KITE: RETRIEVAL AS A DECISION PROCEDURE FOR LONG-TERM CONVERSATIONAL MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Many long-term memory systems for LLM agents are built around embedding-based retrieval. Although effective for semantic matching, similarity ranking alone does not establish whether retrieved memories satisfy the conditions implied by a question. In this paper, we introduce KITE (**K**nowledge-**I**ndexed **T**emporal **E**vidence), a vector-free framework that formulates memory retrieval as an explicit decision procedure. KITE organizes multi-session conversations as structured knowledge indexed by a governed topic vocabulary, which evolves through controlled expansion and consolidation. At query time, questions are compiled into inspectable retrieval plans over the same knowledge index that guides memory construction. The plans are then executed through fixed lookup and ranking operations, while sparse lexical retrieval recovers surface details omitted from the structured representation. Experiments on LoCoMo and LongMemEval-S show that KITE achieves 93.5% and 86.0% accuracy while using only about 1.6k reader-context tokens per question. These results demonstrate that conversational memory can move beyond similarity ranking by unifying memory construction and query compilation around a governed index, turning retrieval into an inspectable process for determining which evidence satisfies the question.

1 INTRODUCTION

As large language model (LLM) agents engage in multi-session interactions, they need to carry useful knowledge forward from earlier exchanges. Although larger context windows expose more history, they do not inherently support selective retention (Hsieh et al., 2024). Persistent memory has therefore become a key component of long-term LLM-agent architectures (Zhong et al., 2024).

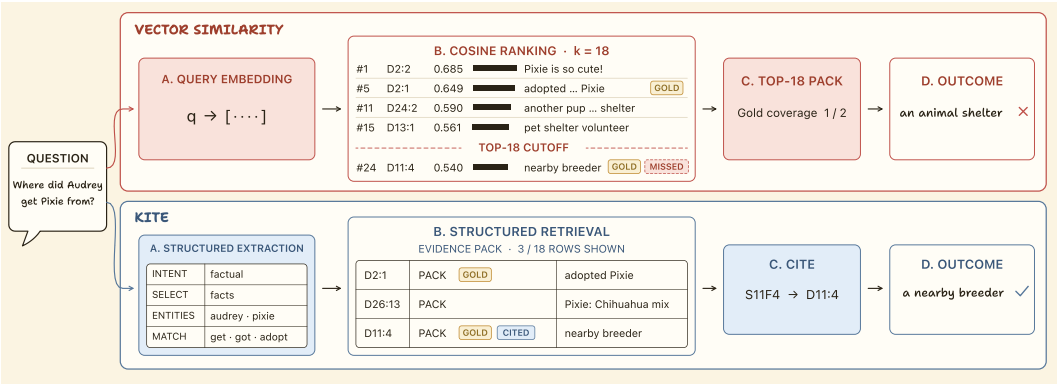

Figure 1: Comparison of embedding retrieval and KITE's retrieval with explicit constraints.

In practice, many memory systems retrieve past information through nearest-neighbor search over embeddings (Packer et al., 2023; Chhikara et al., 2025). Although effective for semantic matching, single-vector similarity ranking alone does not ensure that the retrieved memories satisfy query conditions (Weller et al., 2026). As Figure 1 illustrates, semantically close memories may outrank evidence that answers the question (Trappolini et al., 2026). In addition, fixed top-k selection returns

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

candidates without explicitly indicating when none satisfies the query (Kim et al., 2025). Beyond retrieval, storing N memories as d-dimensional dense vectors requires $O(Nd)$ space, while encoder changes often require rebuilding the entire index (Monath et al., 2024; Goswami et al., 2025).

These limitations of embedding-based retrieval motivate a formulation of memory retrieval as an explicit decision procedure that evaluates query conditions rather than relying on similarity ranking alone (Wang et al., 2026). By exposing matching conditions and intermediate results, this approach makes retrieval decisions inspectable and supports failure diagnosis (Ru et al., 2024). However, as conversations evolve, the same concepts may be represented by different labels or graph nodes, complicating consistent indexing (Zhang & Soh, 2024; Mo et al., 2025). Moreover, questions may refer to stored information through different wording or implicit cues, making relevant evidence harder to retrieve (Hu et al., 2026b). These challenges call for a governed knowledge index that evolves with the conversation, together with query compilation grounded in the same representation.

To this end, we propose KITE[1] (**K**nowledge-**I**ndexed **T**emporal **E**vidence), a vector-free framework for long-horizon conversational memory that organizes multi-session interactions into structured knowledge indexed by a governed topic vocabulary. Each memory entry is further enriched with typed contextual and temporal facets, providing complementary access paths beyond topics. At query time, KITE compiles a natural-language question into an inspectable retrieval plan over the same governed knowledge index, specifying the conditions to apply. The validated plan is then executed through explicit lookup and ranking operations, making evidence selection deterministic. In addition, KITE supplements the primary path with sparse lexical retrieval to recover surface details missed during memory construction. Experiments on LoCoMo (Maharana et al., 2024) and LongMemEval-S (Wu et al., 2025) show that KITE achieves accuracies of 93.5% and 86.0%, respectively, while remaining token-efficient by limiting reader context to a compact evidence set.

