

GRACE-Mem: Graph Retrieval with Agentic Curation of Evidence for Long-Term Conversational Memory

Heng Wang, Xuan Ni Chen, Pei-Yun Chang, Chih-Hsi Chen,
Hsin-Yu Huang, Chienyi Lin, I-Ling Chung

GIGABYTE Technology
arthur.wang0413@gigabyte.com

Abstract

Long-term assistants must answer questions over conversations beyond the context window while grounding responses in verifiable dialogue evidence. Existing memory systems retrieve compact representations, but these may omit critical details, while similarity search may favor related records over answer-bearing source turns. We introduce **GRACE-Mem**, a long-term memory architecture that separates candidate discovery, candidate reranking, and final evidence construction. It uses graph records for discovery, compressed records for reranking, and provenance-linked raw dialogue spans as final evidence. After hybrid graph retrieval, multi-hop expansion, and two-stage reranking, the top- K records seed an Evidence Curation Agent. The agent searches the raw corpus, removes irrelevant candidates, recovers missing evidence, and returns provenance identifiers to reconstruct the final context from raw dialogue. GRACE-Mem achieves **85.3%** on LoCoMo and **73.2%** on LongMemEval, surpassing the previously reported state of the art, AMA, by 7.9 and 3.4 percentage points, respectively.

1 Introduction

Long-term assistants must answer from conversations that exceed the context window while still supporting claims with verifiable dialogue evidence (Maharana et al., 2024; Wu et al., 2025). Existing memory systems therefore store summaries, facts, entities, or graph relations and retrieve only a small subset at answer time (Chhikara et al., 2025; Rasmussen et al., 2025). These compact records are useful indexes but fragile final evidence: generation or compression can omit speakers, temporal qualifiers, or conditions, while similarity retrieval can rank topically related records above answer-bearing source turns (Weller et al., 2025; Derehag et al., 2026). The central limitation

is that retrieval representations are often treated as answering evidence.

The core problem is thus not simply how to store more memory, but how to use compact memory to find the right source evidence. We introduce **GRACE-Mem**, a long-term memory architecture that separates three roles that are often conflated: graph records support candidate discovery, compressed records support candidate reranking, and raw dialogue spans provide complete, source-level answering evidence.

GRACE-Mem first retrieves and expands entity-relation candidates in a memory graph and reranks the structured records. It then merges their linked compressed records with directly retrieved ones and reranks the deduplicated pool. The top- K provenance-linked records seed an Evidence Curation Agent, which can verify, remove, or supplement candidates by inspecting the raw corpus. The final context is rebuilt from the selected raw dialogue spans; if curation fails, an identity fallback preserves the upstream compressed seed context.

Our main contributions are as follows:

- We propose a source-evidence-centric long-term memory architecture that separates retrieval representations from raw answering evidence and reconstructs the final context through provenance links.
- We introduce a complementary retrieval and hierarchical reranking method that selects entity and relation candidates at the graph level, then jointly reranks their linked compressed records with direct-retrieval results to improve the coverage and selection of answer-relevant evidence.
- We develop a post-retrieval Evidence Curation Agent that verifies and filters candidates against the raw dialogue, removes topically

related but insufficient evidence, and recovers source evidence missed by initial retrieval.

To support reproducibility, we release our code, prompt templates, and evaluation scripts.¹

2 Related Work

2.1 Long-Term Conversational Memory via Summarization and Compression

Longer context windows do not guarantee effective use of growing conversation histories: irrelevant background can dilute query-relevant evidence and distract the answer model. Memory systems therefore compress conversation histories into external representations and retrieve them on demand. MemGPT pages information between working context and external storage through an OS-inspired hierarchy (Packer et al., 2023), while SeCom segments conversations by topic and applies LLMingua-2 compression to reduce fragmentation and retrieval noise (Pan et al., 2025).

LangMem (LangChain, 2026) provides practical long-term memory tooling, while NEMORI (Ma et al., 2026) learns retention through adaptive memory distillation rather than fixed heuristics.

Generated summaries may omit source details or introduce interpretations not explicitly stated in the original dialogue. GRACE-Mem instead uses compressed records, entities, and relations as retrieval-oriented memory views, while preserving session-message indices to the corresponding raw dialogue spans for source-level evidence recovery.

2.2 Structured and Evolving Long-Term Memory

Structured memory connects entities, relations, and memory units rather than treating them as isolated fragments. A-MEM builds evolving associations among memory notes (Xu et al., 2025); Zep maintains a temporally aware knowledge graph (Rasmussen et al., 2025); and Mem0 applies LLM-driven memory operations to consolidate salient facts (Chhikara et al., 2025).

AMA (Huang et al., 2026) uses specialized agents for memory construction, retrieval, judging, and refresh; Other systems separately represent facts, experiences, summaries, and beliefs (Latimer et al., 2025).

However, practical memory construction must still manage duplicate entities, unsupported

merges, information updates, and source traceability. GRACE-Mem adopts a staged construction process that extracts entities before relations, combines lexical and semantic signals for entity alignment, and applies conservative ADD or UPDATE decisions when integrating new interactions. Relations are constructed only between reconciled entities, and each record retains an index to its corresponding raw dialogue span.

2.3 Structure-Aware Retrieval and Evidence Selection

Standard RAG retrieves a top- k set of candidates from an indexed corpus before generation (Lewis et al., 2020). In recurring conversations, however, semantic similarity alone may return memories that are topically related but insufficiently discriminative. LightRAG combines graph-based indexing with query-side keyword extraction, using low-level keywords to retrieve specific entities and details and high-level keywords to retrieve broader relational concepts (Guo et al., 2024). GRACE-Mem likewise extracts query keywords at retrieval time, but uses a single keyword set as an auxiliary lexical signal in hybrid retrieval.

Other graph-based retrieval methods provide complementary mechanisms. MRAgent allows an LLM to iteratively traverse a Cue-Tag-Content graph for adaptive retrieval (Ji et al., 2026). SARAG shows that spreading activation can connect multi-hop evidence without LLM-controlled traversal (Pavlović et al., 2025).

Similarity-based retrieval also has capacity limits (Weller et al., 2025), and reranking may remove gold evidence from a high-recall candidate pool (Derehag et al., 2026). Direct corpus interfaces address exact names, numbers, and combinations of sparse clues through adaptive text search (Sen et al., 2026; Li et al., 2026), but their effectiveness depends on the execution harness and evidence-delivery path. GRACE-Mem therefore separates candidate discovery from evidence selection: hybrid entity and relation retrieval seeds spreading activation, merged compressed-record candidates are reranked, and a GREP/READ/FINAL agent may selectively inspect the raw text. If this stage fails, identity fallback preserves the reranked compressed seed context.

Positioning of our work. GRACE-Mem integrates incremental graph memory, temporal normalization, graph expansion, and post-retrieval

¹<https://github.com/gbtailab-tech/GRACE-Mem>

corpus inspection within a unified architecture. Compared with flat top- k retrieval or multi-round LLM-controlled graph traversal, it keeps source-linked raw dialogue as the basis for evidence verification while using graph structure to connect candidate evidence.

3 Method

We first provide a brief overview of GRACE-Mem in Figure 1. During ingestion, GRACE-Mem converts individual dialogue units into complementary compressed-record, entity, and relation views, resolves newly extracted entities against existing memory, and preserves links to the original dialogue. During retrieval, the system combines hybrid entity and relation search with spreading activation, reranks the resulting graph records, collects their linked compressed records, merges them with directly retrieved records, and reranks the merged pool. The corpus-agent filter is applied after compressed-record reranking, as detailed in Algorithm 1.

The remainder of this section is organized as follows. Section 3.1 describes the memory state and ingestion pipeline. Section 3.2 presents hybrid entity and relation retrieval together with query-conditioned spreading activation. Finally, Section 3.3 details graph-level reranking, compressed-record candidate construction, and compressed-record reranking.

3.1 Memory State and Ingestion

At step t , GRACE-Mem maintains a memory graph and three auxiliary vector indexes:

$$\mathcal{M}_t = (G_t, \mathcal{V}_t^E, \mathcal{V}_t^R, \mathcal{V}_t^S), \quad G_t = (\mathcal{N}_t, \mathcal{R}_t, \mathcal{P}_t). \quad (1)$$

Here, $\mathcal{N}_t$, $\mathcal{R}_t$, and $\mathcal{P}_t$ denote entity nodes, relation edges, and provenance-linked source records, respectively. The auxiliary indexes $\mathcal{V}_t^E$, $\mathcal{V}_t^R$, and $\mathcal{V}_t^S$ store entity, relation, and compressed-record embeddings. The graph and indexes support retrieval, whereas the source records in $\mathcal{P}_t$ provide the authoritative dialogue evidence used for context reconstruction.

For each dialogue unit D_t , the ingestion pipeline produces a compressed dialogue record and sequentially extracts entities and relations:

$$s_t = \text{Compress}_\rho(D_t), \quad \bar{\mathcal{E}}_t = \Phi_{\text{ent}}(D_t), \quad (2) \\ \bar{\mathcal{R}}_t = \Phi_{\text{rel}}(D_t, \bar{\mathcal{E}}_t).$$

Here, s_t is the compressed record produced by LLM_{Lingua-2}, ρ is the retained-length ratio, and the barred variables denote uncommitted write-time records. Conditioning relation extraction on $\bar{\mathcal{E}}_t$ restricts relation endpoints to the previously extracted entities.

Entity embeddings are computed from the concatenation of each entity’s name, type, and description. Relation embeddings use the names of both endpoints, the relation description, and its keywords, while compressed-record embeddings are computed directly from s_t . Provenance fields are stored separately and are not included in the embedding inputs.

For each newly extracted entity e , the system retrieves possible matches from the existing entity set and decides whether to create a new node or update an existing one:

$$C(e) = \text{HybridRetrieve}(e, \mathcal{N}_{t-1}), \quad (3) \\ (\text{op}_e, \text{id}_e^*) = \Phi_{\text{op}}(e, C(e)).$$

The decision op_e is either ADD or UPDATE. For ADD, the system creates a new node; for UPDATE, id_e^* identifies the selected existing entity. Candidate matching takes the union of BM25 and dense-vector retrieval. The BM25 branch uses a score-relative threshold, whereas the dense branch applies a similarity threshold and a maximum candidate count.

Each dialogue unit is stored with its raw text, dialogue timestamp, session identifier, and message identifier. After entity resolution, the system writes the resolved entities, relations, compressed record, and provenance record to their corresponding memory structures. Before insertion, the endpoints of each extracted relation are replaced with the entity identifiers assigned during resolution.

