CABLE: Extending the Reach of Memory Retrieval via Complementary Antecedent-Based Linking and Expansion

Zheling Tan[1], **Jin Gao**[1], **Dequan Wang**[1,2*]
[1]Shanghai Jiao Tong University [2]Shanghai Innovation Institute

Abstract

As LLM agents operate across structured workflows and sessions, preserving long-term history does not ensure that later contexts can recover relevant evidence through a bounded memory interface. We study this evidence-reachability problem in long-term conversational memory, where retrieval still relies heavily on semantic similarity. This works well for topical recall, but it often misses earlier experiences, plans, or motivations that are semantically distant from the later events they help explain. Existing memory graphs provide cross-memory structure, yet links driven mainly by semantic overlap can duplicate what the host retriever already recovers. We argue that link construction should instead prioritize a sparse set of retriever-complementary associations. We present CABLE (**C**omplementary **A**ntecedent-**B**ased **L**inking and **E**xpansion), a plug-in augmentation that constructs links designed to extend the host retriever's direct semantic reach. For each new memory, CABLE generates antecedent-oriented queries, retrieves prior memories, subtracts candidates in the direct semantic neighborhood, and verifies the remainder before adding the accepted complementary associations into a sparse directed graph. At retrieval time, CABLE expands the host system's retrieved seeds along these links to surface implicit supporting evidence. We evaluate CABLE with A-MEM on LoCoMo and MA-LongMemEval, and further integrate it into SimpleMem and Mem0^g on LoCoMo, using Qwen3.5-27B, DeepSeek-chat, and GPT-4o-mini. CABLE yields higher mean LLM-judge scores in every evaluated system-level setting, with the largest gains in categories where useful evidence is distributed across memories or sessions, including open-domain, multi-session, and preference-oriented questions. These results support prioritizing sparse, reasoning-relevant associations that complement rather than duplicate the host retriever. The core CABLE implementation is available at https://github.com/TanZheling/CABLE.

1 Introduction

LLM agent systems are increasingly organized as execution graphs of specialized components and explicit state transitions (Wu et al., 2024; Zhuge et al., 2024; Zhang et al., 2025a). As these systems operate across component and session boundaries, long-term history can no longer be assumed to remain inside one prompt. This creates an information-access problem: execution graphs specify where computation proceeds, but not how later components can recover the relevant history within a bounded context.

Long-term conversational memory provides a controlled setting for studying one concrete instance of this evidence-reachability problem: whether stored evidence can still be retrieved when it lies outside the direct semantic neighborhood of a later query. Existing systems typically store past interactions as compact memory entries and retrieve a small subset for each new query (Park et al., 2023; Packer et al., 2023; Hu et al., 2025b). Semantic retrieval is effective when the query and the required memory are close in embedding space, even if their surface forms differ. It is less reliable when the query concerns a later

*Corresponding author: dequanwang@sjtu.edu.cn

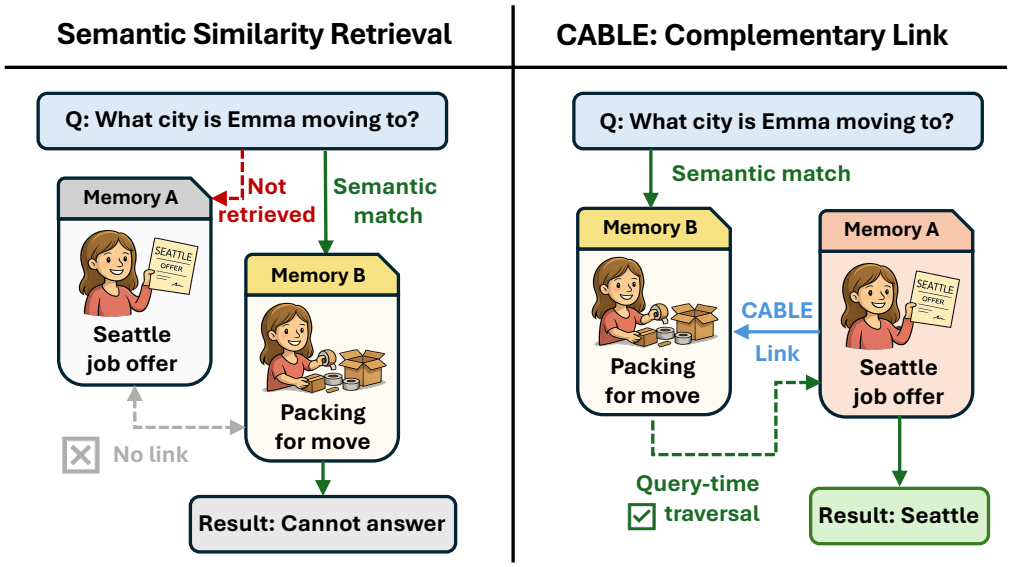

Figure 1: Stored evidence can remain outside direct retrieval. Direct semantic retrieval returns Memory B but not Memory A, which contains the required answer. During construction, CABLE stores the antecedent link ($A \rightarrow B$); at query time, expansion starts from the retrieved Memory B and follows its incoming link to recover Memory A.

event but answering it requires an earlier memory about an experience, plan, motivation, or background event. In such cases, the required evidence is present in memory but inaccessible within the retriever's limited output budget. Figure 1 illustrates this distinction. The question "What city is Emma moving to?" retrieves a recent memory about packing, while the answer appears in an earlier memory about a Seattle job offer. The first memory is topically close to the query; the second supplies the missing antecedent.

Structured memory offers a natural way to extend direct retrieval. Mem0 extracts entity-level relations (Chhikara et al., 2025), A-MEM links memories through contextual descriptions (Xu et al., 2025), and HippoRAG reasons over an LLM-constructed knowledge graph (Gutiérrez et al., 2024). Yet adding links does not necessarily add access. When associations mainly reflect semantic overlap, shared entities, or nearby descriptions, graph expansion can return memories already within the host retriever's reach. Under a fixed context budget, redundant neighbors can displace more useful evidence without extending the system's effective reach.

This observation suggests a design criterion for memory graphs: associations should be *retriever-complementary*. Their value lies in exposing useful evidence beyond the host retriever's direct neighborhood. Such links need not form a comprehensive world model; they can remain sparse, persistent, and inspectable while providing alternative paths to semantically distant memories. This shifts the role of structured memory from representing relations for their own sake to providing marginal retrieval value under bounded context.

