From Retrieval to Reconstruction: Constructing Evolvable Cognitive Memory for Long-term Dialogue

Anonymous ACL submission

Abstract

Large Language Models (LLMs) serving as long-term personal companions require memory systems that support complex reasoning over extended dialogues. However, existing Retrieval-Augmented Generation (RAG) frameworks treat memory as passive storage, suffering from semantic collapse—where subjective beliefs and objective facts become indistinguishable—and rigid navigation that fails to connect dispersed information. We introduce **CogMem**, a cognitive memory architecture that reconstructs dialogue context through explicit structure. CogMem organizes memory via the **PEC^{2}F (Person-Event-Concept-Claim-Fact)** schema, using dedicated **Claim nodes** to separate subjective attributions from objective records and prevent information contamination. Rather than static retrieval, CogMem employs a deterministic controller with four cognitive operators (anchoring, traversal, intersection, evidence grounding) that dynamically navigate the graph based on query intent. Comprehensive evaluation on **LoCoMo** and **LongMemEval** shows that CogMem achieves state-of-the-art performance across all task categories, surpassing neural memory systems and graph retrieval methods.These results demonstrate that structured cognitive modeling significantly improves long-term memory accuracy and robustness compared to passive vector retrieval.

1 Introduction

The evolution of Large Language Models (LLMs) from stateless text generators to autonomous personal agents has created an urgent need for persistent, high-quality memory (Wang et al., 2024; Xi et al., 2025). While recent advancements in scaling context windows (Team et al., 2024) allow models to buffer vast amounts of recent interaction, extended context alone cannot capture the structured, evolving nature of human biography. True long-term intelligence requires not just storage, but the ability to reconstruct narratives, extract key insights, and understand social contexts of multi-speaker environments (Tulving et al., 1972).

Despite this need, current approaches predominantly follow a paradigm of **passive retrieval**. Standard Retrieval-Augmented Generation (RAG) (Lewis et al., 2020) and early memory systems typically vectorize text chunks into a flat index. We argue that this "store-and-retrieve" paradigm has two key limitations. First, vector-based methods face **Semantic Collapse** (Mallen et al., 2023). In a dense vector space, an objective fact (e.g., "*Jon lost his job*") and a subjective opinion about it (e.g., "*Gina thinks Jon's job loss is a mistake*") often become very similar due to high semantic overlap. This leads to memory contamination, where subjective biases are mistakenly treated as objective facts, violating the principles of Theory of Mind (ToM) (Rabinowitz et al., 2018). Second, static retrieval pipelines lack **intentionality**. Complex dialogue reasoning often requires connecting information across time. A rigid retrieval algorithm cannot dynamically adjust its search strategy based on intermediate results, failing to perform the multi-hop reasoning required for deep understanding (Press et al., 2023).

To address these issues, we propose **CogMem**, a schema-aware cognitive architecture that shifts the design philosophy from passive retrieval to **agentic active recall**. Inspired by the Complementary Learning Systems (CLS) theory (Kumaran et al., 2016), CogMem unifies memory storage and reasoning through a novel **PEC^{2}F (Person-Event-Concept-Claim-Fact)** schema. To activate this structure, we implement a **Cognitive Search Agent** based on the ReAct framework (Yao et al., 2022). Instead of static matching, the agent uses graph operators—such as intersection and temporal scanning—to actively navigate the memory space, reconstructing answers through dynamic reasoning paths.

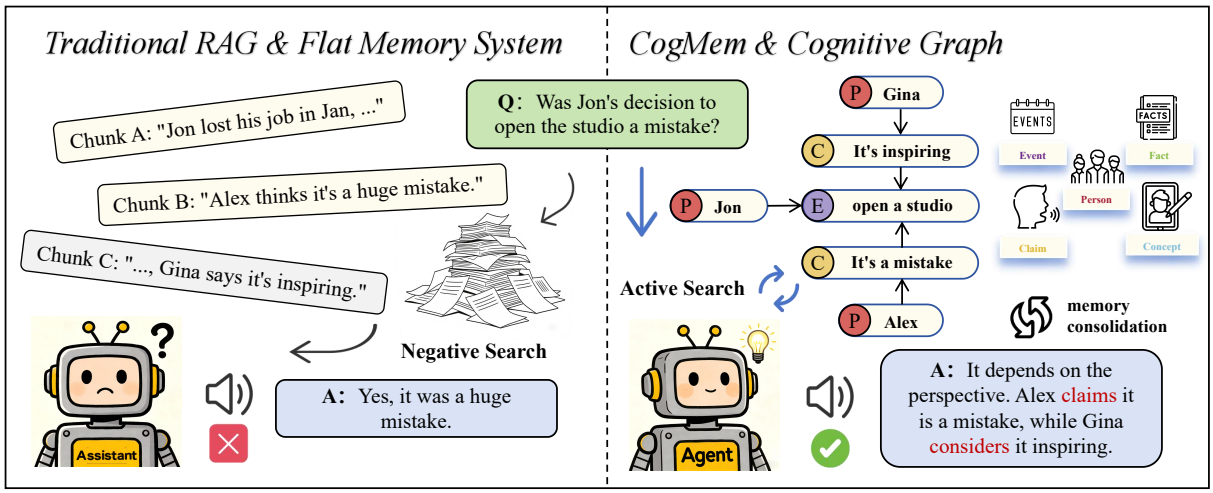

Figure 1: **The Paradigm Shift from Passive Retrieval to Active Reconstruction.** *Left(Traditional RAG & Flat Memory System)* Relies on Passive Retrieval over flat text chunks. The coarse granularity of chunks introduces noise and fails to retrieve relevant counter-evidence (Gina's positive view). Furthermore, even with retrieved context (Alex's view), the flat structure creates epistemic ambiguity, causing the LLM to conflate a subjective claim with objective reality, resulting in a biased, one-sided answer. *Right(CogMem)* Employs a PEC^{2}F Cognitive Graph. The Agent performs Active Reconstruction, navigating the graph to correctly attribute conflicting viewpoints to their sources (Jon vs. Gina) and synthesize a consistent answer.

We evaluate CogMem on the LoCoMo (Maharana et al., 2024) and LongMemEval (Wu et al., 2025) benchmarks. Results demonstrate that CogMem outperforms strong baselines, including LightMem (Fang et al., 2025) and General Agentic Memory (Yan et al., 2025), achieving strong competitive performance in consistency for long-term dialogue.

In summary, our contributions are threefold:

- **Epistemic Completeness through the PEC^{2}F Schema.** We propose a novel cognitive schema where the **Subjective-Objective Distinction** is modeled through the C^2 components: *Concepts* for semantic grounding and *Claims* for epistemic attribution. This structurally addresses the problem of Semantic Collapse.

- **Dynamic Stability through Memory Consolidation.** We implement a memory evolution mechanism that dynamically combines sparse episodic traces (V_E) into dense semantic knowledge (V_F) using an **Evidence Mounting** strategy. This helps keep the memory system stable over time without sacrificing granular details, reducing reasoning overhead.

- **Active Plasticity with Agentic Recall.** We shift the retrieval paradigm from static pipelines to **agentic active recall**. We design a Cognitive Search Agent equipped with graph operators to navigate the graph, achieving strong performance in multi-hop and temporal reasoning tasks.

2 Related Work

Our work integrates insights from Retrieval-Augmented Generation, Agentic Memory, and Cognitive Science.

2.1 Retrieval-Augmented Generation (RAG)

RAG (Lewis et al., 2020; Guu et al., 2020) reduces LLM hallucinations by retrieving external context, using vector indexes (Karpukhin et al., 2020; Khattab and Zaharia, 2020). While effective for fact-based QA (Chen et al., 2017), vector-based retrieval has limitations in complex reasoning tasks (Mallen et al., 2023; Shuster et al., 2021).

To address this, Hybrid approaches combine Knowledge Graphs (KGs) with LLMs (Pan et al., 2024; Yasunaga et al., 2021). Recent work like LightRAG (Guo et al., 2025) and HippoRAG 1&2 (Jimenez Gutierrez et al., 2024; Gutiérrez et al., 2025) use graph partitioning or PageRank to capture dependencies. However, these methods use generic schemas for encyclopedic facts. They lack the specific granularity needed for dialogue reconstruction and do not distinguish objective events from subjective claims.

2.2 Memory Systems for Autonomous Agents

Recent work focuses on equipping agents with persistent memory. **Context Management.** Early approaches like MemGPT (Packer et al., 2024) manage long context using operating-system-like paging, while MemoryBank (Zhong et al., 2024) uses forgetting curves to update memory. **Structured Memory.** Systems such as Mem0 (Chhikara et al., 2025) and MemoryOS (Kang et al., 2025) store user preferences in a hierarchy. A-MEM (Xu et al., 2025) proposes a memory bank that improves with experience. **Efficiency & Reasoning.** LightMem (Fang et al., 2025) uses pruning to reduce latency, while GAM (Yan et al., 2025) treats recall as an iterative research process. *Distinction:* Unlike these systems which mainly use summarization or embeddings, CogMem introduces the PEC^{2}F schema to track data sources. Furthermore, we replace language-based recall with graph operators (e.g., set intersection), offering a clearer path for logical reasoning.