3.2 Hybrid Graph Retrieval

Given a question q , the query analyzer extracts keywords and constructs normalized sentence- and keyword-level representations:

$$K_q = \Phi_{\text{qa}}(q) = \{k_1, \dots, k_m\}, \\ \mathbf{z}_q = \text{Embed}(q), \quad \mathbf{q}_{\text{vec}} = \frac{\mathbf{z}_q}{\|\mathbf{z}_q\|_2}, \quad (4) \\ \mathbf{z}_i = \text{Embed}(k_i), \\ \mathbf{v}_i = \frac{\mathbf{z}_i}{\|\mathbf{z}_i\|_2}, \quad i = 1, \dots, m.$$

The sentence-level representation $\mathbf{q}_{\text{vec}}$ is used for dense entity retrieval and query-conditioned

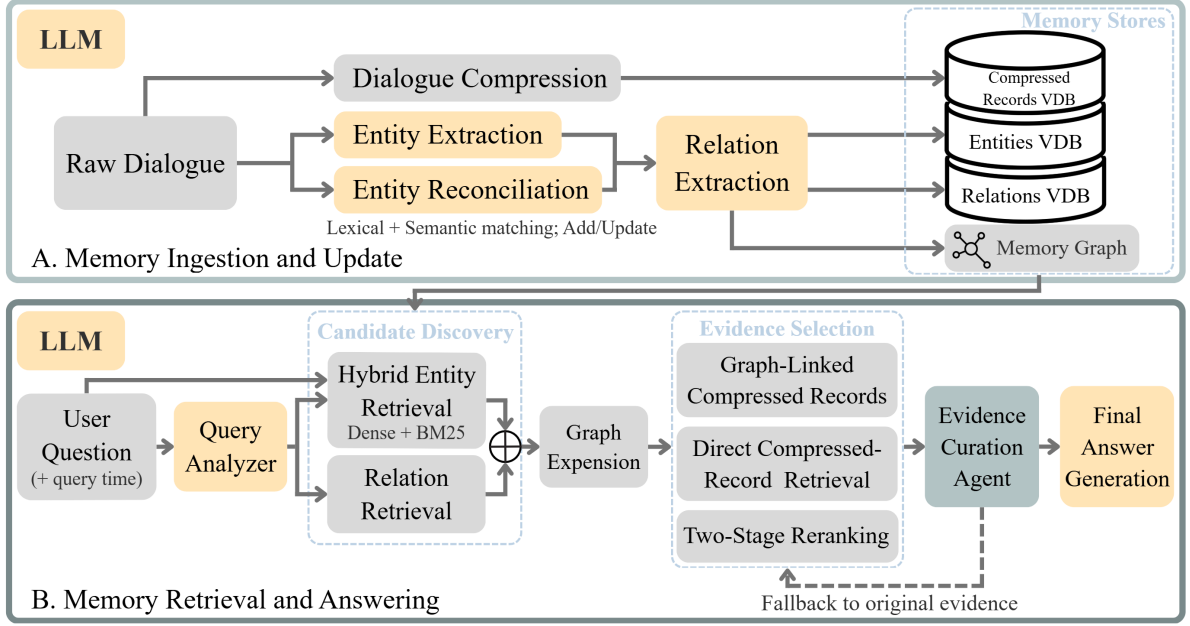


Figure 1: Overview of GRACE-Mem. The ingestion pipeline constructs complementary provenance-linked graph and vector views from each dialogue unit. The retrieval pipeline combines hybrid entity and relation retrieval, spreading activation, graph-level reranking, graph-linked and directly retrieved compressed-record candidate merging, compressed-record reranking, and corpus-agent evidence curation.

graph expansion. Each keyword k_i is used in textual form for sparse entity retrieval and in vector form $\mathbf{v}_i$ for dense relation retrieval. This separation preserves the complete query semantics for entity retrieval while retaining fine-grained relation cues from individual keywords. We denote the normalized embedding stored for relation r by $\mathbf{v}_r$.

3.2.1 Hybrid Entity and Relation Retrieval

The dense entity branch searches $\mathcal{V}_t^E$ with the complete query embedding and retains candidates above a similarity threshold. For sparse retrieval, each keyword independently searches the entity name and description fields. Its BM25 score is the maximum of the two field scores, and candidates are retained relative to the highest score obtained for that keyword.

The dense and sparse entity candidates form the unweighted seed set, while relation candidates are retrieved independently for each keyword embedding:

$$E^{(0)} = E^{\text{dense}} \cup \bigcup_{i=1}^m E_i^{\text{sparse}}, \quad (5)$$

$$R^{(0)} = \bigcup_{i=1}^m \text{TopK}_{K_R, \tau_r}(\mathbf{v}_i; \mathcal{V}_t^R).$$

Here, $\text{TopK}_{K_R, \tau_r}$ returns at most K_R highest-ranked relations whose similarity exceeds the re-

lation threshold τ_r . Dense and sparse retrieval scores are used only for candidate selection and are not propagated as initial activation values. Independent keyword searches preserve relation-specific semantics that may be diluted when all keywords are represented by a single embedding.

3.2.2 Query-Conditioned Spreading Activation

Starting from the initially retrieved entity set $E^{(0)}$, GRACE-Mem performs query-conditioned spreading activation over the conversational memory graph G_t , based on the propagation formulation used in SA-RAG (Pavlović et al., 2025). Propagation is restricted to at most H hops, and all initially retrieved entities are retained in the expanded candidate set.

For each relation r , its query relevance and rescaled propagation weight are defined as

$$w(r) = [\cos(\mathbf{q}_{\text{vec}}, \mathbf{v}_r)]_+,$$

$$w'(r) = \frac{[w(r) - c]_+}{1 - c}, \quad (6)$$

where $[x]_+ = \max(x, 0)$ and $c \in [0, 1]$ is the relation-relevance cutoff. Relations whose relevance does not exceed c contribute no activation, while the remaining values are rescaled into propagation weights.

The activation value $a(e)$ of each entity is initialized as

$$a(e) = \begin{cases} 1, & e \in E^{(0)}, \\ 0, & \text{otherwise.} \end{cases} \quad (7)$$

Propagation is performed using a FIFO breadth-first queue of entity–depth pairs, initialized with $\{(e, 0) : e \in E^{(0)}\}$. An entity is expanded only when its depth is smaller than H . When an entity n is removed from the queue, each explored relation $r_{n \rightarrow u}$ contributes to the activation of its target entity u according to

$$a(u) \leftarrow \min \{1, a(u) + w' (r_{n \rightarrow u}) a(n)\}. \quad (8)$$

Activation contributions are added as incoming relations are processed. A node may therefore accumulate contributions from multiple predecessors before it is expanded. Each entity is expanded at most once; activation received after expansion updates its final score but is not propagated further. This FIFO procedure avoids repeated propagation through cycles, while clipping activation at 1 prevents runaway score growth.

The resulting entity set is

$$E^{\text{SA}} = E^{(0)} \cup \left\{ e \in \mathcal{N}_t \setminus E^{(0)} \mid a(e) > \tau_a \right\}. \quad (9)$$

where τ_a is the activation threshold. No fixed output-size limit is imposed during spreading activation.

3.3 Compressed-Record Candidate Construction and Two-Stage Reranking

The graph-level candidate set combines the one-hop neighborhood of the expanded entities and retrieved relation endpoints with the directly retrieved relation subgraph:

$$H_q = \text{OneHop} \left(G_t, E^{\text{SA}} \cup \text{Endpoints} \left(R^{(0)} \right) \right) \cup \text{Subgraph} \left(G_t, R^{(0)} \right). \quad (10)$$

The graph-level reranker independently ranks the candidate entities and relations:

$$\begin{aligned} (\hat{E}_q, \hat{R}_q) &= \text{RR}_G(q, H_q; K_G^E, K_G^R), \\ \hat{H}_q &= \hat{E}_q \cup \hat{R}_q. \end{aligned} \quad (11)$$

Here, K_G^E and K_G^R control the maximum numbers of entities and relations retained before compressed-record candidate construction. Each retained entity or relation contains an identifier

linking it to the compressed record from which it was constructed.

We merge compressed records linked to retained graph candidates with those retrieved directly from the compressed-record vector index, which can recover dialogue units missed by imperfect graph construction. Duplicate record identifiers are removed:

$$\begin{aligned} \Sigma_q^G &= \text{Records} \left(\hat{H}_q \right), \\ \Sigma_q^D &= \text{TopK}_{K_D} \left(\mathbf{q}_{\text{vec}}; \mathcal{V}_t^S \right), \\ \Sigma_q &= \text{Unique} \left(\Sigma_q^G \cup \Sigma_q^D \right). \end{aligned} \quad (12)$$

A compressed-record reranker retains at most K provenance-linked candidates:

$$\hat{\Sigma}_q = \text{RR}_S(q, \Sigma_q; K). \quad (13)$$

Graph reranking selects structured records, whereas compressed-record reranking selects their linked records together with direct-retrieval results. The retained records and provenance identifiers seed the Evidence Curation Agent, which uses corpus-access tools to inspect raw evidence only after reranking:

$$\begin{aligned} S_q &= \text{ProvIds} \left(\hat{\Sigma}_q \right), \\ R_0 &= \text{RenderRecords} \left(\hat{\Sigma}_q \right). \end{aligned} \quad (14)$$

3.4 Evidence Curation Agent

Algorithm 1 Agentic Evidence Curation

Require: query q , question date t_q , category d , compressed-record seed IDs S , base compressed context R_0 , corpus $\mathcal{C}$, call cap B , ID cap K , calibration-enabled categories W

Ensure: final evidence context R

Stage I: Agentic Evidence Curation

- 1: $(F, V, ok) \leftarrow \text{AGENTSEARCH}(q, S, \mathcal{C}, B)$
 $\triangleright$ GREP/READ/VEC, then FINAL
- 2: **if** $\neg ok$ **or** $F = \emptyset$ **then**
- 3: **return** R_0 $\triangleright$ identity fallback
- 4: **end if**
- 5: $F \leftarrow (F \cap S) \cup ((F \setminus S) \cap V)$
 $\triangleright$ keep only corpus-verified additions
- 6: **if** $F = \emptyset$ **then**
- 7: **return** R_0 $\triangleright$ identity fallback
- 8: **end if**
- 9: $F \leftarrow \text{CAP}(F, K)$
- 10: record provenance for all IDs in F

Stage II: Evidence Validation

- 11: $P \leftarrow S \setminus F$ $\triangleright$ omitted seeds
 - 12: **if** $d \in W$ **and** $|F| < K$ **and** $P \neq \emptyset$ **then**
 - 13: $A \leftarrow \text{COVERAGECHECK}(q, t_q, P, \mathcal{C})$
 - 14: $F \leftarrow \text{CAP}(F \cup A, K)$
 - 15: **end if**
 - 16: **return** $\text{REBUILD}(F, \mathcal{C})$
-