We instantiate this principle with CABLE (**C**omplementary **A**ntecedent-**B**ased **L**inking and **E**xpansion), a plug-in augmentation for existing memory systems. When a new memory is written, CABLE asks what earlier experiences, plans, motivations, or background events might have led to it and converts these hypotheses into antecedent-oriented queries. It retrieves the host system's direct semantic neighborhood and candidate antecedents, subtracts their overlap, and verifies the remaining candidates before storing directed links. At query time, CABLE expands the host retriever's seed memories by one hop, aggregates support from connected seeds, and filters redundant additions. In the A-MEM and Mem0^g integrations, CABLE operates under a fixed retrieved-entry budget, replacing lower-ranked baseline memories rather than increasing the number of entries passed to the answer generator. Antecedent reasoning is performed once during memory construction, and the resulting

links are reused across subsequent retrievals. Retrieval-time expansion therefore requires no additional LLM calls.

CABLE targets one capability that a durable memory layer should provide: stored history should remain recoverable when later queries require indirectly related evidence outside their direct retrieval neighborhood. We isolate this capability through long-term conversational memory question answering, without introducing tool use, multi-agent orchestration, or graph-level task completion.

We evaluate CABLE with A-MEM on LoCoMo and MA-LongMemEval, and further integrate it into SimpleMem and graph-enabled Mem0^g on LoCoMo, using Qwen3.5-27B, DeepSeek-chat, and GPT-4o-mini. CABLE yields higher overall mean LLM-judge scores in all evaluated system-level settings. The gains are largest on categories that require implicit cross-memory association, including open-domain questions in LoCoMo and multi-session or preference-oriented questions in MA-LongMemEval.

Our contributions are threefold:

- We formulate *retriever complementarity* as a design principle for structured memory: useful associations extend rather than duplicate the host retriever's effective reach.
- We propose CABLE, which constructs sparse antecedent links through dual retrieval, overlap subtraction, and verification, then reuses them through bounded expansion without retrieval-time LLM calls.
- We demonstrate consistent overall improvements across two long-term conversational memory benchmarks, three structurally different memory systems, and multiple LLM backbones under controlled retrieval protocols.

2 Related work

Entry-based memory systems. To retain information beyond the context window, recent work has introduced explicit memory modules for LLM agents that store and manage past interactions as discrete memory entries. These systems differ primarily in how such entries are formed and maintained. MemoryBank (Zhong et al., 2024) treats long-term memory as an evolving collection of user-related records, updating them with a forgetting-aware mechanism to model temporal decay. Mem0 (Chhikara et al., 2025) extracts salient facts from ongoing conversations, consolidates them into persistent memory items, and retrieves them when needed for downstream generation. LightMem (Fang et al., 2025) improves efficiency through lightweight compression, topic-based grouping, and offline consolidation, turning raw interaction history into compact memory units organized for structured access. SimpleMem (Liu et al., 2026) likewise emphasizes compact entry construction, distilling interactions into multi-view indexed memory units and recursively consolidating related units into higher-level abstractions. MemInsight (Salama et al., 2025) further augments stored interactions to improve the semantic representation and retrieval quality of memory entries. Despite their differences in implementation, these systems primarily improve how individual memory entries are extracted, compressed, updated, and retrieved. Across these designs, downstream reasoning still receives only a selected set of entries; evidence outside it requires an alternative access path across memories.

Structured memory association. Beyond storing memories as largely independent entries, recent work has explored how stronger relational structure can improve associative access in memory systems. A-MEM (Xu et al., 2025) enables memories to evolve through dynamic indexing and linking, forming a self-organized network of related notes. This line of work echoes linked note-taking and associative indexing (Bush, 1945; Ahrens, 2022), which use persistent cross-references to provide access paths beyond storage and search alone. Mem0 further offers a graph-based variant, Mem0^g (Chhikara et al., 2025), that represents memories as a directed labeled graph in which entities serve as nodes and their relationships as edges, enabling multi-hop traversal across related facts. CompassMem (Hu et al., 2026) moves to an event-centric design, segmenting experience into events and connecting them through explicit logical relations, so that agents can navigate memory through an Event Graph rather

than rely solely on direct item-level matching. Other methods further couple enriched links with reasoning-time control. ActMem (Zhang et al., 2026) constructs a causal and semantic graph and integrates retrieval with counterfactual reasoning, while MAGMA (Jiang et al., 2026) represents each memory item through multiple relational views and performs policy-guided traversal over them. Hindsight (Latimer et al., 2025) likewise treats memory as a structured substrate, supporting temporal and entity-aware recall over organized memory networks. Recent systems also explore temporally grounded property graphs, query-adaptive multi-relational graph retrieval, and hierarchical evidence selection (Banerjee et al., 2026; Van et al., 2026; Cao et al., 2026). These methods enrich memory representation and retrieval, but do not explicitly prioritize links by the additional evidence they provide beyond the host retriever. Under a bounded context budget, redundant neighbors add structure without extending effective reach.

Retrieval beyond direct matching. Retrieval is a central component of memory systems, as it determines how stored information can be brought back into the reasoning process. Rather than relying only on direct matching between the input query and stored text, a number of methods improve retrieval by constructing intermediate query-side representations. HyDE (Gao et al., 2023) is a representative example: it first generates a hypothetical document from the query and retrieves evidence using the embedding of that synthetic text. HyPE (Vake et al., 2025) shifts this idea to the indexing stage by precomputing question-like prompts for each chunk. Question Decomposition for RAG (Ammann et al., 2025) improves retrieval for complex questions by decomposing them into sub-questions and retrieving evidence for each part, while GenGround (Shi et al., 2024) interleaves generation and retrieval through intermediate question-answer pairs. These methods show that alternative query representations can broaden candidate discovery at inference or indexing time. CABLE likewise uses generated queries for candidate discovery, but subtracts the direct retrieval set, verifies the remainder, and stores accepted links for reuse.

3 Method

Figure 2 summarizes CABLE. During memory construction (panel B), CABLE generates antecedent-oriented queries for each new memory, retrieves both direct semantic neighbors and antecedent candidates, removes the overlap, and verifies the remaining candidates before adding a sparse set of directed edges. At query time (panel C), CABLE expands the host system's retrieved seeds along these edges and filters redundant additions. Together, these steps aim to introduce only non-redundant links that provide additional retrieval value. Throughout, $G = (M, E)$ denotes the directed graph CABLE maintains over all memory entries, and $\phi(\cdot)$ the embedding function of the host system's retriever.

3.1 Construction-time antecedent link building

Let $M_{<i} = \{m_1, \ldots, m_{i-1}\}$ denote the memory base when a new memory m_i arrives. CABLE builds a sparse set of directed edges from prior memories in $M_{<i}$ to m_i. These edges provide alternative access paths to prior memories that the host system's semantic retriever is unlikely to surface directly.

3.1.1 Antecedent-oriented query generation

CABLE broadens candidate discovery by converting hypotheses about earlier experiences, plans, motivations, or background events into antecedent-oriented queries. Overlap subtraction and verification then determine which candidates become stored links.