The key contributions of this paper are as follows:

- **Retrieval as a deterministic decision procedure.** We formulate memory retrieval as a decision procedure that evaluates explicit query conditions against indexed evidence. Validated plans are then executed through fixed lookup and ranking operations, making evidence selection deterministic for a given plan and inspectable. These mechanisms establish a practical alternative to similarity-based retrieval while maintaining high accuracy with compact reader contexts.
- **A governed and evolving knowledge index.** We organize memories using a topic vocabulary that evolves across multi-session conversations under controlled expansion and consolidation, complemented by typed facets that provide finer-grained access to evidence. This design limits topic fragmentation and keeps memories coherently organized as conversations grow.
- **A shared index for memory construction and query compilation.** We align question interpretation with memory organization through a shared knowledge index. By mapping concepts in the question to existing index keys, we ground retrieval plans in stored topics and entities, reducing the semantic gap between natural-language phrasing and indexed knowledge.

2 Related Work

2.1 Long-Term Memory for LLM Agents

Extending the context window of language models is a direct approach to supporting long-horizon interaction (Beltagy et al., 2020; Bulatov et al., 2022). However, longer contexts increase inference costs, while model performance remains susceptible to position-dependent degradation (Liu et al., 2024; Hutter et al., 2025). Persistent-memory systems instead preserve information across sessions beyond the active context window. For example, MemGPT (Packer et al., 2023) manages information transfer between the model context and the external storage. In addition, MemoryBank (Zhong et al., 2024) incorporates time-dependent forgetting, while Mem0 (Chhikara et al., 2025) extracts salient facts into an editable memory store and supports add, update, and delete operations.

Beyond individual memory operations, several works investigate broader mechanisms for managing information accumulated over long interactions. A prominent direction explores system-level memory architectures with hierarchical storage and lifecycle control (Kang et al., 2025; Li et al., 2025).

[1] https://anonymous.4open.science/r/kite-memory/

Meanwhile, MEM1 (Zhou et al., 2026) consolidates the interaction history into a compact internal state that supports continued reasoning. Together, these approaches show that long-term memory depends not only on persistence, but also on how accumulated information is organized for later use.

2.2 Memory Organization and Index Governance

Recent work has explored structured representations for organizing conversational memory. A-MEM (Xu et al., 2025) builds interconnected notes whose attributes and links evolve as new memories arrive. Similarly, Zep (Rasmussen et al., 2025) organizes conversational knowledge as a temporal knowledge graph that tracks facts and their validity over time. Alongside these relational approaches, Memanto (Abtahi et al., 2026) structures memories through predefined semantic types, while EverMemOS (Hu et al., 2026a) consolidates episodic MemCells into thematic MemScenes. In general, these approaches provide richer structures for organizing related memories over time.

As these structures grow, a related challenge is maintaining consistent representations of recurring concepts. EDC (Zhang & Soh, 2024) canonicalizes extracted relations into a consolidated schema, including settings without a predefined target schema. KGGen (Mo et al., 2025) addresses the same problem from a graph-construction perspective by clustering entities and relations to reduce fragmentation. These works highlight the importance of keeping related concepts aligned and reducing redundant representations, providing a more stable basis for subsequent retrieval.

2.3 Query Alignment and Executable Retrieval

Once memories are organized, retrieval must connect each question to the representations in which evidence is stored. Sparse lexical methods such as BM25 (Robertson & Zaragoza, 2009) rank candidates by scoring matches between query terms and stored text. However, questions may express the same information in different words, making relevant memories harder to find through lexical matching alone. QueryLink (Hu et al., 2026b) addresses this mismatch by constructing semantic representations of questions and memories at different levels of detail for similarity-based retrieval.

Complementary work makes memory access explicit through executable operations. ChatDB (Hu et al., 2023) treats databases as symbolic memory, allowing an LLM to generate SQL instructions for retrieval and manipulation. More generally, Text2Mem (Wang et al., 2026) translates natural-language memory commands into validated operations with a common execution interface. For more structured knowledge, KAG (Liang et al., 2025) combines logical-form planning with hybrid retrieval over knowledge graphs and text, while supporting schema-free knowledge extraction. Executable programs can further support reasoning after retrieval, as in TReMu (Ge et al., 2025), which performs temporal calculations over retrieved timeline summaries. These approaches establish executable operations as a general mechanism for explicit access and reasoning over stored evidence.

This emphasis on explicit operations also aligns with diagnostic work that examines intermediate evidence. RAGChecker (Ru et al., 2024) separates retrieval from generation to localize failures within the pipeline. In long-term memory, MemTrace (Long et al., 2026) distinguishes whether source evidence is reachable from whether it is successfully used. These evaluations highlight the value of making the evidence process inspectable rather than relying on final-answer accuracy alone.

