2502.14802*.

Yuyang Hu, Shichun Liu, Yanwei Yue, Guibin Zhang, Boyang Liu, Fangyi Zhu, Jiahang Lin, Honglin Guo, Shihan Dou, Zhiheng Xi, and 1 others. 2025. Memory in the age of ai agents. *arXiv preprint arXiv:2512.13564*.

Le Huang, Hengzhi Lan, Zijun Sun, Chuan Shi, and Ting Bai. 2024. Emotional rag: Enhancing role-playing agents through emotional retrieval. In *2024 IEEE International Conference on Knowledge Graph (ICKG)*, pages 120–127. IEEE.

Dongming Jiang, Yi Li, Songtao Wei, Jinxin Yang, Ayushi Kishore, Alysa Zhao, Dingyi Kang, Xu Hu, Feng Chen, Qiannan Li, and 1 others. 2026. Anatomy of agentic memory: Taxonomy and empirical analysis of evaluation and system limitations. *arXiv preprint arXiv:2602.19320*.

Wenqi Jiang, Suvinay Subramanian, Cat Graves, Gustavo Alonso, Amir Yazdanbakhsh, and Vidushi Dadu. 2025. Rago: Systematic performance optimization for retrieval-augmented generation serving. In *Proceedings of the 52nd Annual International Symposium on Computer Architecture*, pages 974–989.

Ziyan Jiang, Xueguang Ma, and Wenhu Chen. 2024. Longrag: Enhancing retrieval-augmented generation with long-context llms. *arXiv preprint arXiv:2406.15319*.

Zhijing Jin, Yuen Chen, Felix Leeb, Luigi Gresele, Ojasv Kamal, Zhiheng Lyu, Kevin Blin, Fernando Gonzalez Adauto, Max Kleiman-Weiner, Mrinmaya Sachan, and 1 others. 2023. Cladder: Assessing causal reasoning in language models. *Advances in Neural Information Processing Systems*, 36:31038–31065.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

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

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

Jikun Kang, Wenqi Wu, Filippos Christianos, Alex J Chan, Fraser Greenlee, George Thomas, Marvin Purtorab, and Andy Toulis. 2025c. Lm2: Large memory models. *arXiv preprint arXiv:2502.06049*.

Urvashi Khandelwal, He He, Peng Qi, and Dan Jurafsky. 2018. Sharp nearby, fuzzy far away: How neural language models use context. *arXiv preprint arXiv:1805.04623*.

Emre Kiciman, Robert Ness, Amit Sharma, and Chenhao Tan. 2023. Causal reasoning and large language models: Opening a new frontier for causality. *Transactions on Machine Learning Research*.

Dharshan Kumaran, Demis Hassabis, and James L McClelland. 2016. What learning systems do intelligent agents need? complementary learning systems theory updated. *Trends in cognitive sciences*, 20(7):512–534.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Yi Li, Lianjie Cao, Faraz Ahmed, Puneet Sharma, and Bingzhe Li. 2026. Hippocampus: An efficient and scalable memory module for agentic ai. *arXiv preprint arXiv:2602.13594*.

Zhiyu Li, Shichao Song, Chenyang Xi, Hanyu Wang, Chen Tang, and 1 others. 2025. Memos: A memory os for ai system. *arXiv preprint arXiv:2507.03724*.

Shuhang Lin, Zhencan Peng, Lingyao Li, Xiao Lin, Xi Zhu, and Yongfeng Zhang. 2025. Cache mechanism for agent rag systems. *arXiv preprint arXiv:2511.02919*.

Lei Liu, Xiaoyan Yang, Yue Shen, Binbin Hu, Zhiqiang Zhang, Jinjie Gu, and Guannan Zhang. 2023. Think-in-memory: Recalling and post-thinking enable llms with long-term memory. *arXiv preprint arXiv:2311.08719*.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. 2024. Evaluating very long-term conversational memory of llm agents. *arXiv preprint arXiv:2402.17753*.

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

Charles Packer, Vivian Fang, Shishir_G Patil, Kevin Lin, Sarah Wooders, and Joseph_E Gonzalez. 2023. Memgpt: Towards llms as operating systems.

Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. 2024. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3580–3599.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22.

Ofir Press, Noah A Smith, and Mike Lewis. 2021a. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*.