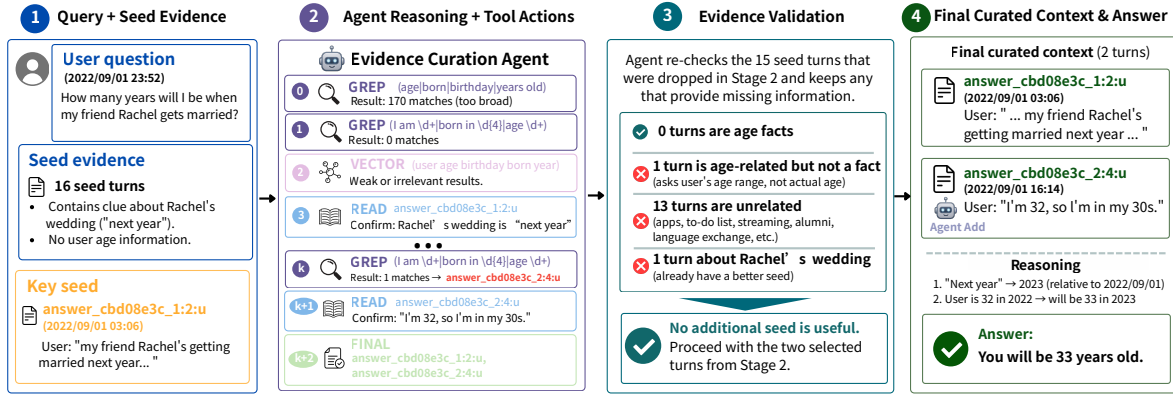


Figure 2: Overview of the Evidence Curation Agent. Given a query and provenance-linked compressed-record seeds, the agent searches the raw corpus through lexical retrieval (GREP), local inspection (READ), and semantic retrieval (VEC); validates evidence coverage; and returns the final curated raw-evidence context.

The Evidence Curation Agent receives a set of provenance-linked compressed-record seed IDs S selected by the upstream retrieval and reranking pipeline. The corresponding compressed records form the initial context available to the agent, while R_0 denotes the complete upstream context rendered from these seeds. The compressed records serve as search anchors rather than as the final evidence supplied to the answering model.

Given the query q , the seed IDs S , and the conversational corpus $\mathcal{C}$, the agent executes AGENTSEARCH with a cap of at most B evidence-access calls. It may use GREP(regex) for lexical or pattern-based corpus search, READ(sid,k) for inspecting raw dialogue around a provenance ID, and VEC(query) for semantic search over the corpus. The search terminates when the agent invokes FINAL(sids) and submits a set of provenance IDs F for downstream answering.

If the agent fails to produce a valid final submission, or if the submitted set is empty, the procedure returns R_0 unchanged. When the call cap is reached before FINAL is emitted, the harness requests a final submission without allowing further corpus access. A missing, unparsable, or empty forced submission is therefore handled by the same identity fallback.

The submitted IDs are then validated against the agent trajectory. Seed IDs may be retained directly because their corresponding records were already provided to the agent. In contrast, a non-seed ID is accepted only if it belongs to V , the set of provenance IDs observed through an evidence-access operation during the current search trajectory. The validated set is therefore

$$F \leftarrow (F \cap S) \cup ((F \setminus S) \cap V).$$

If validation removes every submitted ID, the procedure again returns R_0 . Otherwise, the validated set is capped at K items, and the provenance of each retained ID is recorded.

For question categories included in the calibration set W , the procedure then applies Evidence Validation when unused evidence capacity remains. Let

$$P = S \setminus F$$

denote the upstream seed IDs omitted by the agent. If $d \in W$, $|F| < K$, and $P \neq \emptyset$, an answer-blind COVERAGECHECK examines the omitted seeds using only the query q , the question date t_q , and their associated corpus content. It returns a set A of omitted seed IDs judged potentially useful for answering the query. These IDs are added to the curated set, subject to the same overall cap:

$$F \leftarrow \text{CAP}(F \cup A, K).$$

Validation is strictly additive: it may restore omitted upstream seeds but never removes evidence retained by the agent. It is also restricted to P , and therefore neither performs a new open-ended corpus search nor introduces provenance IDs outside the original seed set S .

Finally, REBUILD($F, \mathcal{C}$) retrieves the raw dialogue spans associated with the resulting provenance IDs and constructs the final evidence context R . Thus, compressed records guide candidate discovery and agentic selection, while the downstream answering model receives provenance-linked raw corpus evidence.

4 Experiments

4.1 Experimental Setup

Benchmarks. We evaluated GRACE-Mem on LoCoMo (1,540 non-adversarial questions spanning single-hop, multi-hop, temporal, and open-domain reasoning) (Maharana et al., 2024) and LongMemEval-S (500 questions spanning recall, preference, temporal reasoning, and knowledge-update categories) (Wu et al., 2025). Throughout this paper, LongMemEval refers to LongMemEval-S unless otherwise specified. Each run used the benchmark’s fixed question set and per-question dialogue corpus. Dataset construction and splits are specified in Appendix B.

Models and implementation. Qwen3-Embedding-0.6B produces dense representations, and Qwen3-Reranker-0.6B performs graph and compressed-record reranking. Main full-pipeline results use GPT-OSS-20B, GPT-OSS-120B and GPT-4o-mini as answer backbones. Full configuration details are provided in Appendix C.

Experimental protocol. Full-pipeline main results and component ablations rerun retrieval and reranking from each benchmark’s dialogue corpus. Agent-layer experiments instead reuse a saved retrieved context as a fixed retrieval input while rerunning the corpus agent over that context. This setup isolates the effect of evidence curation from variation in upstream retrieval and reranking.

Evaluation. We report answer accuracy as the primary metric, using GPT-4o-mini as the evaluation judge. Comparisons within the same experiment fix the questions, corpus, answer prompt, and judge; results from systems using different backbones or evaluation harnesses are treated as contextual rather than paired comparisons.

4.2 Main Results

GRACE-Mem achieves 84.6, 86.9, and 85.3 on LoCoMo with OSS-20B, OSS-120B and GPT-4o-mini, respectively, and 77.6, 80.2, and 73.2 on LongMemEval (Table 1). On LoCoMo, all three GRACE-Mem configurations outperform the reported memory-system baselines, with OSS-120B reaching the strongest overall score. On LongMemEval, GRACE-Mem likewise improves substantially over the reported memory baselines. Because published systems differ in answer backbones, judges, and evaluation harnesses, these

Table 1: Overall LLM-as-Judge accuracy (%) on LoCoMo and LongMemEval. Baselines are listed in chronological order. Results for LangMem, A-MEM, MemGPT, Mem0, Zep and AMA are reported from cited AMA. Results for Nemori are reported from (Ma et al., 2026).

Model / Baseline	LoCoMo Overall	LongMemEval Overall
<i>Published memory systems (Backbone: GPT-4o-mini)</i>		
RAG (Lewis et al., 2020)	30.2	37.4
MemGPT (Packer et al., 2023)	41.5	55.4
Zep (Rasmussen et al., 2025)	58.5	63.2
A-MEM (Xu et al., 2025)	52.5	53.8
Mem0 (Chhikara et al., 2025)	65.3	57.4
Nemori (Ma et al., 2026)	73.0	64.2
AMA (Huang et al., 2026)	77.4	69.8
LangMem (LangChain, 2026)	51.5	52.8
<i>GRACE-Mem</i>		
OSS-20B	84.6	77.6
OSS-120B	86.9	80.2
GPT-4o-mini	85.3	73.2

cross-system results are interpreted as contextual comparisons; the controlled analyses in the following sections are used to isolate the contributions of individual GRACE-Mem components. Notably, the subsequent gold-evidence analysis (Appendix A.2) shows that GRACE-Mem already approaches the empirical gold-evidence performance observed on LoCoMo, suggesting that its strong performance is accompanied by relatively little remaining retrieval-side headroom. Complete per-category benchmark results are reported in Appendix A.

4.3 Effect of the Agent Curation of Evidence

The effects of the agentic evidence layer vary across answer backbones and benchmarks (Table 2). On LongMemEval, filter-fetch improves all three models, and evidence validation provides further gains. The largest improvement occurs for OSS-120B, which increases from 76.0 to 80.2, followed by OSS-20B from 76.2 to 77.6. LoCoMo shows a more mixed pattern. Evidence validation improves OSS-20B from 84.0 to 84.6 and GPT-4o-mini from 84.2 to 85.3. For OSS-120B, however, the 87.3 baseline remains stronger than both evidence-layer variants, suggesting that filtering can remove useful context when the original retrieval already provides sufficient evidence. Overall, evidence validation is more reliable than filter-fetch alone, but its benefit remains model- and benchmark-dependent. Full results for all answer backbones are reported in Appendix A.3.

Table 2: Accuracy (%) of the main evidence-layer configurations.

Configuration	LongMemEval			LoCoMo		
	OSS-20B	OSS-120B	GPT-4o-mini	OSS-20B	OSS-120B	GPT-4o-mini
Baseline	76.2	76.0	69.2	84.0	87.3	84.2
filter-fetch	77.2	79.0	71.8	82.3	86.1	83.6
filter-fetch + evidence validation	77.6	80.2	73.2	84.6	86.9	85.3

4.4 Targeted Component Ablations

All ablations are conducted on the retrieval pipeline alone. Removing graph retrieval reduces accuracy by 17.6 points on LoCoMo and 8.3 points on LongMemEval, confirming its importance for connected evidence discovery. Removing direct compressed-record retrieval causes smaller but consistent drops of 3.8 and 6.5 points, showing that it complements graph retrieval by recovering candidates missed by the structured path. Using compressed records as final evidence further reduces performance by 7.8 and 6.2 points, indicating that compressed representations are useful for retrieval but cannot fully replace source dialogue at answer time.

On LoCoMo, removing graph retrieval reduces multi-hop accuracy from 81.6 to 57.4 and temporal accuracy from 74.5 to 48.3, suggesting that graph structure is especially important when evidence must be connected across turns or events. Using compressed records alone also lowers temporal accuracy to 50.6 on LoCoMo and 59.8 on LongMemEval. Meanwhile, removing direct retrieval reduces LongMemEval knowledge-update accuracy from 71.8 to 57.7, indicating that direct retrieval provides an important recovery path when graph construction does not capture all relevant evidence.

Overall, these results support the role-separated design of GRACE-Mem: graph retrieval provides structured candidate discovery, direct retrieval improves evidence coverage, and the original dialogue preserves the source-level details required for accurate answering.

5 Discussion

Our results suggest that retrieval representations and answer evidence should be treated as distinct components. Graph structure, entity-relation information, and compressed dialogue records are useful for discovering and linking candidate evidence, but replacing provenance-linked source dialogue with compressed representations leads to substan-

tial performance degradation on both benchmarks. This indicates that structured memory is better used as an index to the original dialogue rather than as a substitute for source evidence.