3 Method

To connect memory organization with evidence selection, KITE formulates long-term conversational memory retrieval as an explicit decision procedure over a governed knowledge index. As illustrated in Figure 2, each session is converted into structured memory entries indexed under a governed topic vocabulary and enriched with contextual and temporal facets. At query time, KITE uses the same index to compile each question into an inspectable retrieval plan that specifies the conditions for selecting evidence. The validated plan is then executed deterministically through fixed lookup and ranking operations, while sparse lexical retrieval provides complementary evidence.

Consider a sequence of dated conversation sessions $U = (u_1, \ldots, u_m)$ ordered chronologically. Given a later natural-language question q, the retrieval task is then to recover evidence from this history that supports an answer. KITE constructs a set of structured memory entries M from U and selects relevant entries through a knowledge-indexed retrieval path.

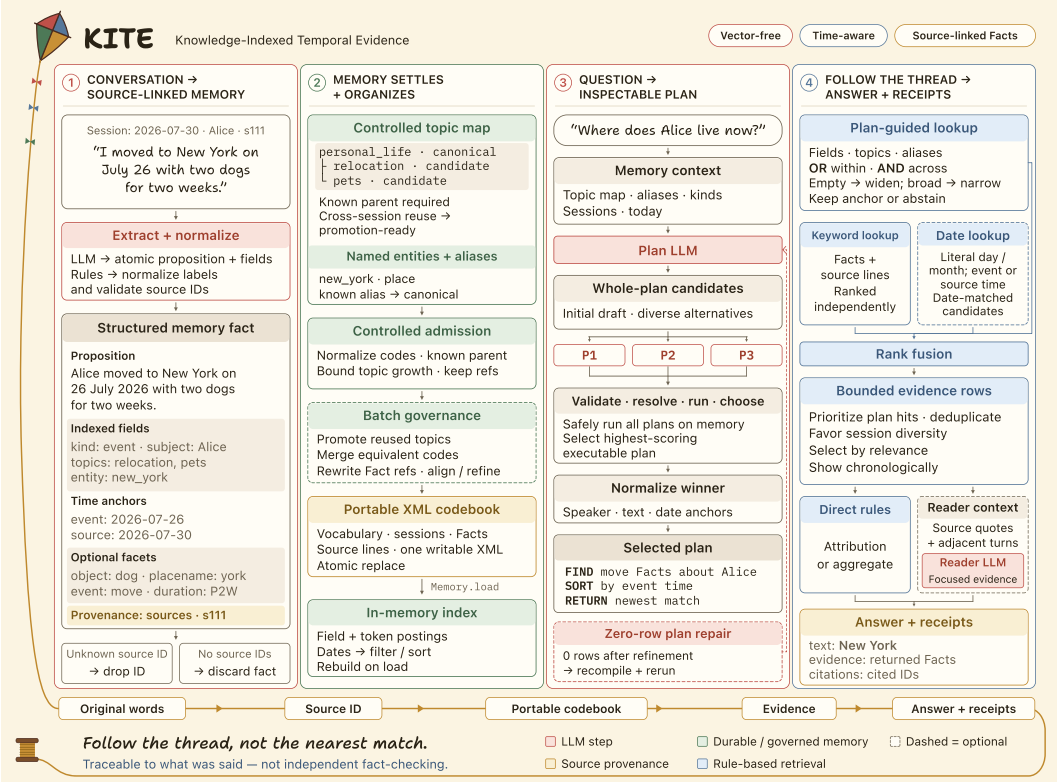

Figure 2: Overview of KITE. Multi-session conversations are converted into structured memory entries indexed by governed topics. Given a question, KITE compiles an inspectable retrieval plan over the same knowledge index to locate relevant contextual and temporal evidence for answering.

Let Σ be the set of governed index keys. Memory indexing and query compilation are defined as

$$\mathcal{I} : M \to 2^{\Sigma}, \qquad \mathcal{Q}_{\mathbf{P}} : \mathcal{X} \to \Pi, \tag{1}$$

where $\mathcal{X}$ is the space of natural-language questions and Π is the space of retrieval plans. For each memory entry $m \in M$, $\mathcal{I}(m) \subseteq \Sigma$ specifies its assigned index keys, and these assignments collectively form the knowledge index $\mathbf{P}$. Given this index, $\mathcal{Q}_{\mathbf{P}}(q) = \pi$ compiles a query q into a retrieval plan π whose indexed references resolve to the same keys used to organize the stored memories.

After validation, the indexed path executes the plan π and returns a ranked evidence sequence E_q as

$$E_q = \mathcal{D}_{\mathbf{P},\theta}(\pi) = \mathrm{Rank}_{\theta}\left(\mathrm{Exec}_{\mathbf{P}}(\pi)\right), \tag{2}$$

where $\mathrm{Exec}_{\mathbf{P}}$ applies the plan over the knowledge index and Rank_{θ} orders the resulting candidates according to a fixed policy θ. For fixed $\mathbf{P}$, π, and θ, E_q is deterministic, and the procedure returns no match instead of forcing unrelated candidates when no indexed evidence satisfies the plan.