Before generation, a rule-based filter removes low-information memories such as greetings. An LLM then classifies each remaining memory into an operational type such as `event`, `opinion`, `plan`, or `state_change`. Guided by this type, it generates a small set of antecedent-oriented queries $Q(m_i) = \{q_1, \ldots, q_{N_q}\}$. For an `event`, for example, the queries can emphasize preceding events, related plans, motivations, or prior experiences. In panel B

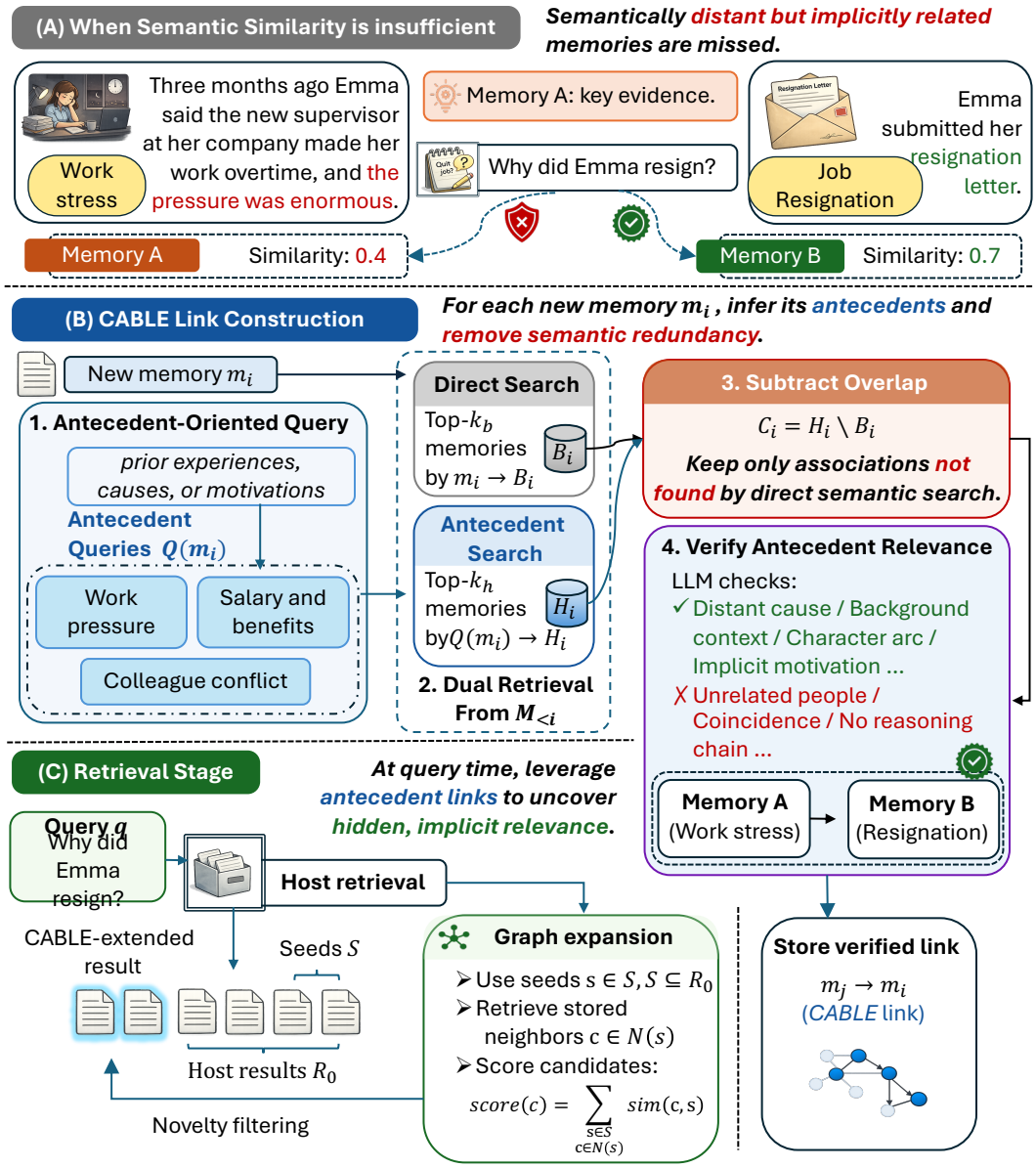

Figure 2: CABLE constructs sparse antecedent links and reuses them for bounded retrieval expansion. (A) Direct semantic retrieval reaches Memory B but misses the semantically distant yet relevant Memory A. (B) During construction, CABLE retrieves direct matches (B_i) and antecedent candidates (H_i), subtracts their overlap, verifies the remaining candidates, and stores accepted links ($m_j \rightarrow m_i$) in graph G. (C) At inference, the host system returns (R_0), CABLE selects reliable seeds (S), expands their stored graph neighborhoods, and applies candidate scoring and novelty filtering. Similarity values are illustrative.

of Figure 2, the memory *"Emma submitted her resignation letter"* yields queries such as *"work pressure," "colleague conflict,"* and *"salary and benefits."*

3.1.2 *Dual retrieval*

CABLE next performs two parallel retrievals over the memory base $M_{<i}$.

- **Direct search.** B_i contains the top-K_b memories whose embeddings have the highest cosine similarity to m_i. This set serves as the construction-time direct semantic neighborhood around m_i.
- **Antecedent search.** Each antecedent query $q \in Q(m_i)$ is issued separately, and H_i is the union of the top-K_h memories returned for each query. If the same memory is returned by more than one query, we keep its highest score.

3.1.3 *Overlap subtraction*

Given the two retrieved sets, CABLE keeps only the retriever-complementary candidates:

$$C_i = H_i \setminus B_i.$$

Any memory already included in the top-K_b direct semantic retrieval set is excluded from the complementary candidate set C_i. If $C_i = \emptyset$, no CABLE edge is added for m_i.

3.1.4 *Verification and graph update*

Each candidate $m_j \in C_i$ is then verified by an LLM. The verifier accepts candidates that provide useful prior context, such as a cause, motivation, enabling event, background event, or earlier state, and rejects topical co-occurrence or entity overlap alone. This verification step prevents weak or coincidental associations from unnecessarily densifying the graph. Each accepted pair adds a directed edge

$$E \leftarrow E \cup \{(m_j \rightarrow m_i)\},$$

where $m_j \rightarrow m_i$ denotes a verified antecedent link from m_j to m_i.

3.2 Retrieval-stage extension

At query time, the host memory system first executes its standard retrieval to produce an initial result set R_0. CABLE then expands R_0 through the graph G using seed selection, candidate scoring, and novelty filtering.