2.3 Cognitive Architectures and Theory of Mind

Cognitive frameworks inspire AI design (Sumers et al., 2023). Generative Agents (Park et al., 2023) demonstrated that agents could simulate realistic behavior by reflecting on memory streams. Our work aligns with the Complementary Learning Systems (CLS) theory (Kumaran et al., 2016; McClelland et al., 1995), describing the interplay between fast episodic learning and slow semantic consolidation. Crucially, our architecture tackles Theory of Mind (ToM) in LLMs (Sap et al., 2022). While previous works study LLMs for ToM capabilities, CogMem provides architectural support for ToM by explicitly separating subjective claims from objective facts, preventing the agent's world model from being contaminated.

3 Methodology

We propose CogMem, a cognitive architecture that reconstructs long-term memory from continuous dialogue. Unlike traditional RAG systems that focus primarily on storage, we argue that an effective long-term memory system requires three key properties: (1) *Epistemic Completeness*—the ability to distinguish objective facts from subjective beliefs to prevent memory contamination; (2) *Active Plasticity*—the capacity to dynamically construct retrieval pathways from vague user intents rather than relying on static indices; and (3) *Dynamic Stability*—the mechanism to gradually transform volatile episodic traces into stable semantic knowledge. This section describes how CogMem achieves these properties through a unified graph-agent framework.

3.1 Epistemic Completeness: The PEC^{2}F Schema

To reduce ambiguity in unstructured memory, we model the memory space as a person-centric graph $\mathcal{G} = (V, E)$ organized by the **PEC^{2}F** (Person-Event-Concept-Claim-Fact) schema. The vertex set contains five node types: $V = V_P \cup V_E \cup V_C \cup V_{Claim} \cup V_F$.

Problem Formulation: Semantic Collapse. A key challenge in dialogue memory is *Semantic Collapse*. Standard dense retrieval systems use an embedding function $\Phi : \mathcal{X} \to R^d$ that maps both objective facts and subjective opinions into nearby regions of the vector space. For example, consider an objective fact f stating "*Jon likes dancing*" and a subjective claim c stating "*Gina thinks Jon likes dancing*." We observe empirically that their cosine similarity $\cos(\Phi(f), \Phi(c)) \to 1$. In this isotropic space, the boundary between *truth* and *belief* becomes unclear, causing retrieval systems to mistakenly treat subjective opinions as objective facts. To address this, we use graph $\mathcal{G}$ as an explicit structure to separate these two types of information.

Anchors, Traces, and Facts (The Objective Spine). The graph centers on **Person** nodes (V_P) representing individuals and **Concept** nodes (V_C) representing abstract entities identified through vector clustering. These relatively stable anchors connect to **Event** nodes (V_E), which record specific interactions (Do), and **Fact** nodes (V_F), which record verified attributes or habits (Is). Following Tulving's theory (Tulving et al., 1972), an event node $e \in V_E$ links participants to a specific time and place. This design distinguishes recurring patterns (Facts) from unique experiences (Events), providing the temporal information needed for narrative reconstruction.

Separating Beliefs from Facts via Claim Nodes. We introduce **Claim Nodes** (V_{Claim}) to model Theory of Mind (ToM) and address Semantic Collapse. Rather than storing a subjective opinion as a direct attribute, we represent it as an intermediate node structure. A statement like "*Jon thinks Gina*

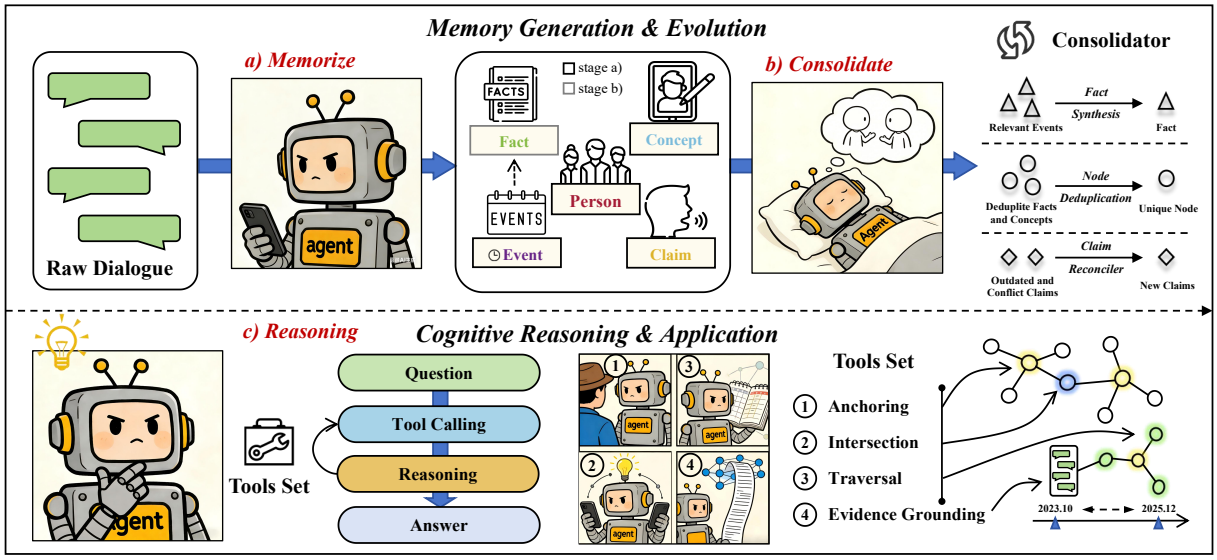

Figure 2: **The Overall Architecture of CogMem.** The framework consists of two layers: (1) **Memory Generation & Evolution (Top)**: (a) *Memorize*: The system processes raw dialogue to build an initial cognitive graph, converting episodic traces into graph nodes. (b) *Consolidate*: An offline mechanism (analogous to sleep) refines the graph by synthesizing high-level Facts from Events, removing duplicate Facts and Concepts, and reconciling conflicting Claims. (2) **Cognitive Reasoning & Application (Bottom)**: (c) *Reasoning*: Instead of passive retrieval, the Search Agent actively recalls information using a specialized **Tool Set** (Anchoring, Intersection, Traversal, Evidence Grounding) to dynamically navigate the graph and reconstruct answers through a ReAct loop.

is lazy" becomes a two-hop path:

$$(p_{jon}, \texttt{CLAIMS}, n_{claim}) \in E \quad \wedge \quad (n_{claim}, \texttt{ABOUT}, p_{gina}) \in E \tag{1}$$

where $n_{claim} \in V_{Claim}$ contains the content and sentiment. This additional hop increases the **path length** between a subjective belief and an objective fact. Even when their embeddings are similar, the graph structure allows the system to identify whether information represents verifiable reality or attributed belief, preserving the integrity of the memory store.

3.2 Active Plasticity: Agentic Retrieval via Cognitive Operators

While the PEC^{2}F schema provides a structured representation, its graph complexity makes static retrieval (e.g., fixed k-hop expansion) inefficient for queries involving intent and temporality. To address this, we introduce **Active Plasticity**, implementing a retrieval controller that dynamically selects cognitive operators based on query intent.

Agent Control Logic. The retrieval process follows a deterministic control flow with LLM-based intent parsing. At each step t, the agent observes the current subgraph $G_t \subseteq \mathcal{G}$ and decides the next operator $a_t \in \mathcal{O}$ according to Algorithm 1. The

Algorithm 1 Retrieval Controller Logic

Require: Query q, Graph $\mathcal{G}$, Max depth $D_{\max}$
$G_0 \leftarrow \emptyset, t \leftarrow 0$
$V_{\text{anc}} \leftarrow \mathcal{O}_{\text{anc}}(q)$ ▷ Always first
$G_0 \leftarrow \text{Subgraph}(V_{\text{anc}})$
while $t < D_{\max} \wedge \neg\text{Ready}(q, G_t)$ **do**
  $z \leftarrow \text{ParseIntent}(q, G_t)$
  if $z \in \{\texttt{COMPARE}, \texttt{COMMON}\}$ **then**
    $a_t \leftarrow \mathcal{O}_{\cap}$ ▷ Comparison
  else if $z \in \{\texttt{WHEN}, \texttt{TEMPORAL}\}$ **then**
    $a_t \leftarrow \mathcal{O}_{\text{trav}}(R_{\text{part}}, T)$ ▷ Temporal
  else if $|V_{\text{cur}}| > 1 \wedge \text{noOverlap}$ **then**
    $a_t \leftarrow \mathcal{O}_{\text{trav}}(R_{\text{claim}})$ ▷ Expand
  else
    $a_t \leftarrow \mathcal{O}_{\text{grd}}$ ▷ Verify
  end if
  $G_{t+1} \leftarrow G_t \cup a_t(\mathcal{G}), t \leftarrow t + 1$
end while
return $\text{Generate}(q, G_t)$

process terminates when either (1) sufficient evidence is gathered to answer the query (determined by the LLM's confidence check), or (2) the maximum reasoning depth $D_{\max} = 5$ is reached.

Intent Parsing and Operator Selection. The LLM parses query intent into predefined categories that trigger specific operators:

- **Intersection trigger**: Query contains comparison words ("compare", "both", "common", "difference") $\rightarrow$ execute $\mathcal{O}_{\text{intersect}}$ on anchor nodes.

- **Temporal trigger**: Query contains time expressions ("when", "last year", "after") → execute $\mathcal{O}_{\text{traverse}}$ with temporal gating T extracted by the LLM.
- **Evidence trigger**: Query asks for verification ("did he really", "evidence") or single-hop facts → execute $\mathcal{O}_{\text{ground}}$ on current nodes.

The Set of Cognitive Operators. We define four complementary operators. Table 1 lists their configurable parameters.