3.1 Governed Knowledge Index

Structured memory entries and provenance. Rather than indexing raw dialogue directly, KITE converts each conversation session into structured memory entries. Each entry can be represented as

$$m = \langle x_m, \rho_m, T_m, A_m \rangle, \tag{3}$$

where x_m is the self-contained memory content and ρ_m records its provenance, including the source session and supporting utterances. The set T_m contains the governed topics assigned to the entry, while A_m stores typed contextual and temporal facets, such as entities and events. Together, these fields separate the evidence itself from the structured information used to index and retrieve it.

For each session, the extraction model processes the dialogue with the current topic vocabulary as context, producing source-linked entries with facet annotations. Relative times are also resolved against the session date whenever possible, while entries without valid provenance are discarded.

Governed topic vocabulary. Rather than assuming a complete ontology in advance, KITE derives its topic vocabulary from the accumulated conversation history. Starting from a small set of canonical roots, the extraction model reuses existing topics when possible and gradually proposes more specific children when new concepts emerge. To preserve consistency during this process, new topics must fit within the existing hierarchy, which helps maintain a stable index over time.

Vocabulary expansion is further controlled by keeping proposed topics provisional until they recur across multiple sessions and time periods. As the vocabulary evolves, potentially redundant sibling topics are consolidated, with their memory entries redirected to the selected canonical topic. These controls reduce unnecessary topic fragmentation while preserving stable index keys for retrieval.

Faceted knowledge indexing. Topics provide broad thematic access to memory, while typed facets offer finer-grained conditions for locating relevant evidence. The index space is defined as

$$\Sigma = \Sigma_{\text{topic}} \cup \Sigma_{\text{entity}} \cup \Sigma_{\text{participant}} \cup \Sigma_{\text{object}} \cup \Sigma_{\text{place}} \cup \Sigma_{\text{event}}. \tag{4}$$

A memory entry may therefore be assigned keys from multiple types, providing complementary access paths to the same evidence. For example, consider the LoCoMo question *In which state is the shelter from which James adopted the puppy?* (Maharana et al., 2024). The dialogue describes James adopting a puppy named Ned from a shelter in Stamford. KITE indexes this entry with `James` as a participant, `Ned` as an entity, `dog` as an object, `Stamford` as a place, and `adoption` as an event. Although the question does not explicitly mention `Ned`, the participant and event keys can provide access to the relevant entry, where the required location information can be recovered.

For each key type a, KITE constructs a posting list:

$$P_a[s] = \{m \in M \mid s \in \mathcal{I}_a(m)\}, \tag{5}$$

where $s \in \Sigma_a$ denotes an index key and $\mathcal{I}_a(m)$ gives the keys of type a assigned to memory entry m. Each posting list therefore provides direct access to the memories associated with a given key. Meanwhile, temporal annotations are stored separately in chronological order, enabling time constraints to be evaluated during retrieval. These structures form the governed knowledge index $\mathbf{P}$.

3.2 Index-Grounded Query Compilation

Shared-index grounding. Query compilation uses the same knowledge index built during memory construction, allowing each question to be interpreted against the vocabulary and entities already stored. As a result, references inferred from the question can be resolved to existing index keys, keeping query interpretation aligned with memory organization. The compiler then maps the question to these keys, while the executor determines which memories satisfy the resulting conditions.

Typed retrieval plans. Each question is compiled into a typed retrieval plan that makes its evidence requirements explicit before execution. Formally, the plan consists of one or more subqueries

$$\pi = (\pi_1, \ldots, \pi_n), \qquad \pi_i = \langle \texttt{select}, \texttt{where}, \texttt{pipe} \rangle, \quad i = 1, \ldots, n. \tag{6}$$

Here, the `select` field determines the retrieval granularity for each subquery, while `where` specifies the indexed and temporal conditions that evidence must satisfy. Finally, the `pipe` field applies fixed operations to the resulting evidence, such as sorting, limiting, or counting.

Subqueries can be linked sequentially, allowing results from an earlier step to constrain later search. These dependencies keep multi-hop retrieval explicit in the plan while preserving fixed execution.

Plan validation. Before execution, each plan is validated against a fixed typed grammar and the knowledge index. Indexed references are resolved to valid keys in the governed representation. Speaker mentions are mapped to participant constraints, while temporal expressions are normalized into explicit ranges. Plans containing unresolved references or unsupported structures are rejected. The validated plan is executed by the retrieval executor without further model intervention, establishing a clear boundary between model-based interpretation and deterministic evidence selection.

3.3 Retrieval as a Decision Procedure

Deterministic plan execution. Given a validated subquery π_i, KITE evaluates its indexed conditions through the corresponding posting lists. Requested keys within the same type are treated as

alternatives, whereas conditions across types must hold jointly. The initial candidate set is therefore

$$C(\pi_i) = \bigcap_{a \in \mathcal{A}(\pi_i)} \left(\bigcup_{s \in \pi_{i,a}} P_a[s] \right), \tag{7}$$

where $\mathcal{A}(\pi_i)$ denotes the constrained key types and $\pi_{i,a} \subseteq \Sigma_a$ contains the requested keys of type a. Temporal and lexical conditions further filter this set before the `pipe` operations are applied.