The post-retrieval agent is better viewed as a corrective mechanism than as a mandatory stage. On LongMemEval, it helps recover evidence missed by graph retrieval, whereas on LoCoMo, additional filtering can remove useful context when the initial retrieval is already sufficient. The gold-evidence analysis shows a similar pattern: LoCoMo is close to retrieval saturation, while LongMemEval retains a larger evidence gap. These results suggest that agentic curation should be applied adaptively according to retrieval sufficiency, query complexity, and the capability of the downstream answer model.

More broadly, our findings highlight the distinction between evidence acquisition and evidence utilization. When the required evidence is already present, retrieving additional context does not necessarily improve final accuracy; residual errors may instead arise from temporal reasoning, cross-turn aggregation, or conflict resolution. Conversely, substantial gaps from gold evidence indicate remaining retrieval-side headroom. Long-term memory systems should therefore be evaluated not only by end-to-end accuracy, but also with evidence-level diagnostics that distinguish failures caused by missing evidence from those caused by ineffective use of available evidence.

6 Conclusion

GRACE-Mem treats traceable dialogue evidence as the memory primitive and uses a hybrid graph retriever plus a corpus agent to deliver it. The key empirical question is not whether more memory is retrieved, but whether the right evidence survives ranking, graph expansion, and tool-mediated context construction. The required ablation matrix makes those stages independently testable. On both benchmarks it stays competitive while keeping answers traceable to source turns.

Limitations

GRACE-Mem leaves several directions for further improvement. First, the effectiveness of evidence curation varies across answer models and benchmarks. When the initial retrieval already provides sufficient evidence, additional filtering or validation may remove useful context and slightly reduce accuracy. This suggests that a fixed agentic procedure is not optimal for every query. Future work could therefore develop adaptive policies that determine whether corpus search, filtering, or evidence validation is needed based on retrieval confidence, evidence sufficiency, and downstream model behavior.

Second, the current ADD/UPDATE mechanism does not explicitly preserve temporal versions of changing information. Extending the memory graph with temporally scoped facts and relations could better distinguish historical from current states and improve knowledge-update and temporal-reasoning queries.

Finally, GRACE-Mem currently applies the same retrieval and curation policy across queries. Dynamically adjusting retrieval depth, graph expansion, and corpus inspection according to query complexity and retrieval uncertainty could improve coverage for sparse or multi-hop questions while preserving sufficient context for simpler cases.

References

- Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. 2025. [Mem0: Building production-ready AI agents with scalable long-term memory](#). *arXiv preprint arXiv:2504.19413*.
- Jesper Derehag, Carlos Calva, and Timmy Ghiurau. 2026. [SmartSearch: How ranking beats structure for conversational memory retrieval](#). *arXiv preprint arXiv:2603.15599*.
- Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2024. [LightRAG: Simple and fast retrieval-augmented generation](#). *arXiv preprint arXiv:2410.05779*.
- Weiwan Huang, Zixuan Wang, Hehai Lin, Sudong Wang, Bo Xu, Qian Li, Beier Zhu, Linyi Yang, and Chengwei Qin. 2026. [AMA: Adaptive memory via multi-agent collaboration](#). In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 3099–3120, San Diego, California, United States. Association for Computational Linguistics.
- Shuo Ji, Yibo Li, and Bryan Hooi. 2026. [Memory is re-constructed, not retrieved: Graph memory for LLM agents](#). *arXiv preprint arXiv:2606.06036*.
- LangChain. 2026. [LangMem](#). Documentation page. Accessed July 28, 2026.
- Chris Latimer, Nicol   Boschi, Andrew Neeser, Chris Bartholomew, Gaurav Srivastava, Xuan Wang, and Naren Ramakrishnan. 2025. [Hindsight is 20/20: Building agent memory that retains, recalls, and reflects](#). *arXiv preprint arXiv:2512.12818*.
- Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich K  ttler, Mike Lewis, Wen-tau Yih, Tim Rock-t  schel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474.
- Zhuofeng Li, Haoxiang Zhang, Cong Wei, Pan Lu, Ping Nie, Yi Lu, Yuyang Bai, Shangbin Feng, Hangxiao Zhu, Ming Zhong, Yuyu Zhang, Jianwen Xie, Yejin Choi, James Zou, Jiawei Han, Wenhui Chen, Jimmy Lin, Dongfu Jiang, and Yu Zhang. 2026. [Beyond semantic similarity: Rethinking retrieval for agentic search via direct corpus interaction](#). *arXiv preprint arXiv:2605.05242*.
- Wenquan Ma, Jiayan Nan, and WenLong Wu. 2026. [What deserves memory: Adaptive memory distillation for LLM agents](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 34789–34812, San Diego, California, United States. Association for Computational Linguistics.
- Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. 2024. [Evaluating very long-term conversational memory of LLM agents](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13851–13870, Bangkok, Thailand. Association for Computational Linguistics.
- Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. 2023. [MemGPT: Towards LLMs as operating systems](#). *arXiv preprint arXiv:2310.08560*.
- Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Xufang Luo, Hao Cheng, Dongsheng Li, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Jianfeng Gao. 2025. SeCom: On memory construction and retrieval for personalized conversational agents. In *The Thirteenth International Conference on Learning Representations*.
- Jovan Pavlovi  , Mikl  s Kr  sz, and L  szl   Hajdu. 2025. [Leveraging spreading activation for improved document retrieval in knowledge-graph-based RAG systems](#). *arXiv preprint arXiv:2512.15922*.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. 2025. [Zep: A temporal knowledge graph architecture for agent memory](#). *arXiv preprint arXiv:2501.13956*.

Sahil Sen, Akhil Kasturi, Elias Lumer, Anmol Gulati, and Vamse Kumar Subbiah. 2026. [Is grep all you need? how agent harnesses reshape agentic search](#). *arXiv preprint arXiv:2605.15184*.

Orion Weller, Michael Boratko, Iftexhar Naim, and Jinhyuk Lee. 2025. [On the theoretical limitations of embedding-based retrieval](#). *arXiv preprint arXiv:2508.21038*.

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

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

A Extended Experimental Analysis

A.1 Per-Category Main Results

Table 3 shows that OSS-120B achieves the strongest overall LoCoMo result and leads the other GRACE-Mem answer backbones across all reported categories. The gains are nevertheless uneven, with larger differences on Multi-Hop, Temporal, and Open Domain questions than on Single-Hop questions. This suggests that the same retrieved evidence can support different levels of downstream reasoning depending on the answer backbone, rather than model scaling acting as a uniform improvement.

Table 4 shows a more model-dependent pattern on LongMemEval. OSS-120B performs best overall and leads on preference, assistant-history, temporal-reasoning, and knowledge-update questions, whereas OSS-20B remains stronger on multi-session questions. GPT-4o-mini is competitive on single-session user and assistant-history questions but weaker on categories requiring evidence integration or state updates. These results reinforce that the effect of the answer backbone is category-specific rather than uniformly positive.

A.2 Gold-Evidence Analysis across Benchmarks

We further examine the remaining performance headroom when the answer model is provided with benchmark-specific gold evidence instead of

evidence retrieved by GRACE-Mem. Rather than treating this setting as a strict upper bound, we use it as an empirical reference for assessing how closely the retrieved context approaches an evidence-sufficient condition and for distinguishing limitations in evidence acquisition from those that remain after relevant evidence has already been supplied.

The two benchmarks require different gold-context constructions. LoCoMo annotations identify individual dialogue turns, which may omit nearby context needed to resolve pronouns, relative temporal expressions, or references to surrounding events. Using only the annotated turns produced 169 additional errors relative to GRACE-Mem, of which 73% were Unknown. We therefore augment each annotated evidence turn with up to two preceding and two following turns. This context expansion improves accuracy by 3.1, 3.2, and 2.6 percentage points for OSS-20B, OSS-120B, and GPT-4o-mini, respectively, allowing the expanded gold-evidence setting to outperform the corresponding unexpanded gold-evidence baseline. LongMemEval, in contrast, provides the complete conversation sessions containing the annotated evidence, so we use the official Oracle context without further expansion. These settings should therefore be interpreted as benchmark-specific gold-evidence conditions rather than directly comparable Oracle contexts.

Table 5 compares GRACE-Mem with these gold-evidence settings. We define

$$\Delta_{\text{gold}} = \text{Acc}_{\text{gold}} - \text{Acc}_{\text{GRACE-Mem}}. \quad (15)$$

On LoCoMo, gold evidence does not consistently improve performance over GRACE-Mem. OSS-20B and OSS-120B score 4.0 and 4.4 points lower with gold evidence, respectively, while GPT-4o-mini gains 0.2 points. This suggests that the expanded gold-turn context is not a strict upper bound for every answer model: additional annotated context may introduce redundancy or alter how evidence is interpreted. Overall, GRACE-Mem already performs competitively with the benchmark-specific gold-evidence setting, and the remaining errors cannot be attributed to retrieval alone.

LongMemEval exhibits a different pattern. Replacing the retrieved context with the official Oracle context improves accuracy by 5.7 points for OSS-20B, 10.4 points for OSS-120B, and

Table 3: LoCoMo per-category accuracy (%) against memory-system baselines.

Model / Baseline	Overall	Single-hop	Multi-hop	Temporal	Open Domain
<i>Published memory systems</i>					
RAG	30.2	32.0	31.3	23.7	32.6
MemGPT	41.5	47.9	22.9	38.8	50.0
Zep	58.5	63.2	50.5	58.9	39.6
A-MEM	52.5	58.2	43.6	54.2	22.9
Mem0	65.3	67.7	62.4	66.0	50.0
Nemori	73.0	81.9	61.7	67.6	45.8
AMA	77.4	84.9	68.1	74.8	47.9
LangMem	51.5	61.6	52.6	25.0	47.7
<i>GRACE-Mem</i>					
OSS-20B	84.6	88.4	84.0	77.3	63.5
OSS-120B	86.9	92.3	86.2	79.8	68.8
GPT-4o-mini	85.3	91.8	84.4	71.3	64.6

Table 4: LongMemEval per-category accuracy (%) against memory-system baselines.