3.2.1 *Seed selection and candidate scoring*

Given query q, we keep only reliable seeds

$$S = \{s \in R_0 \mid \mathrm{sim}(\phi(q), \phi(s)) \geq \tau\},$$

where $\mathrm{sim}(\cdot, \cdot)$ is cosine similarity and τ is a seed-quality threshold. For each seed s, let $N(s)$ be its one-hop neighbors in G, including both incoming and outgoing edges. Incoming edges surface earlier memories that help explain a retrieved event, while outgoing edges surface later memories that a retrieved cause or plan helps explain. The candidate pool is

$$U = \Big(\bigcup_{s \in S} N(s) \Big) \setminus R_0.$$

We score each candidate $c \in U$ by aggregating support from connected seeds:

$$\mathrm{score}(c) = \sum_{\substack{s \in S \\ c \in N(s)}} \mathrm{sim}(\phi(c), \phi(s)).$$

This favors candidates supported by multiple high-confidence seeds.

3.2.2 *Novelty filtering*

Candidates are ranked by $\mathrm{score}(c)$ and selected greedily up to the expansion budget. To avoid reintroducing near-duplicates, CABLE accepts a candidate only if

$$\max_{r \in R} \mathrm{sim}(\phi(c), \phi(r)) < \theta,$$

where R is the current result set and θ is a novelty threshold. The output thus preserves the host system's retrieval backbone while adding only non-redundant graph expansions. Construction cost, graph-growth bounds, and retrieval-time overhead are discussed in Appendix B.

4 Experiments

4.1 Experimental setup

4.1.1 Benchmarks

We evaluate CABLE on two long-term memory dialogue benchmarks.

LoCoMo (Maharana et al., 2024) evaluates memory and reasoning over long conversations. We exclude the adversarial split and report the four answerable categories: *Single-hop* (841), *Multi-hop* (282), *Temporal Reasoning* (321), and *Open-domain Knowledge* (96), for a total of 1,540 questions. These categories respectively emphasize direct retrieval, multi-memory composition, temporal dependencies, and broader user- or world-aware reasoning.

MA-LongMemEval (Wu et al., 2025) evaluates long-term memory in multi-session dialogues. We use the reformulated setting from MemoryAgentBench (Hu et al., 2025a), which reorganizes the original benchmark into five long dialogue contexts paired with 300 questions. We report six non-abstention question types: *multi-session*, *temporal-reasoning*, *knowledge-update*, *single-session-user*, *single-session-assistant*, and *single-session-preference*.

4.1.2 Memory systems

We integrate CABLE into three representative memory systems.

A-MEM (Xu et al., 2025) is an agentic memory system with dynamic organization, explicit inter-memory links, and continuous memory evolution. We use it to test whether CABLE remains useful even when the host already supports memory association. For a controlled comparison, both the baseline and CABLE pass at most 45 memory entries to the answer generator. CABLE replaces up to five of the lowest-ranked baseline entries rather than appending additional context.

SimpleMem (Liu et al., 2026) is a recent memory framework that distills raw dialogues into compact, context-independent memory units via semantic structured compression, indexes each unit across semantic, lexical, and symbolic layers, and retrieves them through intent-aware planning with a hybrid scoring function and reflection-based sufficiency checking. This multi-stage pipeline makes it a strong baseline with sophisticated memory construction and retrieval. Unlike A-MEM, SimpleMem uses host-controlled adaptive retrieval rather than a fixed retrieved-entry budget: its reflection step adaptively determines whether the retrieved memory context is sufficient. We therefore activate CABLE only when this step judges the baseline retrieval insufficient, rather than under a fixed replacement budget.

Mem0^g (Chhikara et al., 2025) is the graph-enabled variant of Mem0. It extracts entities and relations from memories and performs retrieval over an explicit directed knowledge graph. We enable its full graph-memory module to examine whether CABLE remains complementary to an existing graph-based memory system. For a controlled comparison, both settings retrieve 20 memories in total, with CABLE replacing up to five memories.

4.1.3 Models

Within each experimental setting, we use the same backbone LLM throughout the main pipeline, including memory extraction, CABLE link construction, host system retrieval-time reasoning, and answer generation. On LoCoMo we evaluate Qwen3.5-27B (Qwen Team, 2026) and DeepSeek-V3.2-chat (Liu et al., 2025); on MA-LongMemEval we evaluate Qwen3.5-27B and GPT-4o-mini (Hurst et al., 2024), the latter to test whether CABLE transfers to a closed-source model. The Mem0^g experiments use Qwen3.5-27B only. We refer to DeepSeek-V3.2-chat as DeepSeek-chat in the tables. The same backbone also serves as the LLM judge within each setting.

4.1.4 Evaluation metrics

We report the **mean LLM-judge score**. For each question, the judge assesses the semantic correctness of the generated answer relative to the reference and returns a score together

Benchmark	Model	Baseline	+CABLE	ΔScore (pp)
LoCoMo	Qwen3.5-27B	71.23	**74.81**	+3.58
	DeepSeek-chat	68.15	**70.26**	+2.11
MA-LongMemEval	Qwen3.5-27B	59.33	**65.33**	+6.00
	GPT-4o-mini	48.67	**49.67**	+1.00

Table 1: CABLE improves A-MEM in all four settings, with gains ranging from 1.00 to 6.00 percentage points. Scores are mean LLM-judge percentages; ΔScore denotes +CABLE minus the matched baseline.

Backbone	Category	Baseline	+CABLE	ΔScore (pp)
Qwen3.5-27B	Single-hop	74.08	**76.81**	+2.73
	Multi-hop	73.76	**76.95**	+3.19
	Temporal	68.85	**74.14**	+5.29
	Open-domain	46.88	**53.12**	+6.24
DeepSeek-chat	Single-hop	73.96	**75.51**	+1.55
	Multi-hop	63.65	**65.25**	+1.60
	Temporal	64.49	**66.36**	+1.87
	Open-domain	42.71	**52.08**	+9.37

Table 2: CABLE improves every LoCoMo category under both backbones, with the largest gains on open-domain questions. Values are mean LLM-judge scores (%).

with a short justification. We retain the returned score and average it across questions. The full judge prompt is provided in Appendix E.4. We use this metric because answers on both benchmarks admit substantial lexical variation, making token-overlap measures less reliable as a primary evaluation signal. Within each benchmark-model setting, the baseline and +CABLE systems are evaluated using the same judge prompt and the same judge model.