Ofir Press, Noah A Smith, and Mike Lewis. 2021b. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*.

Hongjin Qian, Zheng Liu, Peitian Zhang, Kelong Mao, Defu Lian, Zhicheng Dou, and Tiejun Huang. 2025. Memorag: Boosting long context processing with global memory-enhanced retrieval augmentation. In *Proceedings of the ACM on Web Conference 2025*, pages 2366–2377.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. 2025. Zep: a temporal knowledge graph architecture for agent memory. *arXiv preprint arXiv:2501.13956*.

Somin Wadhwa, Silvio Amir, and Byron C Wallace. 2023. Revisiting relation extraction in the era of large language models. In *Proceedings of the conference. association for computational linguistics. meeting*, volume 2023, page 15566.

Yu Wang and Xi Chen. 2025. Mirix: Multi-agent memory system for llm-based agents. *arXiv preprint arXiv:2507.07957*.

Zheng Wang, Shu Teo, Jieer Ouyang, Yongjun Xu, and Wei Shi. 2024. M-rag: Reinforcing large language model performance through retrieval-augmented generation with multiple partitions. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1966–1978.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. 2024. Longmemeval: Benchmarking chat assistants on long-term interactive memory. *arXiv preprint arXiv:2410.10813*.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, and 1 others. 2025. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101.

Shijia Xu, Zhou Wu, Xiaolong Jia, Yu Wang, Kai Liu, and April Xiaowen Dong. 2026. Self-correcting rag: Enhancing faithfulness via mmkp context selection and nli-guided mcts. *Preprint*, arXiv:2604.10734.

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

Yiming Zeng, Wanhao Yu, Zexin Li, Tao Ren, Yu Ma, Jinghan Cao, Xiyan Chen, and Tingting Yu. 2025. Bridging the editing gap in llms: Fineedit for precise and targeted text modifications. *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 2193–2206.

Zhuosheng Zhang, Yao Yao, Aston Zhang, Xiangru Tang, Xinbei Ma, Zhiwei He, Yiming Wang, Mark Gerstein, Rui Wang, Gongshen Liu, and 1 others.

2025. Igniting language intelligence: The hitchhiker's guide from chain-of-thought reasoning to language agents. *ACM Computing Surveys*, 57(8):1–39.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623.

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

A Related Work

Following the framing in main text, we organize related work along the same progression: from context window extension to retrieval augmented generation (RAG) and finally to memory augmented generation (MAG), and then discuss structured/graph memories and causal reasoning, which are central to long-horizon agentic interactions.

Context-window Extension. A direct line of work extends the effective context length of Transformers by modifying attention or positional extrapolation. Longformer (Beltagy et al., 2020b) introduces sparse attention patterns to scale to long documents, reducing quadratic cost while retaining locality and selected global connectivity. ALiBi (Press et al., 2021b) (Attention with Linear Biases) enables length extrapolation by injecting distance-aware linear biases into attention scores, improving robustness when testing on longer sequences than those seen in training. Recent efforts also add explicit memory modules or hybrid mechanisms to push beyond pure attention-window scaling. For example, LM2 (Kang et al., 2025c) proposes a decoder-only architecture augmented with an auxiliary memory to mitigate long-context limitations. MemoRAG (Qian et al., 2025) similarly emphasizes global-memory-enhanced retrieval to boost long-context processing when raw context is insufficient or inefficient. While these approaches improve long-range coverage, they do not, by themselves, address the continual, evolving, and write-back nature of agent memory required for multi-session interactions.

Retrieval Augmented Generation. RAG (Lewis et al., 2020) augments an LLM with external retrieval over a fixed corpus, classically retrieving supporting passages and conditioning generation on them. Subsequent work explores better integration with long-context models and more scalable retrieval pipelines. LongRAG (Jiang et al., 2024) studies how to exploit long-context LLMs together with retrieval, improving the ability to incorporate larger retrieved evidence sets. Other systems focus on structuring the retrieved memory space or optimizing the RAG serving stack: M-RAG (Wang et al., 2024) uses multiple partitions to encourage fine-grained retrieval focus, while RAGO (Jiang et al., 2025) provides a systematic framework for performance optimization in RAG serving. Furthermore, to improve the reliability of retrieved context, approaches like Self-Correcting RAG (Xu et al., 2026) enhance generation faithfulness via selective context filtering and inference-guided search. However, standard RAG typically assumes a static knowledge base. In contrast, agentic settings require memory that is continuously updated (the feedback loop described in the main text). This motivates the shift to MAG systems, where memory is dynamic and evolves with interaction histories.