In practice, the executor narrows the candidate memories according to the conditions encoded in the plan. For dependent subqueries, earlier results are carried forward as constraints on later retrieval. With these operations fixed, the same index and validated plan always yield the same candidate set.

Controlled relaxation and no-match. When exact execution produces no candidates, KITE applies a predefined sequence of relaxations. Each step broadens only designated conditions while preserving bindings established by earlier retrieval stages. The effective conditions remain explicit throughout the process, and execution stops at the first non-empty result. If all permitted relaxations fail, the indexed path explicitly returns no match under the current index and execution policy.

Explicit evidence ranking. After condition evaluation determines the candidate set, KITE orders the retained memories using a fixed scoring policy

$$S_\theta(m, \pi) = \sum_{j \in \mathcal{J}} w_j \, \phi_j(m, \pi), \tag{8}$$

where each $\phi_j(m, \pi)$ captures an interpretable match between memory entry m and retrieval plan π, and w_j is its weight under policy θ. In practice, exact key matches receive the strongest preference, while broader hierarchical matches or agreement on additional facets provide supporting signals.

For example, a memory matching both the queried entity and event receives a higher score than one linked only through a broader topic. Importantly, ranking is applied only after condition checks have admitted the candidates, so it changes their ordering rather than eligibility. For fixed $\mathbf{P}$, π, and θ, the resulting scores are deterministic, while the basis of each ranking decision remains inspectable.

Sparse lexical support. Structured memory is designed to capture reusable concepts and retrieval cues, but this abstraction may omit surface-form details preserved in the source text, such as rare names or numerical values. To retain access to such information, KITE augments the indexed path with a deterministic sparse lexical channel. Accordingly, the ranked evidence from both paths is merged under a fixed fusion rule, without relying on dense embeddings or neural reranking.

A retrieval attempt may fail to produce a structured match without failing to find anything at all. In that case the result is recorded as a fallback, and the lexical channel may still provide evidence to the reader, marked as such rather than as a match. The reader thus gains additional evidence while the meaning of a match remains unchanged.

4 Experiments

4.1 Experimental Setup

Benchmarks. We evaluate KITE on LoCoMo (Maharana et al., 2024) and LongMemEval-S (Wu et al., 2025). Specifically, the LoCoMo experiments cover 1,540 questions across ten multi-session conversations, while LongMemEval-S comprises 500 questions over longer interaction histories.

Baselines. We compare KITE against prior systems, namely Mem0 (Chhikara et al., 2025), Zep (Rasmussen et al., 2025), MemOS (Li et al., 2025), and EverMemOS (Hu et al., 2026a). We also consider Naive RAG as a retrieval baseline and full context as a non-retrieval reference.

Evaluation protocol. We evaluate answer quality using LLM-as-a-judge accuracy, reported overall and by question category. Throughout the KITE experiments, `gpt-4.1-mini` serves as both the answer reader and the judge. The reader receives only the question and retrieved evidence, without access to the full conversation history, and answers strictly based on the supplied context.

Table 1: Comparison of memory systems on LoCoMo using LLM-as-a-judge accuracy (%).

System	Avg. Tokens	Accuracy (%)				
		Single	Multi	Temp	Open	Overall
Naive RAG	**0.7k**	42.6	42.6	67.9	71.9	49.7
Full Context	20.3k	94.5	90.1	88.2	70.8	90.9
Mem0	1.0k	69.0	61.7	58.3	50.0	64.2
Zep	1.4k	90.8	81.9	77.3	75.0	85.4
MemOS	2.5k	85.4	79.4	75.1	64.6	80.8
EverMemOS	2.3k	**96.7**	91.8	89.7	76.0	93.1
KITE	1.5k	95.2	**93.3**	**92.2**	**83.3**	**93.5**

Avg. Tokens denotes average reader-context tokens. Best results are **bolded**.

Table 2: Comparison of memory systems on LongMemEval-S using LLM-as-a-judge accuracy (%).

System	Avg. Tokens	Accuracy (%)						
		SS-U	SS-A	SS-P	Multi	Know	Temp	Overall
Naive RAG	5.1k	90.0	**98.2**	53.3	48.9	67.9	39.8	61.0
Full Context	105.1k	87.1	89.3	36.7	45.1	76.9	31.6	56.8
Mem0	**1.1k**	82.9	26.8	90.0	63.2	66.7	72.2	66.4
Zep	1.6k	92.9	75.0	53.3	47.4	74.4	54.1	63.2
MemOS	1.4k	95.7	67.9	**96.7**	70.7	74.4	77.4	77.8
EverMemOS	2.8k	97.1	85.7	93.3	73.7	89.7	77.4	83.0
KITE	1.6k	**98.6**	85.7	63.3	**82.0**	**91.0**	**85.7**	**86.0**

SS-U, SS-A, and SS-P denote single-session user, assistant, and preference questions; Know and Temp refer to knowledge-update and temporal-reasoning questions.