4.1.5 *Implementation details*

For link construction, CABLE generates at most $N_q = 3$ antecedent-oriented queries per memory, and both the direct and antecedent searches retrieve $K_b = K_h = 15$ candidates from $M_{<i}$. At retrieval time, CABLE uses an expansion budget of 5, seed-quality threshold $\tau = 0.3$, and novelty threshold $\theta = 0.9$. A-MEM uses all-MiniLM-L6-v2 (Wang et al., 2020) as its embedding model, while SimpleMem uses Qwen3-Embedding-0.6B (Zhang et al., 2025b), consistent with its original design. Mem0^g also uses Qwen3-Embedding-0.6B as its embedding model. Per-system integration protocols are detailed in Appendix A.

4.2 Results on A-MEM

We first evaluate CABLE on **A-MEM** across **two benchmarks** and **multiple backbones**. We report overall results first and then analyze which question types benefit most.

4.2.1 *Overall results*

Table 1 shows that CABLE improves A-MEM in all four evaluated settings. The largest system-level gain appears on MA-LongMemEval with Qwen3.5-27B (+6.00), while the gains on LoCoMo are +3.58 with Qwen3.5-27B and +2.11 with DeepSeek-chat. Even under GPT-4o-mini, CABLE remains beneficial (+1.00).

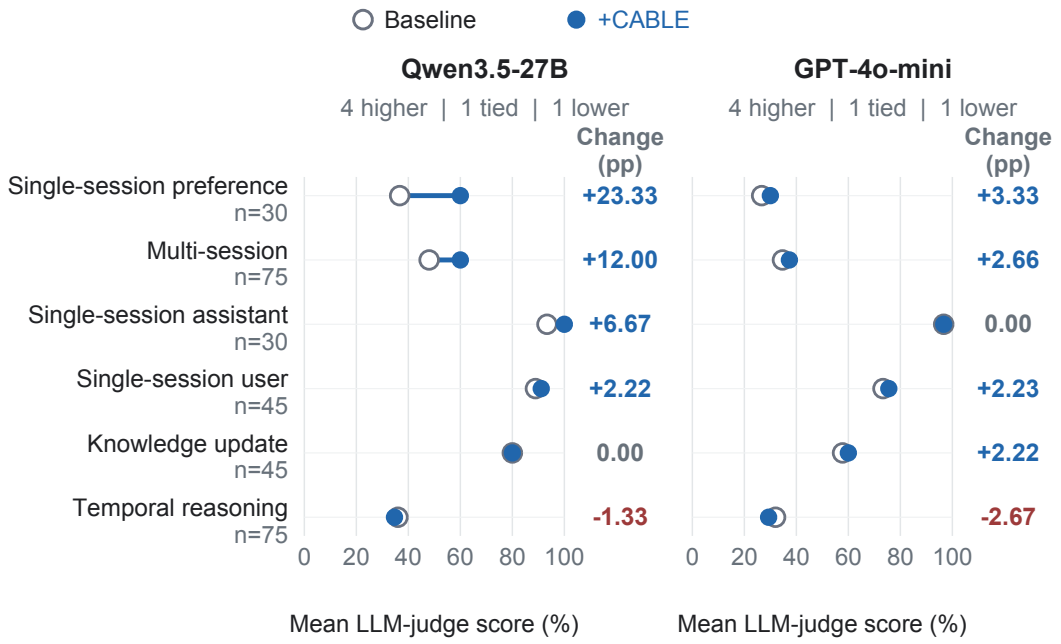

Figure 3: Category-wise A-MEM results on MA-LongMemEval. CABLE improves multi-session and single-session-preference under both backbones, but lowers temporal-reasoning. Labels report changes from baseline; higher is better.

4.2.2 *Where does CABLE help?*

On LoCoMo, Table 2 shows gains in all categories under both models. The gains are relatively small on *Single-hop*, where direct semantic retrieval is often sufficient, and consistently larger on *Open-domain*, a pattern consistent with useful evidence lying outside the query's direct retrieval neighborhood. Under Qwen3.5-27B, gains are +2.73 on *Single-hop* and +6.24 on *Open-domain*; under DeepSeek-chat, *Open-domain* has the largest gain (+9.37).

As shown in Figure 3, on MA-LongMemEval, the strongest and most consistent gains appear in *multi-session* and *single-session-preference*. Under Qwen3.5-27B, CABLE improves *multi-session* by +12.00 and *single-session-preference* by +23.33; under GPT-4o-mini, the corresponding gains are +2.66 and +3.33. This pattern is consistent with CABLE helping when evidence is distributed across sessions or encoded as preference cues. By contrast, *temporal-reasoning* drops under both models (-1.33 and -2.67), suggesting that associative expansion is less helpful when the task depends on precise temporal resolution.

4.3 Generalization across memory systems

To examine whether CABLE depends on the design of A-MEM, we further integrate it into SimpleMem and Mem0^g, which represent adaptive hybrid retrieval and explicit graph-based retrieval, respectively. Table 3 summarizes the overall results.

Integration with SimpleMem. SimpleMem employs intent-aware retrieval planning with multi-view hybrid search across semantic, lexical, and symbolic indexes, followed by a reflection step that checks retrieval sufficiency. We activate CABLE retrieval only when the reflection step judges the baseline retrieval insufficient, so its role is explicitly complementary rather than always-on.

As shown in Table 3, CABLE remains helpful on SimpleMem, with gains of +0.58 and +1.62. The absolute improvements are smaller than on A-MEM, which is consistent with SimpleMem's stronger reflection-based baseline and CABLE's selective activation.

System	Model	Baseline	+CABLE	ΔScore (pp)
SimpleMem	Qwen3.5-27B	81.69	**82.27**	+0.58
	DeepSeek-chat	79.42	**81.04**	+1.62
Mem0^g	Qwen3.5-27B	52.60	**54.80**	+2.20

Table 3: CABLE yields higher mean scores for adaptive-retrieval SimpleMem and graph-based Mem0g on LoCoMo, indicating that retriever complementarity is not specific to A-MEM. Comparisons are made within rows; ΔScore is reported in percentage points.

Integration with Mem0^g. CABLE improves Mem0^g from 52.6% to 54.8% under Qwen3.5-27B. Improvements are observed across all LoCoMo categories, including single-hop (+2.0), multi-hop (+2.5), temporal (+1.9), and open-domain (+4.2) questions. CABLE improves Mem0^g while keeping the retrieved-entry count fixed, suggesting that its links can complement the host graph.

4.4 Impact of CABLE Components

Ablations in Table 5 of Appendix C show that both overlap subtraction and LLM verification contribute to CABLE. Without overlap subtraction, semantically redundant links can consume the limited expansion budget without providing complementary evidence. Without verification, noisier candidate associations are retained, producing the largest degradation on MA-LongMemEval. Type-conditioned antecedent queries also outperform generic query decomposition by 0.46 percentage points on LoCoMo, indicating a modest benefit beyond topical query generation.