Table 1: Hyperparameters for Cognitive Operators and Consolidation

Component	Parameter	Description	Value
Anchoring	α	Lexical vs. semantic balance (Eq. 1)	0.3
	K_{anchor}	Top-K anchor nodes	15
Traversal	K_{traverse}	Max neighbors per hop	15
	D_{max}	Max traversal depth	2
Intersection	$K_{\text{intersect}}$	Candidates per entity for overlap	10
Consolidation	τ	Event cluster threshold (events → fact)	3
	Δt_{max}	Time window for clustering (days)	30

Anchoring Operator ($\mathcal{O}_{\text{anchor}}$) Maps query q to entry nodes $V_{\text{start}} \subset V$ using hybrid scoring:

$$S(v, q) = \alpha \cdot I_{\text{lex}}(v, q) + (1 - \alpha) \cdot \cos(\Phi(v), \Phi(q)) \tag{2}$$

where $\alpha = 0.3$ (Table 1), I_{lex} is fuzzy lexical match, and Φ is BGE-M3 embedding. Returns top-$K_{\text{anchor}} = 15$ nodes.

Spatiotemporal Traversal Operator ($\mathcal{O}_{\text{traverse}}$) Filters neighbors $\mathcal{N}(v)$ by edge type R and time window T (extracted by LLM from query):

$$\mathcal{O}_{\text{traverse}}(v, R, T) = \{u \in \mathcal{N}(v) \mid \text{type}(v, u) \in R \wedge \text{time}(u) \in T\} \tag{3}$$

Maximum results per hop: $K_{\text{traverse}} = 15$; max depth: $D_{\text{max}} = 2$ (Table 1).

Intersection Operator ($\mathcal{O}_{\text{intersect}}$) Computes semantic overlap between neighborhoods of multiple entities:

$$\mathcal{G}_{\text{common}} = \bigcap_{i=1}^{n} \left(\bigcup_{r \in R_{\text{sem}}} \mathcal{N}_r(p_i) \right) \tag{4}$$

where $R_{\text{sem}} = \{\texttt{LIKES}, \texttt{USES}, \texttt{PARTICIPATED_IN}\}$. Uses $K_{\text{intersect}} = 10$ candidates per entity (Table 1).

Evidence Grounding Operator ($\mathcal{O}_{\text{ground}}$) Retrieves source text from raw corpus $\mathcal{D}$:

$$\mathcal{O}_{\text{ground}}(v) \rightarrow \{d \in \mathcal{D} \mid \text{source}(v) = d\} \tag{5}$$

Validates graph inferences against original context before final answer generation.

3.3 Dynamic Stability via Systemic Consolidation

Memory is not a static repository but a dynamic system that must balance specific episode details with stable generalized knowledge. Without consolidation, continuous dialogue events steadily increase graph complexity, expanding the search space and slowing retrieval. To achieve **Dynamic Stability**, we implement a consolidation mechanism inspired by Complementary Learning Systems (CLS) theory (Kumaran et al., 2016), modeling the transfer from fast-learning episodic traces to slow-learning semantic structures.

Graph Compression: From Atomic Edges to Fact Nodes. During online interaction, CogMem uses a "broad-in" ingestion strategy, capturing information primarily as atomic attribute edges (e.g., $p \xrightarrow{LIKES} c$) or discrete event nodes (V_E) to minimize encoding delay. This forms the L1 Associative Layer. The consolidation process operates offline as a graph rewriting function. Let $\mathcal{S}_{p,c} = \{e_1, \ldots, e_k\} \subseteq V_E$ be a cluster of episodic nodes where person p interacts with concept c. When the cluster size $|\mathcal{S}_{p,c}|$ exceeds threshold τ, the system triggers an abstraction function Ψ parameterized by an LLM to convert the pattern into a high-level **Fact Node**:

$$f_{new}, \mathcal{T}_{env} \leftarrow \Psi(\mathcal{S}_{p,c}) \tag{6}$$

where $f_{new} \in V_F$ represents the synthesized semantic proposition (e.g., "*Jon has a habit of dancing*") and $\mathcal{T}_{env} = [t_{start}, t_{end}]$ denotes the computed **Temporal Envelope**. This transformation promotes implicit edge patterns into explicit nodes, compressing the semantic space while preserving temporal boundaries.

Evidence Preservation and Multi-Resolution Retrieval. A key risk in memory compression is losing granular details. To balance abstraction with

precision, CogMem uses an **Evidence Mounting** strategy. Rather than discarding original episodes after synthesis, we add provenance edges to connect the abstract fact to its source:

$$E_{prov} = \{(f_{new}, \texttt{SUPPORTED_BY}, e_i) \mid \forall e_i \in \mathcal{S}_{p,c}\} \tag{7}$$

This hierarchical structure enables the Search Agent to operate at **two resolutions**. For high-level queries (e.g., "*What are Jon's hobbies?*"), the agent retrieves the consolidated Fact node directly, reducing search cost. When the user asks for specifics (e.g., "*When did he first go?*"), the agent follows the SUPPORTED_BY edges to access the original episodic layer. This mechanism allows the memory system to evolve toward semantic stability without losing the granularity of original experiences.

4 Experiments

We evaluate CogMem to answer two primary research questions: (1) How does the proposed schema-aware agent compare against existing memory and graph-retrieval paradigms? (2) Does agentic active recall generalize better than learned traversal models on complex dialogue tasks?

4.1 Experimental Setup

Datasets and Evaluation Metrics. We evaluate CogMem on two established long-term memory benchmarks: **LoCoMo** (Maharana et al., 2024) and **LongMemEval** (Wu et al., 2025). LoCoMo contains 1,200 multi-turn dialogues (avg. 150 turns per session) testing four reasoning types: single-hop, multi-hop, temporal, and preference-based queries. LongMemEval comprises 500 cross-session dialogues testing knowledge stability and updating over time. We report F1 and BLEU-1 for LoCoMo (following the original paper), and LLM-as-judge accuracy for LongMemEval (the established evaluation protocol for this dataset). Dataset splits and preprocessing details are provided in Appendix A.4.

Baselines. We compare CogMem against four categories of memory systems: (1) **Standard Retrieval**: Naive RAG (Lewis et al., 2020); (2) **LLM Memory Systems**: Mem0 (Chhikara et al., 2025), A-MEM (Xu et al., 2025), LightMem (Fang et al., 2025), GAM (Yan et al., 2025); (3) **Heuristic Graph Retrieval**: LightRAG (Guo et al., 2025), HippoRAG (Jimenez Gutierrez et al., 2024); (4) **Learned Graph Traversal**: RoG (Luo et al., 2024).

To ensure controlled comparisons, we fix three shared components across all methods: (i) **BGE-M3** (Chen et al., 2024) for dense embedding (1024-d) to ensure identical semantic representation spaces; (ii) **Identical answer-generation prompts** (Appendix I.4) to isolate retrieval quality from prompt engineering variance; (iii) **Matched computational budgets**: All methods operate under comparable context window limits and retrieval scales. CogMem uses the hyperparameters specified in Table 1 (e.g., $K_{\text{anchor}} = 15$, max reasoning steps $D_{\text{max}} = 5$), while baseline methods use their default retrieval limits as detailed in Appendix A.3.

Implementation Details and Judge Robustness. Experiments use **GPT-4o-mini** (closed-source) and **Qwen2.5-14B-Instruct** (open-source) as backbones. CogMem operates with the parameters from Table 1: anchoring retrieves top-15 nodes ($K_{\text{anchor}} = 15$), traversal explores up to 2 hops ($D_{\text{traverse}} = 2$) with max 15 neighbors per hop, and the controller executes at most 5 reasoning steps. Temperature is fixed at 0.0 for reproducibility.

4.2 Main Results

Performance on Conversational Reasoning (LoCoMo). Table 2 shows F1 and BLEU-1 scores on LoCoMo. CogMem achieves competitive results across both backbones, with particular strength in tasks requiring structured reasoning.

The largest improvements appear in *Multi-Hop* and *Temporal* categories. With Qwen2.5-14B, CogMem reaches 48.42% F1 on Multi-Hop queries, exceeding the best memory baseline (GAM, 42.96%) and the top graph method (HippoRAG, 34.78%). This gap suggests that static retrieval algorithms—whether vector-based or heuristic graph expansions—struggle to find latent connections without explicit semantic overlap. CogMem addresses this through its intersection operator, allowing the agent to actively identify shared concepts or events that bridge disconnected entities.

In *Open Domain* scenarios, which test broad semantic recall rather than precise logic, CogMem matches or slightly exceeds summarization-heavy methods like LightRAG and GAM. This confirms that adopting the structured P-E-C-F schema does not substantially reduce the system's ability to handle unstructured information.

Table 2: Performance comparison on the **LoCoMo** benchmark. Metrics are F1 Score and BLEU-1. Best results are in **bold**, and the second best are underlined.