Implementation details. Unless otherwise specified, KITE combines plan-guided retrieval over the knowledge index with a sparse lexical channel under a fixed fusion strategy. We use small benchmark-specific seed vocabularies and apply the same governance procedure across both benchmarks. Further implementation details are provided in Appendices B–C, and complete code is available at `https://anonymous.4open.science/r/kite-memory/`.

4.2 Benchmark Performance

Superior vector-free performance. Tables 1 and 2 show that KITE achieves the highest overall accuracy among the compared systems without dense embeddings or neural reranking. It reaches 93.5% on LoCoMo, outperforming EverMemOS by 0.4 percentage points, and 86.0% on the LongMemEval-S questions, widening the margin to 3.0 percentage points. Compared with full-context inference, KITE further improves accuracy by 2.6 and 29.2 percentage points on LoCoMo and LongMemEval-S, respectively. Overall, these results show that strong long-term memory performance does not require dense similarity retrieval or full-history input to the reader.

Strong temporal and multi-hop reasoning. Notably, the strongest advantage of KITE appears in temporal reasoning, where it achieves 92.2% on LoCoMo and 85.7% on LongMemEval-S, with the latter exceeding the next-best result by 8.3 percentage points. This gain is consistent with KITE's explicit treatment of time, where temporal information is preserved during memory construction and translated into constraints at query time to identify chronologically appropriate evidence. Beyond temporal reasoning, KITE also excels on multi-hop questions by carrying earlier evidence across dependent retrieval steps to connect information between sessions.

Compact reader context. Despite its high accuracy, KITE uses an average of 1.5k reader-context tokens per question on LoCoMo and 1.6k on LongMemEval-S. By comparison, EverMemOS uses 2.3k and 2.8k tokens on the same benchmarks respectively, while full-context inference requires

Table 3: Accuracy (%) and recall (%) of retrieval strategies over fixed memories.

Retrieval Strategy	LoCoMo		LongMemEval-S	
	Acc.	Rec.	Acc.	Rec.
Dense	85.4	78.1	76.9	90.5
BM25	81.3	73.2	72.4	84.6
Hybrid	87.8	79.9	78.0	92.8
KITE	**93.5**	**87.4**	**86.0**	**97.6**

Table 4: Comparison of topic vocabulary policies for memory indexing on LoCoMo.

Vocabulary Policy	Avg. Topics	Avg. Tokens	Acc. (%)	Rec. (%)
Fixed Vocabulary	12.0	2.1k	91.0	**94.2**
Immediate Expansion	72.4	**1.5k**	87.9	78.3
Delayed Expansion	62.5	1.6k	91.7	82.7
KITE	58.0	1.6k	**93.5**	87.4

20.3k and 105.1k. Together, these results show that KITE combines leading overall accuracy with a compact evidence set, substantially reducing the amount of information presented to the reader.

Preference reasoning remains challenging. However, KITE is less competitive on single-session preference questions, achieving 63.3% on LongMemEval-S. This result points to the difficulty of consistently aligning indirect or paraphrased expressions with the indexed memory representation.

4.3 Effectiveness of Deterministic Retrieval

To isolate the effect of retrieval from memory construction, we freeze the memory store and compare retrieval strategies over the same underlying representations. Specifically, we compare dense-vector retrieval, BM25, and their hybrid with KITE's plan-guided retrieval. To ensure a controlled comparison, all methods share identical reader settings and reader-context token budgets, with LLM-as-a-judge accuracy and evidence recall measuring answer quality and retrieval effectiveness.

Table 3 shows that KITE achieves the highest accuracy and recall on both benchmarks with memory construction held fixed. Notably, recall is consistently lower than accuracy on LoCoMo because the gold evidence often includes more dialogue turns than are strictly necessary to answer the question. A correct answer may therefore be produced without retrieving every annotated turn. Even so, KITE improves recall over the hybrid baseline by 7.5 percentage points, indicating that KITE retrieves a more complete set of supporting evidence, rather than only the most relevant individual matches.

By contrast, LongMemEval-S measures recall at the session level, leading to uniformly higher values across methods. Nevertheless, KITE remains consistently stronger in both recall and answer accuracy. This result further supports the effectiveness of KITE's deterministic retrieval, which evaluates explicit query conditions before ranking rather than relying on relevance scores alone.

4.4 Effectiveness of Vocabulary Governance

In addition, we examine how governance of the topic vocabulary affects index growth and query compilation on LoCoMo. All variants start from the same canonical topic roots and process conversations in chronological order. *Fixed Vocabulary* keeps the initial topic set unchanged, whereas *Immediate Expansion* admits valid new topics as soon as they are proposed. *Delayed Expansion* keeps proposed topics provisional until they recur across multiple sessions and time periods, while KITE further consolidates redundant sibling topics. Each variant uses its resulting topic vocabulary for both memory indexing and query compilation, with all other settings remaining fixed.