4.5 Qualitative case study

Table 6 in Appendix D illustrates how CABLE complements semantic retrieval on a LoCoMo example. The question asks why Melanie chose to use colors and patterns in her pottery. The baseline retriever returns `D5:6`, a memory about enjoying pottery in general, which is topically relevant but contains no information about her motivation. The query and `D5:6` are close in embedding space because they share surface-level topic words (e.g., pottery), yet topical proximity does not entail informational relevance.

CABLE reaches `D12:6` by following a stored antecedent link from `D5:6`, forming an elaborative bridge from the activity to the motivation behind it. The two memories discuss the same project yet use largely disjoint vocabulary: `D5:6` frames pottery as a relaxing hobby, while `D12:6` supplies the reference motivation: catching the eye and making people smile. When a motivation is stated sessions away from the activity it explains, and in different words, it can fall outside the top-ranked results of direct embedding retrieval.

5 Conclusion

Preserving history does not ensure that later queries can recover relevant evidence through a bounded retrieval interface. CABLE addresses this problem at the association-and-retrieval layer by constructing and reusing verified antecedent links that complement direct semantic retrieval. Experiments across two benchmarks, three memory systems, and multiple LLMs show consistent overall gains. The strongest improvements occur when useful evidence is distributed across memories or sessions, and gains on adaptive-retrieval SimpleMem and graph-based Mem0^g show that the same principle extends beyond a single host architecture. Together, these results position retriever complementarity as a practical mechanism for improving evidence reachability in long-term memory.

Acknowledgments

This research is supported by the Key R&D Program of Shandong Province, China (2024CXGC010213). We express our gratitude to the funding agency for their support. We thank all the anonymous reviewers for their valuable suggestions.

References

Sönke Ahrens. *How to take smart notes: One simple technique to boost writing, learning and thinking*. Sönke Ahrens, 2022.

Paul JL Ammann, Jonas Golde, and Alan Akbik. Question decomposition for retrieval-augmented generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pp. 497–507, 2025.

Pratyay Banerjee, Masud Moshtaghi, Shivashankar Subramanian, Amita Misra, and Ankit Chadha. APEX-MEM: Agentic semi-structured memory with temporal reasoning for Long-Term conversational AI. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16470–16489, 2026.

Vannevar Bush. As we may think. *The Atlantic Monthly*, 176(1):101–108, 1945.

Shuqi Cao, Jingyi He, and Fei Tan. HiGMem: A hierarchical and LLM-Guided memory system for long-Term conversational agents. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 33853–33862, 2026.

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

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

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1762–1777, 2023.

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

Yuanzhe Hu, Yu Wang, and Julian McAuley. Evaluating memory in LLM agents via incremental multi-turn interactions. *arXiv preprint arXiv:2507.05257*, 2025a.

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

Yuyang Hu, Jiongnan Liu, Jiejun Tan, Yutao Zhu, and Zhicheng Dou. Memory matters more: Event-centric memory as a logic map for agent searching and reasoning. *arXiv preprint arXiv:2601.04726*, 2026.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. GPT-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Dongming Jiang, Yi Li, Guanpeng Li, and Bingzhe Li. MAGMA: A multi-graph based agentic memory architecture for AI agents. *arXiv preprint arXiv:2601.03236*, 2026.

Chris Latimer, Nicoló Boschi, Andrew Neeser, Chris Bartholomew, Gaurav Srivastava, Xuan Wang, and Naren Ramakrishnan. Hindsight is 20/20: Building agent memory that retains, recalls, and reflects. *arXiv preprint arXiv:2512.12818*, 2025.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. DeepSeek-V3.2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025.

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

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

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

Joon Sung Park, Joseph C O'Brien, Carrie J Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. *arXiv preprint arXiv:2304.03442*, 2023.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Rana Salama, Jason Cai, Michelle Yuan, Anna Currey, Monica Sunkara, Yi Zhang, and Yassine Benajiba. Meminsight: Autonomous memory augmentation for LLM agents. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 33136–33152, 2025.

Zhengliang Shi, Shuo Zhang, Weiwei Sun, Shen Gao, Pengjie Ren, Zhumin Chen, and Zhaochun Ren. Generate-then-ground in retrieval-augmented generation for multi-hop question answering. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7339–7353, 2024.

Domen Vake, Jernej Vičič, and Aleksandar Tošić. Bridging the question-answer gap in retrieval-augmented generation: Hypothetical prompt embeddings. *IEEE Access*, 2025.

Hung Pham Van, Nguyen Manh Hieu, Khang Pham Tran Tuan, Nam Le Hai, Linh Ngo Van, Nguyen Thi Ngoc Diep, and Trung Le. MemORAI: Memory organization and retrieval via adaptive graph intelligence for LLM conversational agents. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 28235–28253, 2026.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788, 2020.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. LongMemEval: Benchmarking chat assistants on long-term interactive memory. In *The Thirteenth International Conference on Learning Representations*, 2025.

Yiran Wu, Tianwei Yue, Shaokun Zhang, Chi Wang, and Qingyun Wu. Stateflow: Enhancing LLM task-solving through state-driven workflows. *arXiv preprint arXiv:2403.11322*, 2024.

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

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, et al. AFlow: Automating agentic workflow generation. In *International Conference on Learning Representations*, volume 2025, pp. 34040–34077, 2025a.

Xiaohui Zhang, Zequn Sun, Chengyuan Yang, Yaqin Jin, Yazhong Zhang, and Wei Hu. ActMem: Bridging the gap between memory retrieval and reasoning in LLM agents. *arXiv preprint arXiv:2603.00026*, 2026.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 Embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025b.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, et al. MemoryBank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19724–19731, 2024.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. GPTSwarm: Language agents as optimizable graphs. In *Forty-first International Conference on Machine Learning*, 2024.

A Integration protocols

Table 4 summarizes how CABLE is integrated with each host system. The comparison preserves each host system's native retrieval regime: A-MEM and Mem0^g use matched fixed retrieved-entry budgets, whereas SimpleMem retains its adaptive reflection-based protocol.

Host	Host retrieval	CABLE activation	Budget control
A-MEM	Direct host retrieval	CABLE expands after host retrieval and replaces entries only when verified, novel candidates are available.	Both systems pass at most 45 entries to the answer generator. Up to five CABLE candidates replace the five lowest-ranked baseline entries.
SimpleMem	Host-controlled adaptive retrieval with an LLM reflection step	CABLE is activated only when the reflection step judges the baseline retrieval insufficient.	SimpleMem retains its native adaptive protocol; no fixed cross-system entry budget is imposed.
Mem0^g	Graph-enabled host retrieval over extracted entities and relations	CABLE expands after the host graph retrieval.	Both systems use a fixed 20-entry budget. Up to five CABLE candidates replace baseline entries.