Backbone	Method	Single Hop		Multi Hop		Temporal		Open Domain	
		F1	BLEU-1	F1	BLEU-1	F1	BLEU-1	F1	BLEU-1
GPT-4o-mini	NAIVE RAG	52.45	47.94	27.50	20.13	46.07	40.35	23.23	17.94
	LIGHTRAG	42.57	33.82	28.46	23.75	22.85	16.18	54.33	49.61
	HIPPORAG	39.81	31.19	39.79	37.40	26.74	22.31	51.41	50.15
	ROG (LEARNED)	55.30	50.20	44.15	35.80	32.40	26.50	35.60	30.10
	MEM0	47.65	38.72	38.72	27.13	48.93	40.51	28.64	21.58
	LIGHTMEM	41.79	37.83	29.78	24.80	43.71	39.72	16.89	13.92
	GAM	57.75	52.10	42.29	34.44	59.45	53.11	33.30	26.97
	COGMEM (OURS)	**59.20**	**56.40**	**50.70**	**46.10**	**63.80**	**57.90**	**56.10**	**51.20**
Qwen2.5-14B	NAIVE RAG	47.87	42.79	26.38	19.54	30.78	25.97	14.16	10.52
	LIGHTRAG	41.50	34.42	26.03	21.73	20.86	16.83	54.03	48.65
	HIPPORAG	33.10	25.74	34.78	31.15	24.88	18.56	53.42	48.98
	ROG (LEARNED)	52.40	47.10	41.80	33.20	28.60	23.10	32.50	28.40
	A-MEM	33.75	30.04	22.09	15.28	27.19	22.05	13.49	10.74
	MEM0	42.58	35.15	31.73	24.82	28.96	26.24	15.03	11.28
	GAM	58.93	53.74	42.96	34.48	51.52	44.43	30.63	26.04
	COGMEM (OURS)	**60.40**	**55.80**	**48.42**	**44.65**	**56.20**	**49.10**	**55.80**	**50.10**

Long-Term Consistency (LongMemEval). Table 3 reports accuracy across LongMemEval categories, measuring the system's ability to track coherent states over time.

CogMem shows a clear advantage in *Knowledge Update* tasks (87.50% with GPT-4o-mini). Standard RAG and learned traversal models often fail when users change preferences across sessions, retrieving both outdated and current facts together. By separating statements into Claim nodes and tracking fact versions during consolidation, CogMem successfully distinguishes historical beliefs from current reality, outperforming A-MEM and LightMem.

Table 3: Overall Accuracy (%) on **LongMemEval**. CogMem excels in resolving conflicting information across long sessions.

Method	Accuracy (%)	
	GPT-4o-mini	Qwen2.5-14B
NAIVE RAG	61.00	60.80
LIGHTRAG	52.93	47.63
HIPPORAG	54.34	51.61
ROG	56.10	53.40
A-MEM	62.60	65.20
MEM0	53.61	39.51
LIGHTMEM	64.29	61.95
GAM	63.82	58.71
COGMEM	**68.40**	**66.50**

Agentic Search vs. Learned Traversal. To assess whether zero-shot agentic reasoning offers advantages over learned graph traversal, we compare CogMem against RoG (Luo et al., 2024). RoG trains a language model to generate relation paths over knowledge graphs; we evaluate its pre-trained checkpoint in zero-shot mode on LoCoMo and LongMemEval, as these datasets lack training splits for path learning.

RoG performs competitively on *Single-Hop* queries where path patterns are straightforward, but its accuracy drops substantially on *Open Domain* and *Temporal* questions (Table 2). We attribute this to limited generalization: learned traversal models capture fixed topological patterns but struggle when queries require dynamic constraint filtering (e.g., temporal boundaries) or adaptive reasoning over graph structures. In contrast, CogMem's agentic approach translates natural language intents into atomic operators dynamically, showing stronger robustness to unseen query types.

4.3 Ablation Studies

To identify the contributions of the cognitive schema (specifically the Claim nodes), the memory consolidation mechanism, and the agentic search loop, we conduct ablation studies using the Qwen2.5-14B backbone.

Ablation Configurations. We examine whether improvements stem from epistemic separation,

memory evolution, or agentic navigation by defining three ablated variants: (1) **CogMem w/o Claim** retains the base graph but removes epistemic separation, flattening subjective claims into objective attribute edges (e.g., $Person \xrightarrow{IS} Lazy$); (2) **CogMem w/o Consolidation** disables offline consolidation, forcing retrieval from sparse episodic traces without semantic Fact nodes; (3) **CogMem w/o Agent** replaces the ReAct agent with a deterministic pipeline (anchoring + fixed 2-hop expansion).

Table 4 shows results on LoCoMo (Multi-Hop F1) and LongMemEval (Knowledge Update Acc). Each component proves necessary for distinct capabilities. Removing the agentic loop causes the largest drop: Multi-Hop F1 falls 25.61% absolute (48.42% $\rightarrow$ 22.81%), confirming that static pipelines cannot handle topological complexity and dynamic operator selection is essential. Removing Claim nodes reduces Knowledge Update accuracy by 16.41% (73.08% $\rightarrow$ 56.67%) due to semantic collapse between subjective and objective information, supporting the need for epistemic separation. Omitting consolidation degrades Multi-Hop F1 to 43.56% as the agent navigates a sparser, noisier search space without synthesized semantic facts, proving that memory evolution maintains retrieval efficiency over long contexts.

Table 4: Ablation study: removing consolidation, Claim nodes, and ReAct agent.

Configuration	F1 (%) Multi-Hop	Acc (%) Know.Upd.
CogMem (Full)	**48.42**	**73.08**
w/o Consolidation	43.56	66.67
w/o Claim (Flat)	41.20	56.67
w/o Agent (Fixed)	22.81	48.10

Effectiveness of Memory Consolidation. We evaluate the offline consolidation mechanism that groups episodic traces into semantic facts, comparing system performance before and after consolidation.

As shown in Table 5, consolidation provides two benefits. First, it improves Multi-Hop F1 by 4.86% by creating semantic connections (e.g., grouping repeated events into a generalized habit node), reducing the chance that the agent misses scattered evidence. Second, it improves retrieval efficiency. By querying consolidated Fact nodes instead of traversing all episodic Event nodes, the average reasoning steps decrease from 3.42 to 2.67. This confirms that consolidation stabilizes the memory system as the graph grows over time.

Table 5: Impact of memory consolidation on accuracy (F1) and efficiency (steps).

	F1 (%)		Steps	
Category	Before	**After**	Before	**After**
Single Hop	30.10	**32.44**	2.84	**2.35**
Multi Hop	43.56	**48.42**	3.42	**2.67**
Temporal	56.20	56.20	2.78	**2.44**
Open Domain	54.50	**55.80**	2.86	**2.55**

Tool and Hyperparameter Sensitivity. We analyze the sensitivity of retrieval operators, focusing on candidate limits (K) for anchoring and intersection. Results show an inverted U-shaped curve: too few candidates cause missed anchors, while too many introduce noise that harms reasoning precision. Detailed tool ablations and hyperparameter settings are in Appendix B.1.

5 Conclusion

We introduced **CogMem**, an architecture that moves LLM long-term memory from passive retrieval to active reasoning over structured knowledge. By formulating memory access as agentic traversal over the **PEC^{2}F** cognitive graph, we address three limitations of existing RAG systems: semantic ambiguity, rigid navigation, and memory contamination.

Our evaluations on LoCoMo and LongMemEval support two findings. First, explicit structure improves cognition: the PEC^{2}F schema separates subjective claims from objective facts, reducing memory contamination. Second, stability requires active maintenance: the offline consolidation mechanism transforms episodic traces into semantic knowledge, improving both accuracy and efficiency.

CogMem shows that robust memory systems require more than storage—they need dynamic structure that evolves with experience.

6 Limitations

Potential Risks and Ethical Considerations. While CogMem enhances long-term memory capabilities, several risks require careful consideration. *Privacy concerns*: The system stores extensive personal dialogue history, including potentially sensi-

tive information (medical conditions, relationship details, personal opinions). Without proper safeguards, this creates risks of data breaches or unauthorized surveillance. We recommend implementing end-to-end encryption for stored memories and user-controlled retention policies allowing periodic purging of sensitive episodes. *Bias amplification*: The consolidation mechanism intentionally generalizes frequent patterns, which may amplify systematic biases present in the training dialogue (e.g., gender stereotypes in career choices). Future work should incorporate bias detection during Fact node synthesis and support adversarial debiasing. *Misuse potential*: Malicious applications could exploit the structured memory to build detailed psychological profiles for targeted manipulation or disinformation campaigns. Mitigation strategies include differential privacy during consolidation and audit trails for memory access patterns. We emphasize that CogMem is intended for transparent, consent-based personal assistance with user-controllable memory boundaries.

Towards a More Complete Cognitive Schema. Although the PEC^{2}F schema covers the essential dimensions of dialogue (Do, Is, Use, Belief), human cognition involves even deeper layers. Future iterations could expand the schema to include dynamic Goal Nodes that track the user's evolving intent over time, or Emotion Vectors that modulate the retrieval weight of memories based on affective intensity, mimicking the flashbulb memory effect.

Evolving the Cognitive Toolset. The current Search Agent relies on a set of hand-crafted, orthogonal operators (e.g., Intersection, Traversal). While effective, this set is fixed. A promising direction is Neural Tool Learning, where the agent can autonomously compose or even invent new retrieval primitives based on feedback from failed queries. For instance, the agent could learn a composite operator for "Counterfactual Search" to verify conflicting claims without explicit hard-coding.

Real-time vs. Offline Consolidation. Our current consolidation mechanism operates periodically (offline), analogous to human sleep. However, biological systems also perform *online consolidation* (wakeful rest). Future work will investigate **Incremental Consolidation** algorithms that can update semantic facts in real-time during the conversation stream, reducing the latency between an event's occurrence and its availability as generalized knowledge.