Model / Baseline	Overall	single-session-user	multi-session	single-session-preference	single-session-assistant	temporal-reasoning	knowledge-update
<i>Published memory systems</i>							
RAG	37.4	68.6	25.4	33.3	71.4	28.0	38.5
MemGPT	55.4	81.4	54.9	20.0	85.7	45.1	41.0
Zep	63.2	92.9	47.4	53.3	75.0	54.1	74.4
A-MEM	53.8	75.0	45.1	36.7	80.4	39.8	60.2
Mem0	57.4	85.7	48.1	33.3	87.5	39.9	65.4
Nemori	64.2	88.6	51.1	46.7	83.9	61.7	61.5
AMA	69.8	98.6	62.4	46.7	96.4	44.4	89.7
LangMem	52.8	75.0	42.4	26.7	77.7	35.3	57.8
<i>GRACE-Mem</i>							
OSS-20B	77.6	88.6	68.4	73.3	94.6	80.3	65.4
OSS-120B	80.2	98.4	66.9	79.2	96.0	82.2	81.9
GPT-4o-mini	73.2	97.1	58.5	51.7	94.2	72.8	73.1

4.8 points for GPT-4o-mini. The substantially larger gaps indicate that evidence acquisition and selection remain important bottlenecks on this benchmark, particularly for OSS-120B. In other words, LongMemEval retains considerably more retrieval-side headroom than LoCoMo.

Taken together, these results reveal different performance regimes across the two benchmarks. GRACE-Mem approaches an evidence-sufficient regime on LoCoMo, where retrieval is nearing saturation and downstream evidence utilization becomes an increasingly important source of residual error. LongMemEval, by contrast, still offers substantial opportunity to improve which evidence is recovered and delivered to the answer model. Because gold annotations may be incomplete and the two benchmarks require different context-construction procedures, these differences

should be interpreted as empirical gold-evidence gaps rather than a strict decomposition of retrieval and answering errors.

A.3 Full Agentic Evidence-Layer Results

Table 6 reports the complete results across all five answer backbones, and Figures 3 and 4 visualize the three primary backbones (OSS-20B, OSS-120B, GPT-4o-mini) on LongMemEval and LoCoMo, respectively. The broader comparison confirms that evidence validation is generally more reliable than filter-fetch alone, but the magnitude of the gain varies by model and benchmark. On LongMemEval, every primary backbone improves monotonically from Baseline to filter-fetch to evidence validation (Figure 3), whereas on LoCoMo the OSS-120B baseline remains the strongest configuration (Figure 4). Among the additional back-

Table 5: GRACE-Mem versus benchmark-specific gold-evidence answering.

Benchmark	Answer model	GRACE-Mem	Gold evidence	Δ_{gold}
LoCoMo	OSS-20B	84.6	80.6	-4.0
LoCoMo	OSS-120B	86.9	82.5	-4.4
LoCoMo	GPT-4o-mini	85.3	85.5	+0.2
LongMemEval	OSS-20B	77.6	83.3	+5.7
LongMemEval	OSS-120B	80.2	90.6	+10.4
LongMemEval	GPT-4o-mini	73.2	78.0	+4.8

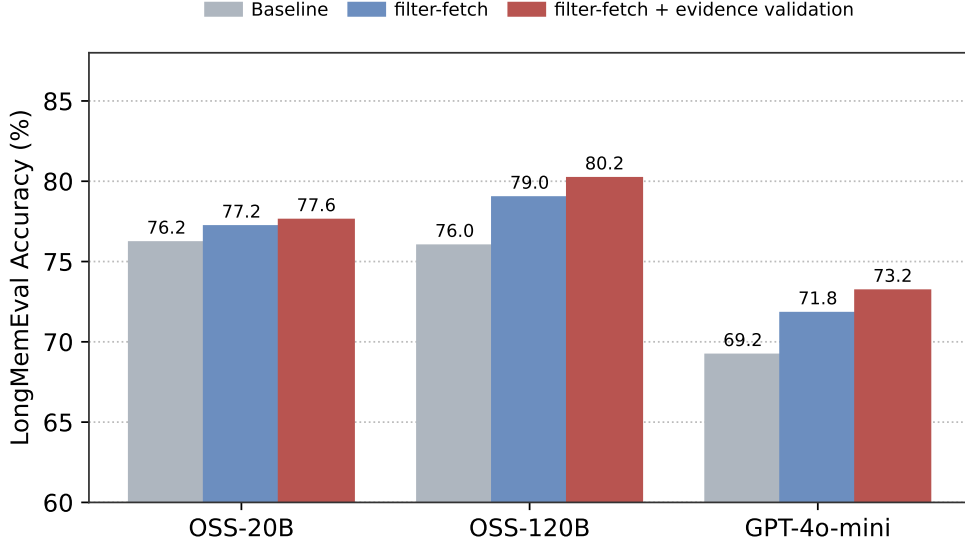


Figure 3: LongMemEval accuracy of the three primary answer backbones (OSS-20B, OSS-120B, GPT-4o-mini) under the three evidence-layer configurations (Baseline, filter-fetch, and filter-fetch + evidence validation). Every backbone improves monotonically as the agent stages are added, with the largest gain on OSS-120B (76.0 $\rightarrow$ 80.2).

bones, Qwen-4B benefits little from validation, whereas Qwen-27B shows larger improvements, especially on LongMemEval and LoCoMo.

B Benchmark Details

LoCoMo. Each conversation spans ~ 600 dialogue turns averaging $\sim 16,000$ tokens. The Adversarial category is excluded (ground-truth answers unavailable).

LongMemEval. Each question is paired with ~ 40 history sessions totaling $\sim 115,000$ tokens. The benchmark evaluates information extraction, multi-session reasoning, temporal reasoning, knowledge updates, and abstention.

C Reproducibility Details

This appendix summarizes the implementation contracts, system settings, and reproduction procedure. Complete prompt templates are provided separately in Appendix G.

C.1 Temporal Normalization and Evidence Contract

Before entity and relation extraction, relative temporal expressions are resolved against the dialogue timestamp. The normalized input may contain temporal markers such as [DATE: ...], [TIME: ...], and [TIMESPAN: ...]. Their payloads are preserved throughout downstream processing and are not recomputed or rewritten.

Each ingestion unit retains the following provenance mapping:

```
summary_id = session_id | message_id
evidence[i].summary_id = summary_id
evidence[i].span = sentence span in the
                    rewritten raw dialogue
```

The provenance anchor is generated outside the LLM and therefore remains a storage-layer invariant rather than a value inferred from generated text.

After parsing and validation, model outputs are converted into the normalized records listed in Table 7, which separates retrieval-oriented records

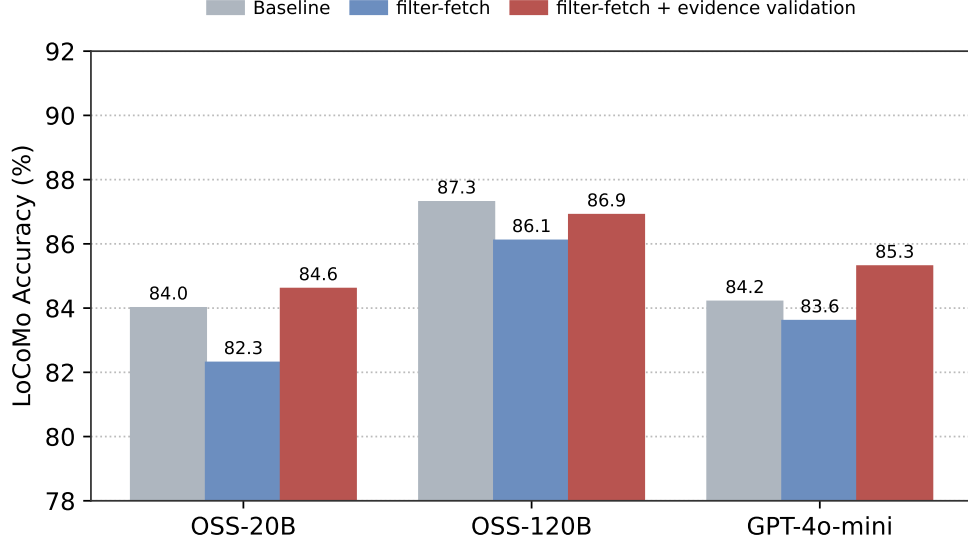


Figure 4: LoCoMo accuracy of the same three primary answer backbones under the same evidence-layer configurations. In contrast to LongMemEval, the OSS-120B baseline (87.3) remains the strongest configuration, indicating that filtering can remove useful context when the original retrieval already provides sufficient evidence.

Table 6: Full accuracy results for agentic evidence-layer configurations.

Configuration	LongMemEval					LoCoMo				
	OSS-20B	OSS-120B	Qwen-4B	Qwen-27B	GPT-4o-mini	OSS-20B	OSS-120B	Qwen-4B	Qwen-27B	GPT-4o-mini
Baseline	76.2	76.0	73.6	80.6	69.2	84.0	87.3	84.2	84.2	84.2
filter-fetch	77.2	79.0	74.2	81.6	71.8	82.3	86.1	80.3	85.7	83.6
filter-fetch + evidence validation	77.6	80.2	73.5	81.8	73.2	84.6	86.9	83.5	86.8	85.3

Table 7: Normalized memory records and their authoritative evidence.

Record	Required fields
Entity	id, name, type, desc, keyword, summary_id
Relation	src_name, src_type, target_name, target_type, desc, keyword, summary_id
Evidence	session/message anchor and raw sentence span as defined by the provenance mapping above, plus turn meta-data

from final answer evidence. Entity and relation records help locate information, while evidence records retain raw dialogue spans and provenance anchors so that the delivered context remains grounded in the original conversation.

C.2 Execution Stages and Invariants

The system performs temporal normalization, entity extraction, relation extraction, entity reconciliation, hybrid retrieval, graph expansion, evidence filtering, and answer generation. Table 8 defines this pipeline as a sequence of stages with explicit inputs, outputs, and constraints. These contracts preserve provenance, bound graph expansion,

and fall back to the upstream context when evidence curation fails, making the evidence path inspectable.

C.3 Default Settings

Table 9 summarizes the main experimental configuration used for all reported results unless stated otherwise.

Frozen retrieval artifacts for Qwen evaluations. Qwen is used only as the answer model. The Qwen evaluations reuse frozen, benchmark-specific GRACE-Mem retrieval artifacts and do not re-ingest the corpus or rebuild the memory graph.

C.4 Reproduction Procedure

To reproduce the experiments, fix the dataset split and the per-question corpus, and ingest each session using the same temporal reference time. Then persist the graph and vector artifacts, run hybrid retrieval and reranking, enable the agent filter with the reported configuration, and generate answers using the benchmark-specific prompt in Appendix G.

Table 8: Execution stages, inputs, outputs, and invariants.

Stage	Input	Output	Invariant
Normalize	raw dialogue, reference time	marked temporal spans	marker payload unchanged
Extract	normalized turns	evidence, entities, relations	entities precede relations
Resolve	new records, graph candidates	Add/Update decisions	endpoints belong to entity set; UPDATE targets are valid graph candidates
Retrieve	query, graph, indexes	top- K compressed-record seed IDs	provenance IDs retained
Expand	seeds, weighted graph	SA-RAG activation subgraph	hop limit respected
Filter	compressed-record seeds, raw corpus, agent tools	final raw evidence IDs	identity fallback on failure
Answer	question, evidence context	English answer	no unsupported final claim

Table 9: Default system configuration.