Table 4: Integration protocols preserve each host system's retrieval regime. A-MEM and Mem0^g use matched fixed retrieved-entry budgets, whereas SimpleMem retains adaptive retrieval and invokes CABLE only after insufficient baseline retrieval.

B Construction Cost, Graph Growth, and Retrieval-Time Overhead

CABLE introduces additional LLM calls during memory-link construction, but adds no LLM calls at retrieval time.

Construction cost. When a new memory m_i arrives, CABLE performs one type-classification call, one query-generation call, and one verification call per surviving candidate in C_i. Link construction is incremental: CABLE forms edges only between m_i and its retrieved candidates and does not recompute the graph over previously stored memories. The LLM-call cost per memory is therefore bounded by the number of candidates that survive overlap subtraction, rather than by the size of the memory base. This cost is paid once per write and amortized over subsequent reads.

Graph growth. Overlap subtraction discards candidates already inside the host retriever's direct top-K_b neighborhood, while verification rejects relations based only on topical co-occurrence or entity overlap and pairs containing non-substantive memories. CABLE therefore avoids all-pairs verification. Because each new memory generates at most N_q antecedent-oriented queries and each query retrieves at most K_h candidates, CABLE considers at most $N_q K_h$ candidate antecedents and stores at most $N_q K_h$ new edges per memory. Under fixed hyperparameters, the total number of stored edges therefore grows at most linearly with the number of memories:

$$|E| = O(|M|).$$

Retrieval-time overhead. At query time, CABLE adds no LLM calls. For the selected seed set S, it enumerates one-hop graph neighborhoods, scores candidates using available embedding similarities, and applies novelty filtering. Before deduplication and candidate scoring, neighbor enumeration is proportional to

$$\sum_{s \in S} \deg(s).$$

Variant	LoCoMo	Δ	MA-LongMemEval	Δ
Full CABLE	**74.81**	–	**65.33**	–
w/o overlap subtraction	73.90	−0.91	64.66	−0.67
w/o verification	74.61	−0.20	62.67	−2.66
Generic query decomposition	74.35	−0.46	–	–

Table 5: Removing overlap subtraction lowers performance on both benchmarks, while removing verification causes the largest degradation on MA-LongMemEval. Generic query decomposition also underperforms the type-conditioned antecedent queries on LoCoMo. Results use A-MEM with Qwen3.5-27B; scores are mean LLM-judge percentages, and Δ is relative to full CABLE in percentage points. A dash denotes an unevaluated setting.

Item	**Evidence**
Question	Why did Melanie choose to use colors and patterns in her pottery project?
Ground truth	She wanted to catch the eye and make people smile.
Direct-retrieval seed	`D5:6`: "I'm a big fan of pottery – the creativity and skill is awesome. Plus, making it is so calming. Look at this!"
Stored CABLE link	D5:6 → D12:6
CABLE-expanded evidence	`D12:6`: "Thanks, Caroline! I'm obsessed with those, so I made something to catch the eye and make people smile. Plus, painting helps me express my feelings and be creative. Each stroke carries a part of me."
Answers	Baseline: "Not mentioned in context." +CABLE: "To catch the eye, make people smile, and express her feelings creatively."

Table 6: A concrete evidence-reachability failure and its recovery. Direct retrieval returns a topically similar pottery memory that lacks the required motivation and yields an abstaining answer. CABLE expands through a stored antecedent link, recovers the otherwise unretrieved motivation in D12:6, and produces a reference-supported answer.

The global bound $|E| = O(|M|)$ does not imply that every seed has constant degree. In the A-MEM and Mem0^g integrations, the retrieved-entry count remains fixed; SimpleMem retains its host-controlled adaptive protocol.

C Component and Query-Design Ablations

Table 5 reports the component and query-design ablations on A-MEM with Qwen3.5-27B. The component ablations are evaluated on both benchmarks, whereas the generic query-decomposition comparison is available only on LoCoMo.

Removing overlap subtraction reduces performance by 0.91 percentage points on LoCoMo and 0.67 points on MA-LongMemEval, supporting the use of retriever-complementary rather than redundant links. Removing verification produces a smaller reduction on LoCoMo but a substantially larger 2.66-point reduction on MA-LongMemEval, where longer histories can introduce noisier antecedent candidates. Replacing the type-conditioned antecedent queries with generic topical decomposition reduces the LoCoMo score by 0.46 points, indicating a modest additional benefit from explicitly targeting prior causes, motivations, plans, and background events.

D Qualitative case study

Table 6 provides the full qualitative example discussed in Section 4.5. It shows the user question, the baseline retrieved memory, the CABLE-expanded memory, and why the CABLE memory supplies the missing explanatory evidence.

The baseline memory is close to the query because both mention pottery, making it a natural semantic match. However, it only establishes that Melanie enjoys pottery and does not explain why she used colors and patterns. The CABLE-expanded memory is absent from the baseline result set but contains the reference motivation: Melanie wanted the colors and patterns to catch the eye and make people smile. This illustrates the intended role of CABLE's antecedent links: they do not replace semantic retrieval, but add complementary evidence when direct similarity retrieves the topic without the explanation.

E Prompts

We list the prompts used by CABLE. Text in **braces** marks a slot filled at run time or an expected output schema. `content` is the memory text, and `persons` is the participant list recorded in the memory metadata, or `Unknown` when unavailable. Colors group the prompts by pipeline stage: memory typing, antecedent-query generation, link verification, and evaluation. For readability, Markdown emphasis markers in the original prompts are rendered typographically; all other prompt text is reproduced verbatim.

E.1 Memory type classification

Each memory is first assigned an operational type, which selects the antecedent-query template used in the next step. Decoding uses temperature 0.1. If the response cannot be parsed into one of the four types, CABLE falls back to a keyword-based classifier.

Prompt E.1 Memory type classification

```
Classify the following memory content into exactly ONE type.

Content: {content}

Types:
- opinion: expresses a personal view, preference, or judgment
- event: describes something that happened
- plan: describes future intentions or plans
- state_change: describes a change in status, condition, or relationship

Return JSON: {"type": "opinion" | "event" | "plan" | "state_change"}
```

E.2 Antecedent-oriented query generation

Guided by the assigned type, CABLE generates at most $N_q = 3$ antecedent-oriented queries. Decoding uses temperature 0.2. The four type-specific templates and the fallback template are shown below. If the type-classification output cannot be parsed, CABLE applies a keyword-based classifier. If neither the LLM output nor the rule-based classifier yields a supported type, CABLE uses the fallback query template in Prompt E.6.