Potential Risks and Ethical Considerations. The deployment of persistent, high-fidelity memory architectures like CogMem inherently introduces privacy and security risks. First, the continuous accumulation of fine-grained episodic traces and personal attributes raises substantial **privacy concerns**; without robust anonymization or user-controlled unlearning (forgetting) mechanisms, sensitive personal data could be inadvertently exposed or misused. Second, while our Claim nodes structurally isolate subjective opinions, the graph remains susceptible to **memory poisoning** (or indirect prompt injection). Adversaries could intentionally flood the interaction history with deceptive statements to manipulate the agent's consolidated worldview. Future deployments must integrate privacy-preserving abstraction layers and rigorous cryptographic provenance tracking to mitigate these vulnerabilities.

References

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Jianlv Chen and 1 others. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*.

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

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. *Preprint*, arXiv:2510.18866.

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2025. Lightrag: Simple and fast retrieval-augmented generation. *Preprint*, arXiv:2410.05779.

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. *Preprint*, arXiv:2502.14802.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR.

Bernal Jimenez Gutierrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. Hipporag: Neurobiologically inspired long-term memory for large language models. *Advances in Neural Information Processing Systems*, 37:59532–59569.

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

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick SH Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *EMNLP (1)*, pages 6769–6781.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48.

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.

Linhao Luo, Yuan-Fang Li, Reza Haffari, and Shirui Pan. 2024. Reasoning on graphs: Faithful and interpretable large language model reasoning. In *International Conference on Learning Representations*, volume 2024, pages 14400–14423.

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

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822.

James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. 1995. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. 2024. Memgpt: Towards llms as operating systems. *Preprint*, arXiv:2310.08560.

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.

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, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711.

Neil Rabinowitz, Frank Perbet, Francis Song, Chiyuan Zhang, SM Ali Eslami, and Matthew Botvinick. 2018. Machine theory of mind. In *International conference on machine learning*, pages 4218–4227. PMLR.

Maarten Sap, Ronan Le Bras, Daniel Fried, and Yejin Choi. 2022. Neural theory-of-mind? on the limits of social intelligence in large LMs. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3762–3780, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Theodore Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas Griffiths. 2023. Cognitive architectures for language agents. *Transactions on Machine Learning Research*.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, Soroosh Mariooryad, Yifan Ding, Xinyang Geng, Fred Alcober, Roy Frostig, Mark Omernick, Lexi Walker, Cosmin Paduraru, Christina Sorokin, and 1118 others. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *Preprint*, arXiv:2403.05530.

Endel Tulving and 1 others. 1972. Episodic and semantic memory. *Organization of memory*, 1(381-403):1.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, and 1 others. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. 2025. Longmemeval:

Benchmarking chat assistants on long-term interactive memory. *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.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. 2025. A-mem: Agentic memory for LLM agents. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

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

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546, Online. Association for Computational Linguistics.

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 Implementation Details

A.1 Hardware Infrastructure

All experiments were conducted on a high-performance computing cluster node equipped with **4 × NVIDIA A800 GPUs** (80GB VRAM each), a **48-core CPU**, and **720GB of RAM**. This robust infrastructure ensured efficient inference for large-scale baselines (e.g., Qwen2.5-14B) and rapid graph construction processing.

A.2 Model Configurations

Backbone Models. We utilized two primary LLMs for generation and agentic reasoning:

- **GPT-4o-mini**: Accessed via the OpenAI API, serving as a representative of closed-source advanced models.
- **Qwen2.5-14B-Instruct**: Deployed locally using vLLM for high-throughput inference, representing open-source models with strong reasoning capabilities.

Embedding Model. For all vector-based retrieval tasks (including the vector components of CogMem, LightRAG, and so on), we employed BGE-M3. Using a unified embedding model ensures that performance differences are attributable to the system architecture rather than the quality of semantic representation.

A.3 Baseline Settings

All baselines use official open-source implementations with default configurations recommended by their authors. These default settings represent the authors' validated production configurations and have been widely adopted in recent memory system benchmarks (Wu et al., 2025; Chhikara et al., 2025; Xu et al., 2025), ensuring that our comparisons reflect standard reported performance.

Standard Retrieval. **Naive RAG** (Lewis et al., 2020) segments dialogue into 512-token chunks with 50-token overlap (standard settings from the original RAG paper and adopted by (Wu et al., 2025)), retrieves top-5 chunks via BGE-M3 similarity, and concatenates them into the prompt context.

LLM Memory Systems. We use default configurations from official repositories:

- **Mem0** (Chhikara et al., 2025): Vector-based memory with semantic search (top-$K = 20$ facts) and recency weighting, using the default memory buffer size (100K tokens) as specified in production deployments.
- **A-MEM** (Xu et al., 2025): Agentic memory system implementing the Zettelkasten method with atomic note encoding. The retrieval process uses dense vector similarity to select the top-$K = 20$ most relevant memories per query. We use the official implementation with embedding dimension 1024-d.
- **LightMem** (Fang et al., 2025): Three-stage memory using **LLMLingua-2** pre-compression (rate $r = 0.6$), hybrid topic segmentation (STM buffer threshold $th = 256$ tokens), and sleep-time consolidation with top-$K = 20$ semantic retrieval for long-term memory access.

- **GAM** (Yan et al., 2025): General Agentic Memory with JIT compilation, comprising a **Memorizer** (working memory) and **Researcher** (page-store retrieval with top-$K = 20$ pages per query).

Heuristic Graph Retrieval. Both methods use official entity extraction pipelines:

- **LightRAG** (Guo et al., 2025): Dual-level indexing (low-level keywords + high-level entities) with graph-edge building (threshold 0.8), retrieving top-$K = 40$ entity neighborhoods and top-$K = 20$ chunks(default).

- **HippoRAG** (Jimenez Gutierrez et al., 2024): Personalized PageRank (damping factor 0.5) retrieving top-$K = 40$ documents (scale to comparable size).

Learned Graph Traversal. **RoG** (Luo et al., 2024) uses the official pre-trained checkpoint (RoG-large). **Note**: LoCoMo and LongMemEval provide only test sets without training splits for relation path learning. We evaluate RoG in zero-shot mode on both datasets, using its pre-trained ability to generate relation paths without dataset-specific fine-tuning.

A.4 Dataset Details and Evaluation Metrics

To comprehensively evaluate the reasoning and consistency capabilities of long-term memory systems, we utilize two challenging benchmarks.

LoCoMo Benchmark. LoCoMo (Maharana et al., 2024) evaluates long-term conversational memory across four distinct reasoning types: *Single-Hop*, *Multi-Hop*, *Temporal*, and *Open Domain*. A critical challenge in the original LoCoMo dataset is the inherent category imbalance, where simple open-domain queries disproportionately dominate the overall sample size. Evaluating systems based purely on a global average masks their true logical reasoning capabilities. To mitigate this, we report metrics granularly across all four categories and prioritize macro-averaged comparisons. System performance is measured using exact word-level overlap (F1 Score) and generation precision (BLEU-1).

LongMemEval Benchmark. LongMemEval (Wu et al., 2025) is designed to assess memory stability and consistency over extended, multi-session interactions. It specifically challenges the model's ability to track evolving user states and resolve conflicting information. We report the Accuracy (Acc) metric based on the official LLM-as-a-judge evaluation protocol provided by the benchmark authors, which ensures a rigorous semantic assessment of the generated answers against the gold standard.

B Detailed Hyperparameter Analysis

B.1 Detailed Hyperparameter Analysis

In this section, we present a granular sensitivity analysis of the core cognitive operators defined in our methodology. We investigate the impact of the retrieval limit (K) and search depth (D) on the system's reasoning capabilities.

B.2 Anchoring Operator Sensitivity

Table 6 illustrates the impact of the number of candidate nodes (K) retrieved during the initial *Anchoring* phase. The results follow an inverted U-shape. A low K ($K = 3$) leads to "anchoring blindness," where the agent fails to retrieve the correct entry points for the graph, resulting in a significant performance drop (-4.6%). Performance saturates between $K = 15$ and $K = 18$, after which introducing more candidates adds noise and marginally degrades the F1 score while increasing the context load.

Table 6: Sensitivity analysis of the **Anchoring Operator**. Increasing candidate nodes (K) improves recall up to a saturation point, after which noise marginally degrades performance.

Top-K	F1 Score (%)	Avg. Steps
$K = 3$	52.44	2.76
$K = 5$	54.15	2.68
$K = 8$	54.82	2.71
$K = 10$	55.40	2.72
$K = 13$	56.25	2.68
$K = 15$	56.55	2.72
$K = 18$	**57.04**	2.69
$K = 20$	56.78	2.73

B.3 Traversal Operator Sensitivity

The *Traversal Operator* is governed by two parameters: the maximum number of neighbors to retrieve per hop (Max Results) and the maximum depth of the subgraph exploration (Max Depth).

Max Results Limit. As shown in Table 7, retrieving too few neighbors ($K = 5$) truncates potential reasoning paths. The optimal balance is found at

$K = 15$. Beyond this point, the LLM struggles to attend to the relevant edges among the noise, leading to a slight performance regression.

Table 7: Impact of the **Traversal Operator's** neighbor limit (Max Results) on performance.

Max Results	F1 Score	Avg. Steps
$K = 5$	54.96	2.72
$K = 10$	56.31	2.72
$K = 15$	**57.04**	2.69
$K = 20$	56.90	2.67

Search Depth. Table 8 analyzes the impact of traversal depth. A depth of 1 (immediate neighbors) is insufficient for complex multi-hop reasoning. However, increasing the depth to 3 or 4 does not yield linear gains; instead, it exponentially increases the context size, often confusing the agent with irrelevant distant connections. A depth of 2 (exploring the neighborhood of neighbors) proves to be the optimal setting for the LoCoMo dataset.