Setting	Value
Record compression	LLMLingua-2, 60%
Embedding model	Qwen3-Embedding-0.6B
Graph database	FalkorDB
Retrieval	BM25+entity+relation, union
Activation hops h	2
Activation edge cutoff c	0.4
Activation threshold τ_a	0.5
Compressed-record rerank pool	top-16
Reranker	Qwen3-Reranker-0.6B
Retrieved seed cap K	16 provenance IDs
Agent tool-call budget B	10 calls
Agent output cap	16 provenance IDs
Pair inclusion	enabled
Fallback	complete seed context

Each experiment should report the answer model, agent model, judge model, retrieval unit, and agent mode, as these settings materially affect both answer accuracy and evidence retrieval performance.

D Reranking Instructions

Reranking scores each candidate using the difference between the yes and no logits or log probabilities. Two instructions are defined, one for entity candidates and one for relationship candidates. Compressed-record reranking reuses the entity instruction.

Shared reranking prompt

SYSTEM:
Judge whether the Document meets the requirements based on the Query and the Instruct provided. Note that the answer can only be "yes" or "no".

USER:
<Instruct>: {instruction}

<Query>: {query}

<Document>: {document}

Entity reranking instruction

You are reranking candidate entities for long-term memory question answering. Judge whether the candidate entity is useful for retrieving evidence needed to answer the query. A relevant entity does not need to contain the final answer directly, but it should help locate, connect, or justify the answer-bearing memory.

First infer the query's retrieval need (temporal, preference, multi-hop, or knowledge-update). Then apply the matching rule below:

- Temporal: prefer entities that represent dates, time ranges, recency, event chronology, or before/after order relevant to the query.
- Preference: prefer entities about stable likes, dislikes, favorites, habits, goals, choices, constraints, or personal attributes.
- Multi-hop: prefer bridge entities that connect the query subject to the answer, even if the entity does not directly state the final answer.
- Knowledge-update: prefer entities pointing to newer, corrected, or currently valid memories; penalize stale entities when the query asks for the latest or updated fact.

Mark as relevant only if the entity can help retrieve or justify the answer. Mark as not relevant if it is only broadly related to the topic, shares a keyword, or is a generic background concept with no direct link to the evidence.

Example -- Query: "When did Audrey make muffins for herself?"
Relevant: "Week of April 3, 2023" -- anchors the answer time.
Not relevant: "Pastries" -- generic topic entity with no direct link to the muffin-making event or its date.

Relationship reranking instruction

You are reranking candidate relationships for long-term memory question answering. Judge whether the candidate relationship expresses a fact, event, temporal link, preference, update, or bridge connection useful for answering the query. A relevant relationship may directly state the answer or provide a necessary reasoning step toward it.

First infer the query's retrieval need (temporal, preference, multi-hop, or knowledge-update). Then apply the matching rule below:

- Temporal: prefer relationships that state when something happened, link an event to a date, describe before/after order, recency, duration, or event chronology.
- Preference: prefer relationships that state likes, dislikes, favorites, habits, goals, choices, constraints, or stable personal tendencies.
- Multi-hop: prefer relationships that connect two useful entities in the reasoning chain (subject-event, event-date, person-preference, or bridge relations).
- Knowledge-update: prefer relationships that indicate newer, corrected, replaced, or currently valid information; penalize stale or superseded facts when the query asks for the latest state.

Mark as relevant only if the relationship helps answer, retrieve evidence for, or justify the answer. Mark as not relevant if it only shares broad topic words or expresses a generic association that does not narrow down the answer.

Example -- Query: "When did Audrey make muffins for herself?"

Relevant: a relationship linking the muffin/pastry event to "Week of April 3, 2023" -- supports the temporal answer.

Not relevant: a relationship about pastries in general with no connection to the specific event or its date.

Instruction-free control prompt. An earlier version of the reranker scored every candidate with a single generic template that asked only whether the document was useful, without a task instruction and without distinguishing candidate types:

Generic reranking control prompt

Query: {query}

Document: {document}

Question: Is the document useful to answer the query?
Answer "Yes" or "No".

From a generic scorer to a memory-evidence scorer. The instruction-free control template leaves *useful* undefined for long-term memory question answering over a knowledge graph, so the reranker tends to conflate topic-level relatedness with answer-bearing relevance: a candidate that merely shares a person, an event, or a keyword with the query can still score highly even when it cannot support the answer. The instruction-conditioned prompt restores that format and adds a task instruction, split by candidate type, that first infers the query's retrieval need (temporal, preference, multi-hop, or knowledge update) and then prefers candidates pointing to the target person, event, time, preference, attribute, or multi-hop bridge relation over candidates that only share broad topic words. The design intent is to steer the

Table 10: Generic reranking control prompt vs. instruction-conditioned reranking prompt on LongMemEval. The instruction-conditioned prompt nearly doubles gold-evidence coverage (recall) and lifts overall accuracy.

Metric / Category	Generic	Instruction	Δ
<i>Retrieval vs. gold evidence</i>			
Gold coverage (recall)	34.5	63.4	+28.9
All-gold-hit rate	35.3	59.9	+24.6
<i>Answer accuracy</i>			
Overall	70.7	73.7	+3.0

reranker toward answer-bearing evidence rather than topically related distractors; as the measurements below show, its dominant measured effect is a large gain in gold-evidence coverage.

Retrieval and end-to-end effect. Table 10 compares a run using the generic reranking control prompt with one using the instruction-conditioned prompt on LongMemEval, reporting gold-evidence coverage and downstream answer accuracy. The instruction-conditioned prompt substantially improves retrieval coverage: gold-evidence recall rises from 34.5% to 63.4%, an increase of 28.9 points. This improvement also leads to better end-to-end performance, with overall answer accuracy increasing by 3.0 points, from 70.7% to 73.7%. These results indicate that the instruction-conditioned prompt improves downstream accuracy primarily by recovering a larger fraction of the answer-bearing evidence. The numbers reflect this controlled comparison and are not the headline system configuration reported in the main results tables.

E Component Ablation Results

We conduct component ablations on the retrieval pipeline before the agentic evidence-curation layer. The reference configuration retains all graph-retrieval, memory-representation, reranking, and temporal-processing components, but excludes both the Evidence Curation Agent and Evidence Validation. We refer to this configuration as the full retrieval pipeline. Complete category-level results are reported in Section E.4.

E.1 Graph-Structured Retrieval

This group evaluates the contribution of the graph retrieval channel and its main internal components. In the full retrieval pipeline, GRACE-Mem retrieves entity and relation candidates using dense

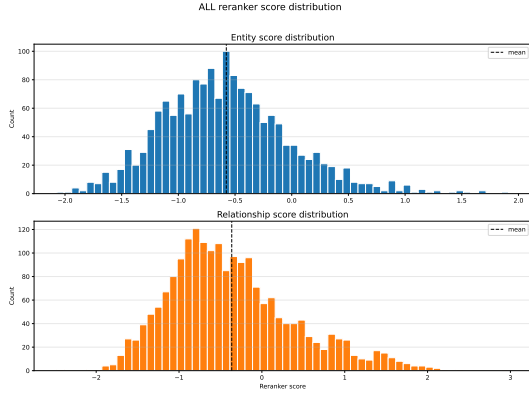


Figure 5: Reranker score distribution under the Generic reranking control prompt, shown separately for entity candidates (top) and relationship candidates (bottom). The dashed line marks the mean. Scores collapse into a narrow unimodal band spanning roughly two score units, leaving little margin between retained and discarded candidates.

and lexical signals, expands the retrieved entities through spreading activation, constructs a graph candidate set, and reranks its entities and relations before collecting their linked compressed records. These graph-linked records are subsequently merged with records obtained through direct dense retrieval.

We construct four ablation variants to isolate the role of each stage. **No graph retrieval** removes the entire graph channel, including entity and relation retrieval, spreading activation, graph candidate construction, and graph-element reranking, leaving only direct dense retrieval over the compressed-record index. **No lexical entity retrieval** disables the BM25-based entity retrieval branch while retaining dense entity retrieval and the remaining graph pipeline, testing whether exact lexical matching provides information beyond semantic similarity. **No spreading activation** retains the initially retrieved graph candidates but disables propagation-based multi-hop expansion, isolating the contribution of graph connectivity to recovering indirectly related evidence. **No graph-level reranking** removes the graph-level entity and relation reranker before compressed-record candidate construction, testing whether the initially retrieved and expanded graph candidates require an additional relevance-selection stage.

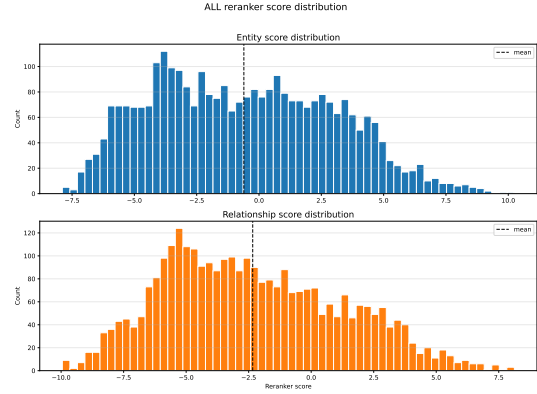


Figure 6: Reranker score distribution under the instruction-conditioned prompt, for the same query batch and the same candidate types as Figure 5, and plotted on the same vertical scale. The score range widens by roughly an order of magnitude (to about ± 10), giving a much larger margin between rejected and endorsed candidates.

Table 11: Graph-structured retrieval ablations on LoCoMo and LongMemEval with OSS-20B.

Configuration	LoCoMo	LongMemEval
Full retrieval pipeline	84.0	76.2
No graph retrieval	66.4 (−17.6)	67.9 (−8.3)
No lexical entity retrieval	80.2 (−3.8)	72.2 (−4.0)
No spreading activation	81.2 (−2.8)	73.0 (−3.2)
No graph-level reranking	81.1 (−2.9)	71.3 (−4.8)

Among the graph-side ablations, removing the entire graph channel produces the largest overall degradation on both benchmarks, reducing OSS-20B accuracy by 8.3 points on LongMemEval and 17.6 points on LoCoMo. The larger LoCoMo drop is most visible in multi-hop and temporal questions, whose accuracy falls from 81.6 to 57.4 and from 74.5 to 48.3, respectively. Removing lexical entity retrieval, spreading activation, or graph-element reranking also lowers accuracy on both benchmarks, with drops ranging from 2.8 to 4.8 points.