Memory Augmented Generation and Agent Memory Systems. MAG systems maintain and update an external memory over time, enabling agents to accumulate knowledge, preserve identity, and remain coherent across sessions. Early and representative directions include memory construction and write-back strategies for long-term agent behavior, such as MemoryBank (Zhong et al., 2024) and generative agents style architectures that emphasize persistent profiles and evolving state grounded in past interactions (Nan et al., 2025; Maharana et al., 2024; Zeng et al., 2025). A growing body of work adopts systems metaphors and designs: MemGPT (Packer et al., 2023) frames LLM agents with an operating-system-like memory hierarchy, emphasizing paging and controlled context management. More recent memory OS systems propose explicit storage hierarchies, efficient memory modules, and controllers (e.g., MemoryOS (Kang et al., 2025b), MemOS (Li et al., 2025), and Hippocampus (Li et al., 2026)) to manage persistence, scalability, and retrieval policies at scale. In addition, practical agent-memory stacks (e.g., Zep (Rasmussen et al., 2025)) offer temporal knowledge-graph-based memory services aimed at real-world deployment constraints.

Structured memory: chains-of-thought and graph-based representations. Beyond flat text buffers or vector stores, several methods explicitly structure memory to support reasoning. Think-in-

Memory (TiM) stores evolving chains-of-thought to improve consistency across long-horizon reasoning, while A-MEM (Xu et al., 2025) is inspired by Zettelkasten-style linking of notes/experiences. These methods highlight the value of representing intermediate reasoning traces or explicit links, but many retrieval pipelines still predominantly rely on semantic similarity as the primary access mechanism. Graph-based approaches have recently gained traction as a way to capture cross-document and cross-episode dependencies. GraphRAG (Edge et al., 2024b) builds entity-centric graphs and community summaries to answer more global questions over large corpora. Zep proposes a temporally-aware knowledge-graph engine (Graphiti) that synthesizes conversational and structured business data while preserving historical relations. The main text notes these graph-based lines explicitly and motivates a key gap: many systems organize memory around associative proximity (semantic relatedness) rather than mechanistic dependency.

Causal reasoning and long-horizon evaluation. Causal reasoning has been highlighted as both important and challenging for LLMs. The work (Kiciman et al., 2023) study LLMs' ability to generate causal arguments across multiple causal tasks and emphasize robustness/failure modes, reinforcing that what happened retrieval is not sufficient for why reasoning in many settings. Benchmarking efforts such as LoCoMo (Maharana et al., 2024) stress long-range temporal and causal dynamics in multi-session conversations and provide evaluation tasks that expose long-horizon memory deficits. The paper's experimental setup also uses LongMemEval (Wu et al., 2024) as an ultra-long context stress test, and evaluates via LLM-as-a-Judge protocols standard in modern instruction-following evaluation. Overall, prior work demonstrates steady progress in (i) scaling context length, (ii) improving retrieval pipelines, and (iii) building structured, evolving memories for agents. The main text positions MAGMA within this trajectory by explicitly targeting multi-relational structure (semantic/temporal/causal/entity) and intent-aware retrieval control.

B System Implementation Details

B.1 Hyperparameter Configuration

Table 6 presents the comprehensive configuration used in our experiments. These parameters were empirically optimized on the LoCoMo benchmark. Notably, MAGMA employs an *Adaptive Scoring* mechanism where weights (λ) shift dynamically based on the detected query intent.

Table 6: Hyperparameter settings for MAGMA. "Traversal Weights" correspond to the intent-specific vector $\mathbf{w}_{T_q}$, while λ_1 and λ_2 control the global balance between structural alignment and semantic affinity (Eq. 5).