Table 4 illustrates the value of governing topic evolution rather than simply restricting or expanding the vocabulary. *Immediate Expansion* yields the lowest recall and accuracy, suggesting that related evidence becomes scattered across too many labels. Conversely, *Fixed Vocabulary* achieves higher recall, but its longer reader context may include tangential memories that complicate answer generation. With controlled admission and consolidation, KITE achieves the highest accuracy with fewer topics than the expansion-only variants and the same average context length as *Delayed Expansion*.

Table 5: Ablation study of KITE on LoCoMo using LLM-as-a-judge accuracy (%).

Category	Indexing	Retrieval		Ranking		Full KITE
	w/o Facets	Indexed Only	Lexical Only	Recency	Neural	
Single	93.2	93.5	85.0	64.4	94.8	**95.2**
Multi	90.4	87.6	86.9	69.1	**95.0**	93.3
Temporal	87.9	86.3	71.3	45.5	91.6	**92.2**
Open	67.7	71.9	64.6	52.1	72.9	**83.3**
Overall	90.0	89.5	81.2	60.6	92.8	**93.5**

4.5 Ablation Studies

We conduct ablation studies on LoCoMo to evaluate the main design choices of KITE. As shown in Table 5, each experiment changes one part of the pipeline while keeping the rest constant. This controlled setup allows the contribution of each design choice to be evaluated separately.

Typed facets. Removing typed facets while retaining topic indexing reduces overall accuracy from 93.5% to 90.0%. The largest declines occur on open-domain and temporal questions, at 15.6 and 4.3 percentage points, respectively. These cases often hinge on distinctions beyond broad thematic alignment, including contextual attributes and time-related cues. Accordingly, such facets complement the topic hierarchy by supplying the finer-grained constraints for isolating supporting evidence.

Retrieval channels. We evaluate the two retrieval channels separately and in combination to assess their individual contributions. Specifically, *Indexed Only* retains only plan-guided retrieval over the governed knowledge index, whereas *Lexical Only* uses the sparse lexical channel alone. The indexed variant outperforms its lexical counterpart across all question categories, while their combination in full KITE achieves the highest overall accuracy. These results support the indexed path as the primary retrieval mechanism, with lexical matching providing complementary evidence.

Explicit ranking. To assess the role of evidence ranking, we replace the explicit scoring policy with two alternatives. *Recency* prioritizes memories from newer sessions, whereas *Neural* evaluates query–memory relevance with a pretrained cross-encoder. The sharp decline under *Recency* indicates that favoring newer memories alone is insufficient for prioritizing supporting evidence. Although *Neural* performs better on multi-hop questions, the explicit policy achieves higher overall accuracy by 0.7 percentage points while keeping ranking decisions inspectable.

5 Conclusion

We introduced KITE, a vector-free framework that formulates long-term conversational memory retrieval as an explicit decision procedure. To this end, KITE organizes conversations into structured memories within a governed knowledge index keyed by topic. At query time, the same index guides the compilation of each question into a retrieval plan, aligning its interpretation with the structure of stored memories. The plan is then executed deterministically to select evidence that satisfies explicit contextual and temporal conditions. In this way, retrieval is driven by whether the available information meets the question's requirements, rather than by semantic similarity alone.

Experiments on LoCoMo and LongMemEval-S show that KITE combines high accuracy with compact reader contexts. It achieves 93.5% and 86.0% using only 1.5k and 1.6k tokens, respectively, with particular strength in temporal and multi-hop reasoning. With the memory store held fixed, plan-guided retrieval excels in answer accuracy and evidence recall, highlighting the role of explicit conditions in recovering supporting information. Vocabulary-governance experiments further suggest that controlled expansion and consolidation improve accuracy by limiting topic fragmentation.

Further improvements could focus on interpreting indirect preference expressions more consistently across memory construction and query compilation. These refinements would strengthen semantic alignment while preserving KITE's deterministic evidence selection via a shared knowledge index.

REFERENCES

Seyed Moein Abtahi, Rasa Rahnema, Hetkumar Patel, Neel Patel, Majid Fekri, and Tara Khani. Memanto: Typed semantic memory with information-theoretic retrieval for long-horizon agents. *arXiv preprint arXiv:2604.22085*, 2026.

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

Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. Recurrent memory transformer. *Advances in Neural Information Processing Systems*, 35:11079–11091, 2022.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready AI agents with scalable long-term memory. In *ECAI 2025: 28th European Conference on Artificial Intelligence*, volume 413 of *Frontiers in Artificial Intelligence and Applications*, pp. 2993–3000. IOS Press, 2025.

Yubin Ge, Salvatore Romeo, Jason Cai, Raphael Shu, Yassine Benajiba, Monica Sunkara, and Yi Zhang. TReMu: Towards neuro-symbolic temporal reasoning for LLM-agents with memory in multi-session dialogues. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 18974–18988, 2025.

Dipam Goswami, Liying Wang, Bartłomiej Twardowski, and Joost van de Weijer. Query drift compensation: Enabling compatibility in continual learning of retrieval embedding models. In *Proceedings of The 4th Conference on Lifelong Learning Agents*, volume 330 of *Proceedings of Machine Learning Research*, pp. 324–340. PMLR, 2025.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? In *First Conference on Language Modeling*, 2024.