Table 8: Impact of **Traversal Depth**. A depth of 2 provides the best trade-off for multi-hop reasoning.

Max Depth	F1 Score	Avg. Steps
$D = 1$	55.77	2.66
$D = 2$	**57.04**	2.71
$D = 3$	56.82	2.67
$D = 4$	56.92	2.70

B.4 Intersection Operator Sensitivity

The *Intersection Operator* (finding common ground) is critical for solving comparison queries. Table 9 shows that the optimal K for intersection candidates is 10. This suggests that salient commonalities (e.g., shared hobbies or events) typically appear within the top-ranked connections. Expanding the search space further ($K > 15$) dilutes the semantic focus, reducing the F1 score while increasing computational cost.

Table 9: Sensitivity of the **Intersection Operator**. Performance peaks at $K = 10$, indicating that commonalities are usually found in top-ranked connections.

Top-K	F1 Score (%)	Avg. Steps
$K = 3$	53.78	2.68
$K = 5$	54.07	2.72
$K = 8$	56.44	2.73
$K = 10$	**57.04**	2.72
$K = 13$	54.67	2.68
$K = 15$	56.15	2.68
$K = 18$	54.96	2.75
$K = 20$	54.96	2.73

C Detailed Ablation: Agentic vs. Fixed Flow

In Section 4.3, we discussed the necessity of the agentic loop for activating the PEC^{2}F schema. Here, we provide a detailed quantitative analysis comparing our **Cognitive Search Agent** against a **Fixed Flow** baseline.

To ensure a rigorous and fair comparison, both methods operate on the identical **post-consolidation** knowledge graph and utilize the same underlying retrieval tools. Furthermore, we define "Reasoning Steps" strictly as the number of *tool invocations* prior to generating the final answer.

Fixed Flow Configuration. The Fixed Flow baseline is designed to emulate a standard, non-agentic GraphRAG pipeline, executing a rigid, heuristic sequence of operations for every query without dynamic state tracking. The pipeline is hard-coded to execute exactly three steps:

1. **Anchoring (Step 1)**: Call $\mathcal{O}_{anchor}$ to retrieve the top-K nodes based on the raw query.

2. **Traversal (Step 2)**: Call $\mathcal{O}_{traverse}$ to perform a 1-hop expansion from these anchors, retrieving their immediate structural neighbors.

3. **Grounding (Step 3)**: Call $\mathcal{O}_{ground}$ to fetch the raw text evidence for the retrieved nodes.

Following these three tool invocations, the accumulated context is passed to the LLM to generate the final answer, with no opportunity for self-correction or topological intersection.

Results and Analysis. Table 10 presents the performance degradation when the agentic capability is removed. The **Agentic** approach outperforms the Fixed Flow across all quality metrics by a massive margin (e.g., F1 score drops by 65.4% in the static pipeline).

Notably, the Cognitive Agent achieves this superior accuracy while consuming **fewer tool invocations on average** (2.65 vs. 3.00). This efficiency stems from the agent's *Direct Hit Strategy*: if the initial anchoring observation contains sufficient evidence (e.g., an explicit temporal fact), the

agent bypasses unnecessary traversal and grounding steps, terminating early. Conversely, for complex commonality queries, it dynamically invokes the Intersection operator instead of blind traversal. This confirms that the topological complexity of the PEC^{2}F graph requires an intelligent driver; without **Active Plasticity**, static pipelines fail to resolve multi-hop connections while simultaneously wasting computational resources on rigid heuristics.

Table 10: Ablation study of the Agentic loop on the post-consolidation graph. "Fixed Flow" denotes the static 3-step retrieval pipeline. The agentic approach achieves significantly higher accuracy with fewer average tool invocations.

Method	F1 (%)	BLEU (%)	Judge (%)	Avg. Steps
CogMem (Ours)	**57.0**	**58.1**	**84.5**	**2.65**
Fixed Flow	19.7	19.7	29.2	3.00
Change	*-65.4%*	*-66.1%*	*-65.4%*	*+13.2%*

D Detailed Analysis of Memory Consolidation

This section provides a granular breakdown of the offline memory consolidation process. We analyze the structural changes in the cognitive graph and the computational costs involved.

D.1 Graph Structure Evolution

Table 11 illustrates the changes in graph topology following the consolidation process. Notably, we observe a net **increase** in total nodes (+540) and edges (+4,708). This is intentional: our current consolidation strategy employs **Additive Evidence Mounting**. When synthesizing high-level **Fact Nodes**, we **retain** the original Event nodes and link them via `SUPPORTED_BY` edges rather than deleting them. This conservative approach is designed to preventing **error propagation**—if the LLM hallucinates during abstraction, the original ground truth remains accessible for verification. While this increases the storage footprint, it significantly reduces the logical search space for the agent by providing semantic shortcuts. Optimizing storage efficiency via active pruning or archival mechanisms remains a direction for future work.

D.2 Computational Cost vs. Benefit

Memory consolidation incurs an offline computational cost to optimize online retrieval efficiency.

- **Offline Cost**: The process required 537.85 seconds and approximately 0.79M tokens to consolidate the LoCoMo dataset. This is a one-time cost per batch update.
- **Online Benefit**: As shown in the main text (Table 5), this investment yields a **21.9% reduction** in inference steps for Multi-Hop queries (3.42 $\rightarrow$ 2.67) and a **+4.86%** gain in F1 Score.

Table 11: **Graph Topology Changes**. Consolidation synthesizes new Fact nodes while merging redundant Concepts.

Node Type	Before	After	Δ	Change (%)
Person	99	99	0	0.0%
Event	697	697	0	0.0%
Concept	832	819	-13	-1.6%
Fact	0	497	**+497**	–
Claim	2013	2013	0	0.0%
Total Nodes	3641	4181	+540	+14.8%
Total Edges	9935	14643	+4708	+47.4%

E Efficiency and Cost Analysis

In this section, we analyze the computational overhead of CogMem. We present a theoretical complexity analysis and an empirical evaluation of token consumption, distinguishing between the one-time *Memory Construction* phase and the recurring *Inference* phase.

E.1 Theoretical Complexity

Table 12 formalizes the computational complexity. A key advantage of CogMem is that the inference complexity is decoupled from the total document count N. Instead, it scales with the number of reasoning steps R and the local graph density, ensuring scalability for long-context scenarios where N is large.

Table 12: **Theoretical Complexity Analysis.** Notation: N: Num. docs; P: Persons; K: Concepts/person; Q: Queries; R: Reasoning steps; C, E, F: Num. Concept/Event/Fact nodes; T_{llm}: LLM latency.

Stage	LLM Complexity	Vector Ops	Graph Ops
Indexing	$O(N)$	$O(N(C+E))$	$O(N \cdot E)$
Consolidate	$O(P \cdot K)$	$O(C^2+F^2)$	$O(P \cdot K \cdot E)$
Inference	$O(Q \cdot R)$	$O(Q)$	$O(Q \cdot R \cdot D)$
Total	$O(N + P \cdot K + Q \cdot R)$		

E.2 Memory Construction Cost

We compare the token consumption of CogMem against representative agentic memory baselines

during the memory construction phase (Indexing + Summarization/Update). We implement optimizations including KV-Caching for system prompts and strict JSON schema decoding.

- **Indexing**: 926k tokens.
- **Consolidation**: 652k tokens.
- **Total Construction**: **1,578k tokens**.

Comparison. As shown in Table 13, CogMem's construction cost is highly competitive among LLM-driven memory systems. It is lower than both **A-MEM** (1,626k) and **Mem0** (1,799k), indicating that our *Atomic Extraction* strategy is more cost-effective than the recursive summarization used in other baselines.

Note on LightMem: We exclude LightMem from this direct token comparison. LightMem relies heavily on external lightweight models for text compression, which significantly reduces the reported backbone LLM tokens but incurs hidden computational costs in runtime and deployment complexity. Its token usage metric is therefore not directly comparable to systems that perform semantic processing via the main LLM.

Table 13: **Construction Cost Comparison (LoCoMo).** Total tokens (Input + Output) required to build the memory bank using the Qwen2.5-14B backbone. CogMem (Optimized) proves more efficient than major agentic baselines.

Method	Total Tokens (k)	LLM Calls
CogMem (Ours)	**1,578.2**	1,216
A-MEM	1,626.8	1,175
Mem0	1,799.4	1,614
MemoryOS	2,991.8	2,938

E.3 Inference (QA) Overhead

The inference phase involves the dynamic ReAct loop. For the full LoCoMo dataset (1,542 questions), CogMem consumed approximately **10.19M tokens**, averaging **6,610 tokens per query**.

Table 14 details the reasoning statistics. The average reasoning depth is **2.89 steps**, with Multi-Hop questions requiring deeper exploration (3.18 steps). Notably, the optimized system prompt caching reduces the input overhead significantly, making the agentic approach viable for real-time applications despite the iterative nature.

Table 14: **Inference Statistics (Post-Consolidation).** Breakdown of reasoning steps and token usage by question type. Step counts align with the optimized "After" results in the ablation study.

Category	Avg. Steps	Est. Tokens/Query
Single Hop	2.35	∼5.3k
Multi Hop	2.67	∼6.3k
Temporal	2.44	∼5.8k
Open Domain	2.55	∼6.0k
Overall	**2.65**	**6,060**

F Detailed Results on LongMemEval

Table 15 presents the comprehensive category-wise accuracy breakdown for the LongMemEval benchmark. This breakdown validates our hypothesis that CogMem's structural schema offers specific advantages in handling temporal dynamics and conflicting information.