Module	Parameter	Value/Range
Embedding	Model (Default)	`all-MiniLM-L6-v2`
	Model (Optional)	`text-embedding-3-small`
	Dimension	384 / 1536
Inference	LLM Backbone	`gpt-4o-mini`
	Temperature	0.0
Retrieval (Phase 1)	RRF Constant (k)	60
	Vector Top-K	20
	$w_{keyword}$ (Fusion)	2.0 – 5.0
	Sim. Threshold	0.10–0.30
Traversal (Phase 2)	Max Depth	5 hops
	Max Nodes	200
	Drop Threshold	0.15
Adaptive Weights	λ_1 (Structure Coef.)	1.0 (Base)
	λ_2 (Semantic Coef.)	0.3 – 0.7
	w_{entity} (in $\mathbf{w}_{T_q}$)	2.5 – 6.0
	$w_{temporal}$ (in $\mathbf{w}_{T_q}$)	0.5 – 4.0
	w_{causal} (in $\mathbf{w}_{T_q}$)	3.0 – 5.0
	w_{phrase} (in $\mathbf{w}_{T_q}$)	2.5 – 5.0

C Prompt Library

MAGMA employs a sophisticated prompt strategy with three distinct types, each optimized for specific cognitive tasks within the memory pipeline.

C.1 Event Extraction Prompt (JSON-Structured)

To ensure robustness against hallucination and parsing errors, this module employs a strict JSON schema enforcement strategy. The prompt explicitly defines the extraction targets to ensure downstream graph integrity, capturing not just entities but also semantic relationships and temporal markers.

System Prompt: Event Extractor

System Role: You are an automated Graph Memory Parser. Your task is to extract structured metadata from raw conversational logs to build a knowledge graph.
Input Data:
- Speaker: {speaker}
- Text: {text}
- Context: {prev_summary}

Instructions: Analyze the input and return **ONLY** a valid JSON object matching the specific schema below. Do not include markdown formatting.
Target Schema:
- `"entities"`: List of proper nouns (People, Locations, Organizations).
- `"topic"`: String (1–3 words representing the main theme).

- `"relationships"`: List of strings describing interactions (e.g., "X researches Y").
- `"semantic_facts"`: List of atomic facts preserving key information.
- `"dates_mentioned"`: List of temporal strings (e.g., "next Friday", "2024-01-01").
- `"summary"`: One-sentence summary preserving speaker attribution.

C.2 Query-Adaptive QA Prompt

The generation prompt begins with a strict persona definition and appends specific reasoning instructions dynamically based on the Router's classification (e.g., Multi-hop, Temporal, Open-domain).

System Prompt: Adaptive QA

System Role: You are a precision QA assistant operating on retrieved memory contexts. Your goal is to answer the user's question accurately using *only* the provided information.
Context: {context}
Current Query:
- Question: {question}
- Constraints: {category_specific_constraints}

Instructions:
1. Use ONLY information explicitly stated in the context.
2. If the answer is not present, respond exactly with "Information not found".
3. Be concise (typically 1–10 words) unless detailed reasoning is required.
4. **{dynamic_instruction}** *// Automatically generated by our engine's query classifier/router (no oracle labels)*

Answer:

***Dynamic Instruction Injection Candidates:**

- **[Multi-hop]:** "Connect related facts across different nodes. For comparison queries (e.g., 'both/all'), identify commonalities between entities rather than listing individual details."
- **[Temporal]:** "Resolve relative dates (e.g., 'yesterday') using the event timestamps. Output dates strictly in 'D Month YYYY' format. Calculate durations if asked."
- **[Open-Domain/Inference]:** "Make reasonable inferences based on the user's personality traits, interests, and past behaviors. Support hypothetical ('would/could') reasoning with evidence."
- **[Single-hop/Factual]:** "Extract the specific entity, name, or method requested. Do not add explanations. Return the exact fact matching the query intent."

C.3 Evaluation Prompt (LLM-as-a-Judge)

To ensure rigorous evaluation beyond simple n-gram overlapping, we employ a semantic scoring mechanism. The Judge LLM evaluates the alignment between the generated response and the ground truth using the following schema.

System Prompt: Semantic Grader

You are an expert evaluator assessing the semantic fidelity of a memory retrieval system. Score the `Candidate Answer` against the `Gold Reference` on a continuous scale [0.0, 1.0].
Scoring Rubric:
- **1.0 (Exact Alignment):** Captures all key entities, temporal markers, and causal relationships. Semantically equivalent.
- **0.8 (Substantially Correct):** Main point is accurate but lacks minor nuances or secondary details.
- **0.6 (Partial Match):** Contains valid information but misses key constraints (e.g., wrong date but correct event).
- **0.4 (Tangential):** Touches on the topic but misses the core information requirement.
- **0.2 (Incoherent):** Factually incorrect with only minimal topical overlap.
- **0.0 (Contradiction/Hallucination):** Completely unrelated or contradicts the ground truth.