E.2 Memory and Evidence Representation

This group examines how memory units are constructed, how candidate records are retrieved, and which representation is ultimately delivered to the answer model. The full retrieval pipeline uses role-separated dialogue units during memory construction, combines graph-linked candidates with direct dense retrieval over the compressed-record index, and reconstructs the final answering context from provenance-linked raw dialogue spans.

We construct three ablation variants to isolate these design choices. **Unsplit turn-pair units** re-

places the role-separated dialogue units used by the full pipeline with a single combined unit containing both sides of each turn pair. This variant tests whether separating speaker-specific content into finer-grained memory units improves retrieval precision and reduces interference from unrelated information within the paired turn. Because this unit construction was evaluated only for LongMemEval, no corresponding LoCoMo result is reported.

No direct compressed-record retrieval removes the dense retrieval path over the compressed-record vector index while retaining graph-based retrieval and the compressed records linked to the selected graph candidates. This variant tests whether direct record-level retrieval complements graph construction by recovering relevant dialogue units that are omitted because of imperfect entity extraction, relation extraction, or graph connectivity.

Compressed records only retains the complete retrieval and reranking pipeline but supplies the selected compressed records directly to the answer model instead of reconstructing the final context from their provenance-linked raw dialogue spans. This variant isolates the contribution of source-level dialogue evidence and tests whether compressed representations alone preserve the speaker, temporal, and contextual details required for answering.

Table 12: Memory and evidence representation ablations on LoCoMo and LongMemEval with OSS-20B.

Configuration	LoCoMo	LongMemEval
Full retrieval pipeline	84.0	76.2
Unsplit turn-pair units	—	69.5 (−6.7)
No direct compressed-record retrieval	80.2 (−3.8)	69.7 (−6.5)
Compressed records only	76.2 (−7.8)	70.0 (−6.2)

On LongMemEval, replacing role-separated evidence units with unsplit turn pairs reduces accuracy by 6.7 points, indicating that finer-grained evidence units support more effective retrieval. We report this ablation only on LongMemEval because LoCoMo uses dialogue-turn units rather than the role-separated units used in LongMemEval, making the same comparison not directly comparable. Direct compressed-record retrieval also contributes across both benchmarks: removing this path lowers accuracy by 6.5 points on LongMemEval and 3.8 points on LoCoMo. Restricting the final context to compressed records causes the largest degradation in this group, reducing ac-

curacy by 6.2 points on LongMemEval and 7.8 points on LoCoMo.

As shown in Table 14 and 15, the decline is particularly pronounced for LongMemEval temporal questions and for LoCoMo multi-hop questions. These results show that compressed-record retrieval is useful for candidate discovery, but compressed-only context does not preserve sufficient detail for temporal and multi-hop reasoning.

E.3 Retrieval and Temporal Design Choices

This group evaluates three design choices within the retrieval pipeline: query-side keyword extraction, temporal normalization of dialogue content, and compressed-record reranking. The main graph retrieval and evidence representation architecture is retained in all variants.

We construct three ablation variants to isolate the contribution of each component. **No keyword extraction** disables query-side keyword generation. Entity retrieval therefore relies on the complete query embedding without keyword-based lexical entity matching, while relation retrieval no longer receives the fine-grained keyword embeddings used to identify relation-specific cues. This variant tests whether extracted keywords provide retrieval signals that are complementary to the full-query representation.

No temporal normalization disables the normalization of relative temporal expressions in dialogue records while leaving the remaining ingestion and retrieval pipeline unchanged. Expressions such as “yesterday,” “last month,” or “two weeks ago” are therefore retained in their original form rather than being grounded to the corresponding dialogue date. This variant tests whether temporally normalized memory representations improve the retrieval and interpretation of time-sensitive evidence across sessions.

No compressed-record reranking removes the final reranking stage over the merged pool of graph-linked and directly retrieved compressed records. Instead, the candidate records produced by the preceding retrieval stages are passed forward without query-conditioned record-level selection. This variant tests whether reranking is necessary to prioritize answer-relevant records and remove candidates that are structurally or semantically related to the query but insufficient for answering it.

Table 13: Retrieval and temporal design ablations on LoCoMo and LongMemEval with OSS-20B.

Configuration	LoCoMo	LongMemEval
Full retrieval pipeline	84.0	76.2
No keyword extraction	79.9 (−4.1)	73.0 (−3.2)
No temporal rewriting	81.0 (−3.0)	75.4 (−0.8)
No compressed-record reranking	79.3 (−4.7)	74.2 (−2.0)

Keyword extraction and compressed-record reranking contribute consistently across both benchmarks. Removing keyword extraction reduces accuracy by 4.1 points on LoCoMo and 3.2 points on LongMemEval, while removing compressed-record reranking causes drops of 4.7 and 2.0 points, respectively.

E.4 Joint Interpretation

Tables 14 and 15 report the complete category-level ablation results. For LongMemEval, User denotes single-session user, Multi-sess. denotes multi-session reasoning, Pref. denotes single-session preference, Assistant denotes single-session assistant, Temporal denotes temporal reasoning, and KU denotes knowledge update.

Across the three ablation groups, graph retrieval, direct compressed-record retrieval, and provenance-linked evidence provide complementary rather than interchangeable functions. The graph path connects related entities and events, direct compressed-record retrieval recovers content outside the selected graph seeds, and fine-grained evidence supplies traceable source material to the answer model.

The category-level results also reveal non-monotone effects. Some component removals improve individual categories despite reducing overall accuracy. For example, removing temporal rewriting increases LongMemEval preference accuracy from 60.0 to 73.3, while removing spreading activation increases multi-session accuracy from 61.7 to 63.9. We therefore interpret the ablations as evidence of complementary overall contributions rather than uniform improvements across every category.

F LLM-as-Judge Evaluation Protocol

All newly generated GRACE-Mem results and controlled ablations in this paper are evaluated by an LLM judge (GPT-4o-mini, temperature 0) that compares the model answer to the gold answer and returns a binary CORRECT/WRONG label. This appendix documents the two components of that

protocol: the single-pass judge and an abstention-specific rubric.

Single-pass judge. For a normal question, the judge receives the question, the gold answer, and the generated answer under a semantic-equivalence rubric: an answer is CORRECT if it refers to the same fact or entity as the gold answer, regardless of surface form. Time questions are evaluated by the same semantic-equivalence criterion—a relative phrasing (“last Tuesday”) or a reformatted date (“May 7th” vs. “7 May”) counts as correct if it denotes the same date. To reduce spurious mismatches on relative-date gold answers, relative temporal expressions in the gold answer (e.g. “the week before 25 May 2023”, “the Sunday before X”, or “a few days before X”) are deterministically expanded to an absolute date or interval and passed to the judge as a normalization hint before grading. The judge emits a one-sentence rationale followed by the label in JSON. Each question is graded once.

LongMemEval abstention (_abs) questions. For LongMemEval only, a subset of questions is *unanswerable*: the asked-about fact was never stated, so the only correct behavior is to abstain (the gold answer is a phrase such as “the information provided is not enough” / “did not mention”, or a count of 0). The generic rubric—which only asks “does the response contain the gold answer”—systematically mislabels clean abstentions as wrong, so these questions are routed to a dedicated rubric. Membership is taken *authoritatively* from the dataset’s _abs tag when available, and otherwise inferred from abstention markers in the gold text. Under the abstention rubric the judge sets CORRECT when the response cleanly declines (states it has no record / insufficient information, or answers “0”/“none” when the event never happened), and WRONG when the response fabricates any concrete number, date, duration, name, or ordering—including the common failure of being led by a similar-but-different distractor mentioned earlier and answering about that instead. Non-abstention questions continue to use the unchanged default rubric.

G Prompt Templates

This appendix presents the core prompt templates used in the reported experiments. We include the behavioral contracts and output formats that

Table 14: Category-level component ablations on LongMemEval with OSS-20B.

Configuration	Overall	User	Multi-sess.	Pref.	Assistant	Temporal	KU
Full retrieval pipeline	76.2	88.6	61.7	60.0	85.7	86.4	71.8
<i>Graph-Structured Retrieval</i>							
No graph retrieval	67.9	88.6	57.1	63.3	87.5	59.1	57.1
No lexical entity retrieval	72.2	91.2	61.4	66.7	91.1	72.7	61.5
No spreading activation	73.0	90.0	63.9	66.7	91.1	71.2	65.4
No graph-level reranking	71.3	90.0	58.6	63.3	89.3	72.0	65.4
<i>Memory and Evidence Representation</i>							
Unsplit turn-pair units	69.5	87.1	63.2	60.0	89.3	65.2	61.5
No direct compressed-record retrieval	69.7	87.1	56.4	70.0	89.3	72.7	57.7
Compressed records only	70.0	89.5	60.9	60.0	89.3	59.8	78.2
<i>Retrieval and Temporal Design Choices</i>							
No keyword extraction	73.0	90.0	63.2	66.7	91.1	73.5	62.8
No temporal rewriting	75.4	94.3	62.4	73.3	87.5	76.5	70.5
No compressed-record reranking	74.2	88.6	62.4	80.0	91.1	75.0	65.4

materially affect the system. Complete verbatim prompts, additional few-shot examples, inactive variants, and implementation wrappers are provided in the accompanying code repository.

G.1 Entity Extraction Prompt

Entity extraction operates on temporally normalized raw dialogue; the normalization is deterministic and does not use an additional LLM prompt.

Prompt for entity extraction

GOAL:
Given the conversation, identify all core entities for knowledge-graph construction.

REFERENCE TIME:
{dialogue_datetime}

RULES:

- Output entities only.
- Use English for all outputs.
- entity_type must be one of: Person, Event, Date, Time, Timespan, Location, Organization, Product, Service, Activity, Topic, Concept.
- Each entity must have a concise and factual description.
- Create Event entities for important actions, plans, decisions, changes, milestones, and incidents.
- Marker tags are authoritative:
[DATE: X] -> Date entity named X
[TIME: X] -> Time entity named X
[TIMESPAN: X] -> Timespan entity named X
- Do not recompute, rename, normalize, or paraphrase marker payloads.
- An Event and its temporal anchor must be represented separately.
- Do not infer an exact date from an unmarked fuzzy expression.

OUTPUT:
("entity" <|> entity_name <|> entity_type <|> entity_description)

Return one flat list separated by <|RECORD|> and terminate with <|COMPLETE|>.

INPUT:
Raw conversation:
{raw_conversation}

G.2 Relationship Extraction Prompt

Relationship extraction is performed in a second stage and may only use entities produced by the preceding entity-extraction stage.

Prompt for relationship extraction

GOAL:
Identify relationships among the provided entities.