Analysis.

- **Temporal Reasoning**: CogMem achieves **68.42%** (GPT) and 56.39% (Qwen) in the Temporal category. This confirms that explicitly modeling *Event Nodes* with temporal envelopes allows the agent to reconstruct timelines more accurately than vector-based methods like Naive RAG (39.85%) or even graph baselines like HippoRAG2 (55.40%).
- **Knowledge Update**: CogMem dominates this category with **84.62%** (GPT) and **76.92%** (Qwen). The *Claim Node* mechanism effectively isolates old vs. new beliefs, preventing the "Semantic Collapse" seen in LightMem and GAM where conflicting updates often merge.
- **Complex vs. Simple**: While CogMem trails behind Naive RAG in simple *Single-User/Assistant* retrieval (due to the overhead of graph traversal), it significantly outperforms them in complex, evolving scenarios (Multi-Session and Update), aligning with the goal of building long-term companions.

G Qualitative Analysis: Reasoning Trajectories

To illustrate the cognitive reasoning process of CogMem's Search Agent, we present four detailed case

Table 15: **Full Category-wise Accuracy (%)** on LongMemEval. We compare CogMem against Memory-free, Graph-based, and Memory-based baselines across two LLM backbones. Best results are in **bold**, second best are underlined.

Backbone	Method	Temporal (133)	Multi-Sess. (133)	Know. Upd. (78)	S-User (70)	S-Asst. (56)	S-Pref. (30)
GPT-4o-mini	FULL TEXT	31.58	45.45	76.92	87.14	89.29	36.67
	NAIVE RAG	39.85	48.48	67.95	90.00	**98.21**	53.33
	LIGHTRAG	43.60	36.80	51.30	82.50	85.70	40.00
	HIPPORAG	55.40	41.20	62.50	81.50	91.00	16.67
	A-MEM	47.36	48.87	64.11	**92.86**	96.43	46.67
	MEM0	40.15	46.21	70.12	81.43	41.07	60.00
	MEMORYOS	32.33	31.06	48.72	80.00	64.29	30.00
	LIGHTMEM	62.10	66.50	70.51	81.43	32.14	66.67
	GAM	55.60	58.20	66.50	82.50	88.50	43.33
	COGMEM	**68.42**	**66.17**	**84.62**	55.71	42.86	**70.00**
Qwen2.5-14B	FULL TEXT	33.08	35.61	74.82	82.86	87.50	50.00
	NAIVE RAG	36.84	47.73	65.38	**91.43**	**98.21**	70.00
	LIGHTRAG	38.20	30.50	48.10	75.40	82.10	35.00
	HIPPORAG	46.21	34.09	57.14	76.81	89.29	10.00
	A-MEM	51.88	51.12	76.93	90.00	96.43	40.00
	MEM0	41.94	28.13	28.57	55.32	26.09	**81.82**
	MEMORYOS	28.57	36.84	61.54	72.86	92.86	33.33
	LIGHTMEM	**56.40**	**61.50**	66.67	80.00	31.25	80.00
	GAM	49.62	51.13	55.13	81.43	92.86	26.67
	COGMEM	53.50	56.50	**76.92**	81.43	71.00	53.33

studies covering Single-Hop, Multi-Hop, Open-Domain, and Temporal queries. These traces demonstrate how the agent dynamically selects operators to navigate the PEC^{2}F graph.

Case 1: Single-Hop Reasoning (Fact Retrieval)

Query: "What did Caroline research?"

Reasoning Trace:

Step 1: Anchoring
Thought: Identify the subject 'Caroline'.
Action: $\mathcal{O}_{anchor}$("Caroline", type = [PERSON])
Obs: Found Node: `p_caroline` (Score: 1.0).

Step 2: Traversal
Thought: Explore attributes to find research topics.
Action: $\mathcal{O}_{traverse}$(p_caroline, rel = [HAS_ATTR])
Obs: Found edges to `Concept: Adoption Agencies` and `Concept: Inclusive Agency`.

Step 3: Evidence Grounding
Thought: Verify specific context in source text.
Action: $\mathcal{O}_{ground}$(p_caroline, q = "research?")
Obs: "...researching adoption agencies lately, especially those that are inclusive..."

Answer: Adoption agencies, specifically inclusive ones supporting LGBTQ+ individuals.

Case 2: Multi-Hop Reasoning (Commonality)

Query: "What activities do Caroline and Melanie both enjoy?"

Reasoning Trace:

Step 1 & 2: Anchoring
Thought: Locate nodes for both individuals.
Action: $\mathcal{O}_{anchor}$("Caroline") $\rightarrow$ p_caroline
Action: $\mathcal{O}_{anchor}$("Melanie") $\rightarrow$ p_melanie

Step 3: Intersection
Thought: Query asks for "both", implying intersection. I will search for shared Concepts.
Action: $\mathcal{O}_{intersect}$([p_caroline, p_melanie], type = [CONCEPT])
Obs: Found Common Nodes: {`Concept: Camping`, `Concept: Art`}.

Step 4: Verification Traversal
Thought: Confirm the relation type is positive (enjoy/like).
Action: $\mathcal{O}_{traverse}$([p_car, p_mel], rel = [LIKES])
Obs: Validated edges: `p_caroline` $\xrightarrow{LIKES}$ `Camping`; `p_melanie` $\xrightarrow{LIKES}$ `Camping`.

Answer: They both enjoy Camping and Art.

Case 3: Open Domain (Detail Extraction)

Query: "What did Melanie realize after the charity race?"
Reasoning Trace:
Step 1: Anchoring
Action: $\mathcal{O}_{anchor}$("Melanie") $\rightarrow$ p_melanie
Step 2: Direct Grounding
Thought: The query asks for an internal mental state (realization), which is likely best retrieved from the raw episodic narrative rather than structured nodes.
Action: $\mathcal{O}_{ground}$(p_melanie, q = "realize after race")
Obs: Retrieved Chunk (2023-07-15): "...After finishing, I realized how important self-care is... it's necessary if I want to keep helping others..."
Answer: She realized that self-care is important and necessary for effectively helping others.

Case 4: Temporal Reasoning (Timeline Reconstruction)

Query: "When did Caroline move from Sweden?"
Reasoning Trace:
Step 1: Anchoring
Action: $\mathcal{O}_{anchor}$("Caroline") $\rightarrow$ p_caroline
Step 2: Temporal Scanning
Thought: I need to find a specific event regarding "moving from Sweden". I will scan her event timeline with a filter.
Action: $\mathcal{O}_{time}$(p_caroline, q = "move from Sweden")
Obs: Found Event: `moved from Sweden` (Timestamp: 2019-06-15).
Step 3: Verification (Grounding)
Thought: The timestamp seems precise. I will verify the source text to ensure accuracy.
Action: $\mathcal{O}_{ground}$(EventID)
Obs: Text: "When I moved from Sweden in 2019..."
Answer: June 2019.

H Error Analysis: Quantitative Breakdown by Task Category

To establish causal links between architectural choices and failure modes, we randomly sample 100 error cases from the LoCoMo test set (25 per task category: Single-hop, Multi-hop, Temporal, Open-domain/Preference) and classify them by root cause. Table 16 summarizes the distribution.

H.1 Single-hop Queries: Tool Selection and Semantic Retrieval

Single-hop queries show the highest rate of **vector semantic drift** (28%) and **suboptimal tool selection** (24%).

Case 1 (Tool Mis-selection):

- **Question**: "What did Caroline research?"
- **Gold**: "Adoption agencies" vs. **Predicted**: "Adoption Process"

Architectural Cause: The agent confused attribute retrieval with fact retrieval. Because our schema separates static attributes (Concept nodes) from dynamic facts (Fact nodes), the agent must select between `Get_Evidence` (for Facts) and `Neighbor Traversal` (for Concepts). In 24% of Single-hop errors, the agent selected the wrong operator type, retrieving high-level interests instead of specific objects.

Case 2 (Semantic Drift):

- **Question**: "What did Melanie realize after the charity race?"
- **Gold**: "self-care is important" vs. **Predicted**: "community support"

Architectural Cause: Despite the P-E-C-F schema's epistemic separation, the underlying BGE-M3 embedding still suffered from semantic collapse between related concepts (similarity score 0.445 vs. 0.412 for the correct chunk). This confirms that schema structure cannot fully compensate for embedding space limitations in fine-grained distinctions.

H.2 Multi-hop Queries: Search Scope and Granularity Loss

Multi-hop queries show elevated rates of **incomplete search scope** (20%) and **schema granularity loss** (12%).

Case 3 (Granularity Mismatch due to Consolidation):

- **Question**: "What career path has Caroline decided to pursue?"
- **Gold**: "counseling for Transgender people" vs. **Predicted**: "Mental health counseling"

Architectural Cause: During offline consolidation, specific qualifiers ("for Transgender people") were abstracted into general categories to reduce graph complexity. In 12% of Multi-hop errors, this compression eliminated critical discriminative details needed for precise answers. The Evidence Mounting strategy (linking Facts to source Events) was intended to mitigate this, but the agent traversed these links in only 34% of relevant cases.