Evaluation Constraints:
1. **Temporal Flexibility:** Accept relative time references (e.g., "next Tuesday") if they resolve to the same period as the Gold Reference.
2. **Semantic Equivalence:** Prioritize informational content over lexical matching.
3. **Adversarial Handling:** If the Gold Reference states "Unanswerable", the Candidate MUST explicitly state lack of information. Any hallucinated fact results in 0.0.

Input: Question: {question} | Gold: {gold} | Candidate: {generated}
Output: JSON `{"score": float, "reasoning": "concise explanation"}`

D Baseline Configurations

To ensure a fair and rigorous comparison, we standardized the experimental environment across all systems. Specifically, we adhered to the following protocols:

- **Full Context Baseline:** We implemented a "Full Context" baseline where the entire available conversation history is fed directly into the LLM's context window (up to the 128k token limit of `gpt-4o-mini`). This serves as a "brute-force" reference to evaluate the model's native long-context capabilities without external retrieval mechanisms.

- **Retrieval-Based Baselines:** For all baseline systems (e.g., AMem, Nemori, MemoryOS), we applied their official default hyperparameters and storage settings to reflect their standard out-of-the-box performance.

- **Unified Backbone Model:** To eliminate performance variance caused by different foundation models, all systems utilized OpenAI's `gpt-4o-mini` for both retrieval reasoning and response generation.

- **Unified Evaluation:** All system outputs were evaluated using the identical *LLM-as-a-Judge* framework (also powered by `gpt-4o-mini` with temperature=0.0), as detailed in Appendix C.

Dataset Statistics. We conducted a comprehensive evaluation on the full LoCoMo benchmark, testing across all five cognitive categories to assess varying levels of retrieval complexity. The detailed distribution of query types is presented in Table 7.

Table 7: Distribution of query categories in the LoCoMo benchmark used for evaluation.

Query Category	Count
Single-Hop Retrieval	841
Adversarial	446
Temporal Reasoning	321
Multi-Hop Reasoning	282
Open Domain	96
Total Samples	**1,986**

E Case Study

To demonstrate MAGMA's reasoning capabilities across different cognitive modalities, we analyze three real-world scenarios from the LoCoMo benchmark. Table 8 provides a side-by-side comparison of MAGMA against key baselines (A-MEM, Nemori, MemoryOS).

E.1 Illustrative Walkthrough: From Memory Construction to Retrieval

To make the case study more concrete, we briefly illustrate how MAGMA processes the Melanie example end to end. Consider three representative facts from the conversation history: (1) Melanie mentions playing the *violin* in an earlier session, (2) she later says that she also plays the *clarinet*, and (3) in a later family-trip session, she refers to her *son*, *two children* in a photo, and a hike done *yesterday*.

During memory construction, MAGMA first segments these utterances into event nodes and places them along a temporal backbone. The system then incrementally enriches this memory with additional structure, such as semantic connections between related musical events, entity-centric links for recurring references to Melanie and her family members, and normalized temporal attributes for relative expressions such as *yesterday*. As a result, the memory is not stored as a flat list of text snippets, but as a small multi-view graph in which the same history can be accessed through different relational paths.

At query time, MAGMA first identifies the dominant retrieval intent and then selects anchor events before traversing the most relevant graph views. For example, for *"What instruments does Melanie play?"*, retrieval focuses on the entity and semantic views, allowing MAGMA to aggregate both the earlier *violin* mention and the later *clarinet* mention. For *"How many children does Melanie have?"*, retrieval centers on the local entity neighborhood, combining the references to a *son*, *two children*, and *brother* into a single evidence set. For *"When did she hike?"*, MAGMA relies primarily on the temporal view, using the normalized representation of *yesterday* to recover the grounded date.

This example highlights the key intuition behind MAGMA: memory construction organizes conversational history into complementary relational views, and query-time retrieval activates different parts of this structure depending on the reasoning need.