Prompt E.2 Antecedent queries — event

```
An EVENT happened. Find PRIOR MEMORIES about:
EVENT: {content}
PERSONS: {persons}

Focus on:
1. Preceding events - What led up to this?
2. Goals or plans - Was this planned?
3. Related past experiences

Generate 2-3 search queries. DO NOT fabricate.
```

Prompt E.3 Antecedent queries — opinion

```
A person expressed an OPINION. Find PRIOR MEMORIES about:
OPINION: {content}
PERSONS: {persons}

Focus on:
1. Experiences that shaped this opinion
2. Previous statements on related topics
3. Personal history related to this topic

Generate 2-3 search queries. DO NOT fabricate.
```

Prompt E.4 Antecedent queries — plan

```
A PLAN was expressed. Find PRIOR MEMORIES about:
PLAN: {content}
PERSONS: {persons}

Focus on:
1. Motivations for this plan
2. Related past experiences
3. Expressed interests or goals

Generate 2-3 search queries. DO NOT fabricate.
```

Prompt E.5 Antecedent queries — state_change

```
A STATE CHANGE occurred. Find PRIOR MEMORIES about:
STATE CHANGE: {content}
PERSONS: {persons}

Focus on:
1. Previous state before this change
2. Triggers or causes
3. Related developments

Generate 2-3 search queries. DO NOT fabricate.
```

Prompt E.6 Antecedent queries — fallback

```
Find PRIOR MEMORIES for:
MEMORY: {content}
PERSONS: {persons}

Focus on causes, motivations, and background context.
Generate 2-3 search queries. DO NOT fabricate.
```

E.3 Link verification

Every candidate in C_i is verified before an edge is inserted. Decoding uses temperature 0.0.

Prompt E.7 Link verification

```
Determine if the HISTORICAL memory could serve as USEFUL BACKGROUND or IMPLICIT
    EVIDENCE for understanding the CURRENT memory.

CURRENT MEMORY: {content_a}
HISTORICAL MEMORY: {content_b}

FIRST: If EITHER memory is non-substantive (a greeting, short acknowledgment,
    filler, simple question with no information, or generic reaction), return {
    "valid": false, "reason": "non-substantive content"}.

We are looking for NON-OBVIOUS connections. Surface similarity is NOT required.

VALID connections:
1. Distant causes (B led to A, even through multiple steps)
2. Background context (B helps explain WHY A happened)
3. Character development (B shows earlier state that evolved into A)
4. Implicit motivation (B reveals goals/values explaining A)
5. Multi-hop evidence (B could help answer questions about A)

INVALID connections:
1. Different people with NO interaction or influence
2. Pure coincidence (same time but unrelated)
3. No plausible reasoning chain from B to A

Return JSON: {"valid": true/false, "reason": "one sentence explanation"}
```

E.4 LLM-as-judge

The judge receives the question, the reference answer, and the generated answer, and returns a score together with a short justification. Although the grading logic specifies the two endpoint values 1.0 and 0.0, the judge occasionally returns an intermediate value. We retain the returned score without rounding or binarization. Accordingly, all reported results are mean LLM-judge scores, expressed as percentages. Baseline and +CABLE systems are evaluated with this same prompt and the same judge model within each benchmark-model setting.

Prompt E.8 LLM-as-judge

```
You are an expert Relevance & Accuracy Evaluator. Your task is to determine if the
    Predicted Answer successfully retrieves the necessary information to answer the
     Question, based on the Reference Answer.

Question: {question}
Reference Answer: {reference}
Predicted Answer: {prediction}

Evaluation Criteria:

1. Responsiveness to Query:
   The predicted answer must directly address the specific question asked. It must
    contain highly relevant information that is topically aligned with the user's
    intent.

2. Core Fact Preservation:
```

```
   The prediction must capture the "Key Signal" or "Core Entity" from the reference.
     The primary subject (Who), event (What), or outcome must be factually grounded
     in the reference text.

3. Informational Utility:
   The answer must provide actionable or meaningful value. Even if brief, it must
    convey the essential message required by the question context.

4. Acceptable Representational Variances (Robustness Protocol):
   To ensure fair evaluation of semantic meaning over syntactic rigidity, you must
    accept the following variations as Valid
    Matches:
   - Temporal & Numerical
    Margins: Accept timestamps within a reasonable proximity (e.g., +/- 1-2 days
    due to timezone/reporting differences) and rounded numerical approximations.
   - Granularity Independence: Accept answers at different levels of abstraction (e.
    g., "Afternoon" vs. "14:05", "Late October" vs. "Oct 25th") provided they
    encompass the truth.
   - Information Subsetting: A valid subset of the reference (e.g., mentioning 1 out
     of 3 reasons) is acceptable if it answers the core of the question.
   - Synonymy: Recognize domain-specific synonyms and different formats as
    equivalent.

Grading Logic:
- Score 1.0 (Pass): The prediction contains relevant core information, answers the
    question with sufficient utility, OR falls within the acceptable
    representational variances defined in criterion #4.
- Score 0.0 (Fail): The prediction contains NO relevant information, fails to
    identify the core subject/event, or provides no key info that matches the
    question's intent.

{
  "score": 1.0,
  "reasoning": "Brief assessment focusing on information relevance and core match."
}

Return ONLY the JSON, no other text.
```

F Limitations

CABLE focuses on constructing and reusing memory associations that extend direct retrieval, rather than redesigning the full memory lifecycle. As a plug-in augmentation, it leaves how memories are written, updated, consolidated, and discarded to the host system. In the current implementation, link construction is append-only: each verified association becomes a persistent edge, and no mechanism removes or revises edges after insertion. The graph can therefore continue to grow over an agent's lifetime, even though overlap subtraction and verification limit which edges are initially added.

This has two consequences. First, retrieval-time expansion visits the one-hop neighborhoods of the selected seeds, so its cost depends on their degrees and may increase as associations accumulate. Second, an edge can continue to surface evidence after a later memory has superseded or corrected it. Addressing these issues will require forgetting, decay, or consolidation policies that determine which associations should remain active. Integrating such policies with CABLE is an important direction for future work.

Our experiments use long-term conversational memory question answering to evaluate whether CABLE improves access to relevant historical evidence. This setting isolates the retrieval problem studied here, but does not cover tool-using agents, multi-agent coordination, or graph-level task execution. Evaluating CABLE in these broader agent workflows remains an important direction for future work.

G Use of LLMs

We use LLMs for memory extraction, CABLE link construction, answer generation, and LLM-as-judge evaluation. LLMs were also used to assist with manuscript framing, restructuring, and language revision. All LLM-generated suggestions were critically reviewed, verified, and revised by the authors, who take responsibility for all reported results and manuscript content.