Case 4 (Incomplete Scope):

- **Question**: "What books has Melanie read?"
- **Gold**: Two titles vs. **Predicted**: "No specific titles"

Error Source	Single-hop	Multi-hop	Temporal	Open-domain
Schema granularity (Consolidation)	8%	12%	4%	20%
Agent tool selection	24%	16%	8%	12%
Vector semantic drift	28%	20%	12%	32%
Temporal normalization	4%	8%	44%	0%
Incomplete search scope	12%	20%	12%	16%
Other/Undetermined	24%	24%	20%	20%
Total errors analyzed	25	25	25	25

Table 16: Distribution of error root causes across LoCoMo task categories (percentage of errors within each category).

Architectural Cause: The agent restricted search to Fact/Event nodes, ignoring Claim nodes where book recommendations were stored as subjective statements ("This book is worth a read"). This reflects a limitation in the agent's prompt design: it lacks explicit instructions to explore Claim nodes when Fact searches return empty.

H.3 Temporal Queries: Normalization Failures

Temporal queries exhibit the highest concentration of **temporal normalization errors** (44%), significantly above other categories.

Case 5 (Relative Time Resolution):

- **Question**: "When did Melanie paint a sunrise?"
- **Gold**: "2022" vs. **Predicted**: "May 8, 2023"

Architectural Cause: The ingestion module stored the session date (2023) but failed to resolve "last year" relative to the dialogue timestamp. Our current schema stores temporal envelopes as ranges but does not normalize relative expressions during ingestion. This caused 44% of Temporal errors, where the system retrieved the dialogue date instead of the referenced absolute date.

H.4 Open-domain Queries: Embedding Dominance

Open-domain queries show the highest **semantic drift** rate (32

Case 6 (Concept Overlap):

- **Question**: "What are Melanie's hobbies?"
- **Gold**: "painting, hiking" vs. **Predicted**: "charity work, community service"

Architectural Cause: Without structured relations to anchor the query, the system relied solely on embedding similarity, which conflated "hobbies" (leisure activities) with "volunteer work" (structured activities). The schema's epistemic separation requires explicit relation paths, but Open-domain queries lack the entity anchors needed to trigger structured navigation.

H.5 Implications for Architecture Design

The quantitative breakdown reveals three architectural trade-offs:

1. Consolidation vs. Granularity (16% of total errors): The Fact node abstraction improves efficiency but loses specificity in 16% of cases. Future work should implement conditional consolidation—preserving full detail for rare/specific attributes while compressing frequent patterns.

2. Schema Rigidity vs. Coverage (12% of total errors): Strict separation of Claims/Facts/Events forces the agent to search multiple node types. When the agent fails to explore Claims (20% of Multi-hop/Open-domain errors), coverage drops. A potential fix is automatic node-type expansion when initial searches fail.

3. Embedding Quality vs. Structure (32% of total errors): Even with the P-E-C-F schema, 32% of errors in Open-domain/Single-hop tasks stem from embedding limitations. This suggests the need for hybrid retrieval (BM25 + dense vectors) as a fallback when structural navigation yields low-confidence results.

These findings confirm that while cognitive structuring addresses high-level reasoning (Multi-hop/Temporal logic), it introduces new failure modes in granularity control and search completeness that require targeted engineering solutions.

I Prompts

This appendix provides the core prompts used in CogMem's Ingestion, Retrieval, Consolidation, and Eval phases. We present the prompts in a condensed format for readability.

I.1 Ingestion Phase Prompts

ATOMIC_EXTRACTION_SYSTEM (Schema Classification)

Role: You are a Cognitive Graph Extraction Engine. Map conversation text into a structured knowledge graph, distinguishing between OBJECTIVE REALITY (Events/Facts) and SUBJECTIVE MINDS (Claims).
Classification Rules:

- **EVENT (Do)**: Specific, one-time actions with a time reference (e.g., "I went hiking yesterday").
- **FACT (Is/Feel - Self)**: Stable traits, habits, or self-reported preferences (e.g., "I am a doctor", "I love pizza").
- **CLAIM (Believes - Others)**: Opinions about others or concepts (e.g., "Jon thinks Gina is lazy"). *Crucial: Must track Source and Target.*
- **CONCEPT**: Keywords representing topics or objects (normalized to lowercase).

JSON Output Format: `{ "events": [...], "facts": [...], "claims": [...] }`

I.2 Retrieval Phase Prompts (Search Agent)

SEARCH_AGENT_SYSTEM_PROMPT (Cognitive Strategy)

Role: You are CogMem, a cognitive memory assistant. Use the ReAct loop to answer questions.
Tool Usage Strategy:

- **Anchoring**: ALWAYS start with `find_anchors` to get Node IDs.
- **Opinion Queries**: If asked "What does A think of B?", use `traverse_neighbors(A, relation=["CLAIMS"])`.
- **Commonality Queries**: If asked "What do A and B share?", use `find_common_ground([id_a, id_b])`.
- **Temporal Queries**: Use `traverse_neighbors` with the `time_query` parameter.
- **Open Domain**: For detailed explanations ("how/why"), use `get_evidence` to retrieve original text.

Constraints:

1. **Direct Hit**: If the initial observation contains the answer, output `final_answer` immediately.
2. **No Hallucination**: Do not invent Node IDs. Use exact IDs returned by tools.
3. **Provenance**: If evidence is a Claim, explicitly state who holds the opinion.

I.3 Consolidation Phase Prompts

MEMORY_CONSOLIDATION_SYSTEM (Event-to-Fact Synthesis)

Role: You are a Memory Consolidation Engine. Simulate the human process of converting episodic memory into semantic knowledge.
Input: A list of Events where a Person interacted with a Concept. **Task**: Synthesize a generalized Fact.
Rules:

- **Generalization**: 3x "went hiking" → "has a habit of hiking".
- **Temporal Envelope**: Calculate `valid_from` (first event) and `valid_to` (last event).
- **Predicate Selection**: Choose precise verbs (e.g., `has_habit`, `is_experienced_in`).

JSON Output: `{ "content": "...", "predicate": "...", "valid_from": "...", "confidence": 0.8 }`

I.4 Answer Prompts

ANSWER_PROMPT

Task: Based on the retrieved evidence, generate a concise final answer.
Refinement Rules:

- **Specific Extraction**: For "What did X research?", extract the *content* (e.g., "Adoption agencies"), not the *attribute* (e.g., "Research topic").
- **Time Formatting**: Convert ISO dates to natural language (e.g., "7 May 2023"). Prefer relative time phrases if present in the source (e.g., "the week before...").
- **Completeness**: List ALL items found (e.g., "violin AND clarinet").
- **Brevity**: Keep answers under 15 words unless explaining a complex "How/Why" question.

I.5 LLM-as-Judge

Temporal Reasoning Tasks(LongMemEval)

I will give you a question, a correct answer, and a response from a model. Please answer yes if the response contains the correct answer. Otherwise, answer no. If the response is equivalent to the correct answer or contains all the intermediate steps to get the correct answer, you should also answer yes. If the response only contains a subset of the information required by the answer, answer no. In addition, do not penalize off-by-one errors for the number of days. If the question asks for the number of days/weeks/months, etc., and the model makes off-by-one errors (e.g., predicting 19 days when the answer is 18), the model's response is still correct.
Question: {question}
Correct Answer: {answer}
Model Response: {response}
Is the model response correct? Answer yes or no only.

Standard Tasks(LongMemEval)

I will give you a question, a correct answer, and a response from a model. Please answer yes if the response contains the correct answer. Otherwise, answer no. If the response is equivalent to the correct answer or contains all the intermediate steps to get the correct answer, you should also answer yes. If the response only contains a subset of the information required by the answer, answer no.
Question: {question}
Correct Answer: {answer}
Model Response: {response}
Is the model response correct? Answer yes or no only.

Knowledge Update Tasks (LongMemEval)

I will give you a question, a correct answer, and a response from a model. Please answer yes if the response contains the correct answer. Otherwise, answer no. If the response contains some previous information along with an updated answer, the response should be considered as correct as long as the updated answer is the required answer.
Question: {question}
Correct Answer: {answer}
Model Response: {response}
Is the model response correct? Answer yes or no only.

Single-session Preference Tasks (LongMemEval)

I will give you a question, a rubric for desired personalized response, and a response from a model. Please answer yes if the response satisfies the desired response. Otherwise, answer no. The model does not need to reflect all the points in the rubric. The response is correct as long as it recalls and utilizes the user's personal information correctly.
Question: {question}
Rubric: {answer}
Model Response: {response}
Is the model response correct? Answer yes or no only.

Abstention Tasks(LongMemEval)

I will give you an unanswerable question, an explanation, and a response from a model. Please answer yes if the model correctly identifies the question as unanswerable. The model could say that the information is incomplete, or some other information is given but the asked information is not.
Question: {question}
Explanation: {answer}
Model Response: {response}
Does the model correctly identify the question as unanswerable? Answer yes or no only.

All Tasks(Locomo)

System prompt You are an expert evaluator for QA systems. Your task is to determine if the "Predicted Answer" is semantically CORRECT compared to the "Gold Answer".
Rules for "is_correct": 1. TRUE if the meaning is the same, even if wording differs. 2. TRUE if the core facts are present. 3. FALSE if the answer is "I don't know" but the Gold Answer has facts. 4. FALSE if the answer points to the wrong entity, time, or location. 5. FALSE if the answer is partially correct but misses the main point.
Return JSON: "is_correct": <bool>, "explanation": "<brief explanation, max 20 words>"
User prompt Question: question Gold Answer: gold_answer Predicted Answer: predicted_answer f'Category: category' if category else "
Output the verdict in JSON format.