E.2 Detailed Analysis

Case 1: Overcoming Information Loss (Recall). For the query regarding instruments, A-MEM failed completely due to its summarization process abstracting away specific details ("violin") from early sessions. Other RAG baselines only retrieved the "clarinet" due to surface-level semantic matching. MAGMA, however, maintains an entity-centric graph structure. Instead of relying on rigid schemas, MAGMA queries the local neighborhood of the `[Entity: Melanie]` node. This allows it to capture diverse natural language predicates (e.g., "playing my violin", "started clarinet") and aggregate disjoint facts into a comprehensive answer, demonstrating robustness against information loss.

Table 8: Case study for failure analysis comparing MAGMA against baselines across three reasoning types. Red text indicates hallucinations or partial failures; **Teal text** indicates correct reasoning derived from graph traversal.

	Query & Type	Baseline Failure Mode	MAGMA Graph Reasoning (Success)
Fact	**Q1: Fact Retrieval** *"What instruments does Melanie play?"*	**A-MEM:** "Memories do not explicitly state..." **MemoryOS:** "Clarinet" ***Failure:*** Baselines relying on top-k vector search missed the distant memory of the "violin" (D2:5) because it appeared in a context about "me-time" rather than explicitly about music.	**"Clarinet and Violin."** ***Mechanism:*** MAGMA utilized the entity-centric subgraph around "Melanie". By traversing dynamic semantic edges (e.g., "playing", "enjoy") to connected event nodes, it aggregated all mentions of musical activities regardless of the specific relation label or distance.
Logic	**Q2: Logical Inference** *"How many children does Melanie have?"*	**Nemori:** "At least two..." **MemoryOS:** "Two" ***Failure:*** Baselines performed surface-level extraction from a photo description showing "two children" (D18:5), failing to account for the "son" mentioned in a separate accident event.	**"At least three."** ***Mechanism:*** MAGMA executed multi-hop inference focused on **Entity Resolution**: 1. Node A (Photo): Identified "two kids" entity. 2. Node B (Accident): Linked "son" (D18:1) via a dynamic relationship edge. 3. Node C (Dialogue): Confirmed "brother" (D18:7) is distinct from the two in the photo. $\rightarrow$ Logic: 2 (Photo) + 1 (Son/Brother) = 3.
Time	**Q3: Temporal Res.** *"When did she hike after the roadtrip?"*	**A-MEM:** "20 October 2023" **MemoryOS:** "29 December 2025" ***Failure:*** A-MEM simply copied the session timestamp. MemoryOS hallucinated a future date. Both failed to resolve the relative time expression.	**"19 October 2023"** ***Mechanism:*** MAGMA's Temporal Parser identified the relative marker "yesterday" in D18:17. Calculation: $T_{\text{session}}(Oct20) - 1\text{ day} = Oct19$. This exact date was anchored to the Event Node, allowing precise retrieval.

Case 2: Multi-Hop Reasoning vs. Surface Extraction. The query "How many children?" exposes a critical weakness in standard RAG: the inability to perform arithmetic across contexts. Baselines simply extracted the explicit mention of "two children" from a photo caption. In contrast, MAGMA treated this as a graph traversal problem focused on entity resolution. It queried the neighborhood of `[Entity: Melanie]` for connected nodes of type `Person`. By analyzing the semantic edges, specifically distinguishing the "two kids" entity in the canyon photo from the "son" entity involved in the car accident, MAGMA synthesized these distinct nodes. It correctly deduced that the "son" (referenced later as "brother") was an additional individual, summing up to a count of "at least three," a logical leap impossible for systems relying solely on vector similarity.

Case 3: Temporal Grounding. When asked "When did she hike?", baselines either hallucinated or defaulted to the conversation timestamp (Oct 20). This ignores the semantic meaning of the user's statement: "we just did it *yesterday*." MAGMA's structured ingestion pipeline normalizes relative dates during graph construction. The event was stored with the resolved attribute `date="2023-10-19"`, making the retrieval trivial and exact, completely bypassing the ambiguity that confused the LLM-based baselines.

F Metric Validation Analysis

To validate our choice of using an LLM-based Judge over traditional lexical metrics, we conducted a granular failure analysis on seven representative test cases. Table 10 details the quantitative breakdown.