REFERENCE TIME:
{dialogue_datetime}

RULES:

- Output relationships only; do not create entities.
- source_entity and target_entity must exactly match a provided entity_name.
- If a required entity is missing, omit the relationship.
- Do not recompute, rename, or paraphrase temporal anchors.
- Prefer relations useful for WHO, WHAT, WHEN, WHERE, and HOW.
- Prefer:
Event -> Date/Time/Timespan for temporal links
Event -> Location for spatial links
- relationship_keywords must contain 2--4 lowercase English keywords.

OUTPUT:
("relationship" <|> source_entity <|> target_entity <|> relationship_description <|> relationship_keywords)

Return one flat list separated by <|RECORD|> and terminate with <|COMPLETE|>.

INPUT:

Table 15: Category-level component ablations on LoCoMo with OSS-20B.

Configuration	Overall	Single-hop	Multi-hop	Open-domain	Temporal
Full retrieval pipeline	84.0	90.1	81.6	68.8	74.5
<i>Graph-Structured Retrieval</i>					
No graph retrieval	66.4	77.6	57.4	55.2	48.3
No lexical entity retrieval	80.2	86.3	76.1	60.2	73.8
No spreading activation	81.2	86.9	78.7	60.4	74.5
No graph-level reranking	81.1	86.3	79.4	62.5	74.5
<i>Memory and Evidence Representation</i>					
No direct compressed-record retrieval	80.2	85.8	77.7	59.4	73.8
Compressed records only	76.2	87.4	77.7	60.4	50.6
<i>Retrieval and Temporal Design Choices</i>					
No keyword extraction	79.9	86.2	76.2	55.2	73.8
No temporal rewriting	81.0	87.4	76.6	59.4	74.5
No compressed-record reranking	79.3	83.8	76.2	62.5	75.1

```
Raw conversation:
{raw_conversation}

Provided entities:
{entities_text}
```

```
===END===

CONSTRAINTS:
- UPDATE target IDs must be copied from the valid
  candidate IDs.
- ADD must use NULL as the target ID.
- merged_description must not be empty.
```

G.3 Entity Reconciliation Prompt

Entity reconciliation determines whether each newly extracted entity should be added as a new graph node or used to update an existing candidate node. The merged description is produced by the same call rather than by a separate consolidation pass.

The two few-shot examples below are reproduced verbatim from the prompt, with line breaks inserted to fit the column. The first shows that a differing type label does not by itself imply a distinct entity; the second shows the subject guard.

Prompt for entity reconciliation

TASK:
Reconcile each newly extracted entity against the existing candidate entities.

DECISION POLICY:

- Choose UPDATE only when the new entity and a candidate clearly denote the same real-world object, person, event, or concept.
- Type differences alone do not imply separation.
- Similarity score is only a hint, not proof.
- Related but distinct entities must remain separate.
- When uncertain, choose ADD.

SUBJECT GUARD:
Events whose primary subjects are different named people must remain separate, even when their names and descriptions are similar.

DESCRIPTION MERGE:
For UPDATE, merged_description must:

- preserve facts from the existing description;
- preserve facts from the new description;
- include new temporal, role, or version information;
- remain concise and factual.

OUTPUT:
===BEGIN===
input_name || input_type || action ||
target_existing_id_or_NULL || canonical_name ||
canonical_type || merged_description

Entity reconciliation few-shot example: UPDATE

[INPUT]
name=OpenAI | type=organization | desc=AI research and deployment company; created GPT models.
Candidates:
- id=org_OpenAI | name=OpenAI | type=org | score=0.91 | desc=An AI company known for foundation models and an API platform.
Valid target_existing_id choices: ["org_OpenAI"]

EXPECTED OUTPUT (between markers):
===BEGIN===
OpenAI||organization||UPDATE||org_OpenAI||OpenAI||organization||An AI research and deployment company known for foundation models and an API platform; creator of GPT models.
===END===

Entity reconciliation few-shot example: conservative ADD

[INPUT]
name=check-up with doctor | type=event | desc=Sam visited the doctor for a routine check-up a few days before May 24.
Candidates:
- id=event_checkup_001 | name=check-up with doctor | type=event | score=0.83 | desc=Evan's son had a

doctor check-up in early August due to recurring stomach issues.
Valid target_existing_id choices: ["event_checkup_001"]

EXPECTED OUTPUT (between markers):
===BEGIN===
check-up with doctor||event||ADD||NULL||check-up with doctor||event||Sam visited the doctor for a routine check-up a few days before May 24.
===END===

G.4 Keyword Extraction Prompt

Prompt for retrieval keyword extraction

You extract retrieval keywords for memory question answering.

Return JSON only with exactly two keys:

```
{
  "high_level_keywords": [
    abstract intent or reasoning type
  ],
  "low_level_keywords": [
    concrete entities, events, dates, and lexical anchors
  ]
}
```

RULES:

- Do not hallucinate missing facts or the final answer.
- low_level_keywords must not be empty.
- Preserve names, organizations, activities, objects, dates, time expressions, and event phrases.
- Ignore first-person pronouns and retain the actual retrieval target.
- For temporal, counting, duration, or comparison questions, include the reasoning intent in high_level_keywords.

QUERY: {query}

G.5 Agent Filter Prompt

The agent performs evidence selection rather than answer generation, validating retrieved candidates and recovering relevant turns missing from the initial set. The VECTOR tool block is injected only when a compressed-record vector index is available for the question; otherwise the tool is not shown to the agent.

System prompt for the evidence-selection agent

You are an evidence-selection agent for a long-term memory question-answering system.

You receive a QUESTION and candidate evidence turns, each with a provenance sid. Candidates may contain distractors, and relevant turns may be missing.

Use the available tools to verify candidates, search the raw conversation corpus, and return the final evidence sids.

TOOLS:

GREP <regex>
Case-insensitive search over all raw conversation turns. Dates, names, numbers, and distinctive phrases are searchable.

READ <sid> [k]

Return the raw turns surrounding sid. The default window is 2.

VECTOR <query>
Semantic search over compressed conversation records. Present only when a compressed-record index exists for the question. Results are unverified and must be checked with READ or GREP before inclusion.

FINAL <sid> <sid> ...
Return the selected evidence sids.

RULES:

- Emit exactly one command as the final line of each response.
- Copy every sid exactly as displayed.
- Do not repeat a search that produced zero results.
- Verify suspicious candidates with READ.
- Remove distractors and preserve every turn that supports the answer.
- Search for missing evidence when the initial candidates are incomplete.
- For counting questions, collect every distinct relevant mention.
- For updated facts, retain the dated evidence needed to identify the most recent value.
- Use no more than {max_calls} tool calls.
- Stop and return FINAL as soon as the evidence is sufficient.

G.5.1 User-turn template

The selected hint is inserted directly, without a field label. The question-date line is present only when the question date is known.

User message for the evidence-selection agent

QUESTION: {question}

QUESTION DATE:
{question_date}
{category_hint}

CANDIDATE EVIDENCE TURNS: {candidates}

Verify the candidates, search for missing evidence, and return the final evidence sids.

G.6 Answer Generation Prompts

All reported LongMemEval settings, including the agent-filter configuration, use the same default answer prompt: the agent filter changes the retrieved context rather than the answering instruction. LoCoMo uses a different template in which the instruction is carried by the user turn.

In the template below, the date block is appended only when the question date is known.

Default LongMemEval answer prompt

SYSTEM:
You are a concise and accurate assistant.

Use the Retrieved Context. If the context is insufficient, general knowledge may be used, but retrieved facts should be preferred.

Answer directly.

Current Date/Time: {question_date}
 Note: When answering temporal questions (e.g., "how long ago", "how many months"), calculate based on this date.

---Retrieved Context---
 {context}

USER:
 Question: {question}

Answer:

LoCoMo answer template

SYSTEM:
 ---Retrieved Context---
 {context}

USER:
 Please answer based on the retrieved knowledge-graph context above. Be concise and accurate.

Question: {question}

Answer:

G.7 Evaluation Judge Prompts

These prompts are used only for automatic evaluation and do not affect retrieval or answer generation.

G.7.1 LongMemEval judge

LongMemEval evaluation template

Evaluate whether the generated answer contains the correct answer to the question.

QUESTION: {question}

REFERENCE ANSWER:
 {gold_answer}

GENERATED ANSWER:
 {predicted_answer}

CATEGORY-SPECIFIC POLICY:
 {category_rubric}

Return JSON:
 {
 "reasoning": "one-sentence explanation",
 "correct": true or false
 }

LongMemEval abstention (_abs) judge. Questions marked _abs ask for information that was never provided. They use the following dedicated template rather than the category-specific template above. The template deliberately distinguishes an answer to the asked-about item from an explicitly labelled comparison with a different, similar item: a concrete value is disqualifying only when it is offered as the answer to the asked-about item.

Table 16: LongMemEval category-specific judging criteria.

Category		Additional criterion
Temporal reasoning	re-	Minor off-by-one duration errors are accepted.
Knowledge update	ason-	The latest required value determines correctness, even if older information is also mentioned.
Preference	ing	Correct use of the user’s personal information is sufficient; not every reference point must be repeated.
Other categories	cate-	The generated answer must contain or be semantically equivalent to the required answer.

LongMemEval abstention evaluation template

Evaluate an ABSTENTION question: the asked-about information was never provided, so the only correct behavior is to abstain.

QUESTION:
 {question}

REFERENCE ANSWER:
 {gold_answer}

GENERATED ANSWER:
 {predicted_answer}

Set \texttt{correct=true} when the response clearly states that it has no record / insufficient information / cannot answer, or correctly answers ``0' or ``none' when the event never occurred. A response remains correct when it abstains on the asked-about item while explicitly identifying a different, similar item as a contrast; any value must be clearly attributed to that different item, not presented as the requested answer. It is also correct to offer conditional help (for example, to calculate the result if the user provides missing input).

Set \texttt{correct=false} when the response gives or borrows a definite number, date, duration, name, or ordering as the answer to the asked-about item, including an unlabelled substitution of a similar-but-different distractor.

Return JSON:
 {
 "reasoning": "one-sentence explanation",
 "correct": true or false
 }

LongMemEval judging schedule. Each item receives a single GPT-4o-mini judgment at temperature 0. Non-_abs items are evaluated using their category-specific templates, whereas _abs items use the abstention template described above.

G.7.2 LoCoMo judge

LoCoMo standard evaluation template

SYSTEM:

You are an expert grader that determines whether an answer matches a gold-standard answer.

USER:

Label the generated answer as CORRECT or WRONG.

Question:

{question}

Gold answer:

{gold_answer}

Generated answer:

{response}

Be generous about semantically equivalent answers and equivalent date formats. Return JSON only:

{"label": "CORRECT"}

or

{"label": "WRONG"}