F.1 Rationale for Semantic Scoring

Our empirical results reveal two critical failure modes where standard metrics (F1, BLEU-1) directly contradict human judgment:

1. **False Rewards** (The "Hallucination" Problem): Lexical metrics heavily reward incorrect answers that share surface-level tokens.
   - In Case 3, a direct negation ("compatible" vs. "*not* compatible") yields a remarkably high F1 of 0.857, treating a fatal contradiction as a near-perfect match.
   - In Case 6, substituting the wrong entity ("John" vs. "Sarah") still achieves **F1**

Table 9: LoCoMo evaluation with F1 and BLEU-1 metrics

	Multi-Hop		Temporal		Open-Domain		Single-Hop		Overall	
Method	F1	BLEU-1	F1	BLEU-1	F1	BLEU-1	F1	BLEU-1	F1	BLEU-1
Full Context	0.182	0.128	0.079	0.055	0.042	0.030	0.229	0.156	0.140	0.096
A-MEM	0.128	0.088	0.128	0.079	0.076	0.051	0.174	0.110	0.116	0.074
MemoryOS	0.365	0.276	0.434	0.369	0.246	0.191	0.493	0.437	0.413	0.355
Nemori	0.363	0.249	0.569	0.479	0.247	0.189	0.548	0.439	0.502	0.403
MAGMA (ours)	0.264	0.172	0.509	0.370	0.180	0.136	0.551	0.477	0.467	0.378

0.750, rewarding the hallucinatory output.

2. **False Penalties** (The "Phrasing" Problem): Valid answers with different formatting or synonyms are unfairly penalized.
    - In Case 4 (Time Notation) and Case 5 (Synonyms), F1 and BLEU scores drop to 0.000 despite the answers being semantically identical.

As shown in Table 10, the LLM-Judge correctly assigns a score of 0.0 to factual errors and 1.0 to semantic matches, aligning perfectly with reasoning requirements.

Table 10: Quantitative Failure Analysis of Lexical Metrics. We present seven controlled cases with their calculated F1 and BLEU-1 scores. The data demonstrates that lexical metrics frequently assign high scores to fatal errors (False Rewards) and zero scores to correct variations (False Penalties), whereas the LLM-Judge correctly assesses semantic validity.

Failure Mode	**Case Detail (Gold / Predicted)**	**Lexical Metrics** *(F1 / BLEU-1)*	**LLM Judge** *(Semantic)*
Case 1: False Reward *(Wrong Fact, High Overlap)*	**Gold:** "three items" **Pred:** "five items" ***Analysis:*** Factually wrong count, but rewarded for sharing the noun "items".	**High** 0.500 / 0.500	**0.0** (Reject)
Case 2: False Penalty *(Verbose Phrasing)*	**Gold:** "18 days" **Pred:** "The total duration was 18 days" ***Analysis:*** Correct answer penalized for low precision due to extra words.	**Low** 0.500 / 0.333	**1.0** (Accept)
Case 3: False Reward *(Negation/Contradiction)*	**Gold:** "compatible with Mac" **Pred:** "**not** compatible with Mac" ***Analysis:*** Fatal contradiction receives near-perfect scores due to high token overlap.	**Very High** 0.857 / 0.750	**0.0** (Reject)
Case 4: False Penalty *(Time Notation)*	**Gold:** "14:00" **Pred:** "2 PM" ***Analysis:*** Different formats result in zero overlap despite identical meaning.	**Zero** 0.000 / 0.000	**1.0** (Accept)
Case 5: False Penalty *(Synonyms)*	**Gold:** "cheap" **Pred:** "inexpensive" ***Analysis:*** Standard metrics cannot handle synonym matching without external resources.	**Zero** 0.000 / 0.000	**1.0** (Accept)
Case 6: False Reward *(Entity Hallucination)*	**Gold:** "John completed the project" **Pred:** "Sarah completed the project" ***Analysis:*** Wrong entity (Sarah vs John), yet high metrics due to shared sentence structure.	**High** 0.750 / 0.750	**0.0** (Reject)
Case 7: False Penalty *(Format Noise)*	**Gold:** "5" **Pred:** "5 (extracted from JSON...)" ***Analysis:*** Correct value embedded in noise results in poor precision metrics.	**Low** 0.286 / 0.167	**1.0** (Accept)