Chenxu Hu, Jie Fu, Chenzhuang Du, Simian Luo, Junbo Zhao, and Hang Zhao. ChatDB: Augmenting LLMs with databases as their symbolic memory. *arXiv preprint arXiv:2306.03901*, 2023.

Chuanrui Hu, Xingze Gao, Zuyi Zhou, Dannong Xu, Yi Bai, Xintong Li, Hui Zhang, Tong Li, Chong Zhang, Lidong Bing, and Yafeng Deng. EverMemOS: A self-organizing memory operating system for structured long-horizon reasoning. *arXiv preprint arXiv:2601.02163*, 2026a.

Xuxian Hu, Zhu Teng, Wei Zhang, Ming He, and Jianping Fan. QueryLink: Leveraging query-memory alignment for long-term reasoning in LLM agents. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 15608–15621, 2026b.

Jan Hutter, David Rau, Maarten Marx, and Jaap Kamps. Lost but not only in the middle: Positional bias in retrieval augmented generation. In *European Conference on Information Retrieval*, pp. 247–261. Springer, 2025.

Jiazheng Kang, Mingming Ji, Zhe Zhao, and Ting Bai. Memory OS of AI agent. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 25961–25970, 2025.

Hyuhng Joon Kim, Youna Kim, Sang-goo Lee, and Taeuk Kim. When to speak, when to abstain: Contrastive decoding with abstention. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9710–9730, 2025.

Zhiyu Li, Chenyang Xi, Chunyu Li, Ding Chen, Boyu Chen, Shichao Song, Simin Niu, Hanyu Wang, Jiawei Yang, Chen Tang, et al. MemOS: A memory OS for AI system. *arXiv preprint arXiv:2507.03724*, 2025.

Lei Liang, Zhongpu Bo, Zhengke Gui, Zhongshu Zhu, Ling Zhong, Peilong Zhao, Mengshu Sun, Zhiqiang Zhang, Jun Zhou, Wenguang Chen, Wen Zhang, and Huajun Chen. KAG: Boosting LLMs in professional domains via knowledge augmented generation. In *Companion Proceedings of the ACM on Web Conference 2025*, pp. 334–343, 2025.

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

Xianxuan Long, Zhikai Chen, Shenglai Zeng, Shouren Wang, Kai Guo, and Jiliang Tang. MemTrace: Probing what final accuracy misses in long-term memory. *arXiv preprint arXiv:2606.17328*, 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.

Belinda Mo, Kyssen Yu, Joshua Kazdan, Joan Cabezas, Proud Mpala, Lisa Yu, Chris Cundy, Charilaos Kanatsoulis, and Sanmi Koyejo. KGGen: Extracting knowledge graphs from plain text with language models. *Advances in Neural Information Processing Systems*, 38:30092–30115, 2025.

Nicholas Monath, Will Sussman Grathwohl, Michael Boratko, Rob Fergus, Andrew McCallum, and Manzil Zaheer. A fresh take on stale embeddings: Improving dense retriever training with corrector networks. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 36072–36087. PMLR, 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.

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

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009.

Dongyu Ru, Lin Qiu, Xiangkun Hu, Tianhang Zhang, Peng Shi, Shuaichen Chang, Cheng Jiayang, Cunxiang Wang, Shichao Sun, Huanyu Li, Zizhao Zhang, Binjie Wang, Jiarong Jiang, Tong He, Zhiguo Wang, Pengfei Liu, Yue Zhang, and Zheng Zhang. RAGChecker: A fine-grained framework for diagnosing retrieval-augmented generation. *Advances in Neural Information Processing Systems*, 37:21999–22027, 2024.

Giovanni Trappolini, Florin Cuconasu, Simone Filice, Yoelle Maarek, and Fabrizio Silvestri. Redefining retrieval evaluation in the era of LLMs. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8359–8375, 2026.

Leo Wang, Lihai Yang, Boyu Chen, Kerun Xu, Gongyi Zou, Bo Tang, Feiyu Xiong, Siheng Chen, and Zhiyu Li. Text2Mem: A unified memory operation language for memory operating system. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 2105–2119, 2026.

Orion Weller, Michael Boratko, Iftekhar Naim, and Jinhyuk Lee. On the theoretical limitations of embedding-based retrieval. In *International Conference on Learning Representations*, pp. 11202–11216, 2026.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. LongMemEval: Benchmarking chat assistants on long-term interactive memory. In *International Conference on Learning Representations*, pp. 86809–86836, 2025.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-Mem: Agentic memory for LLM agents. *Advances in Neural Information Processing Systems*, 38:17577–17604, 2025.

Bowen Zhang and Harold Soh. Extract, define, canonicalize: An LLM-based framework for knowledge graph construction. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 9820–9836, 2024.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. MemoryBank: Enhancing large language models with long-term memory. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17):19724–19731, 2024.

Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Bryan Kian Hsiang Low, and Paul Liang. MEM1: Learning to synergize memory and reasoning for efficient long-horizon agents. In *International Conference on Learning Representations*, pp. 58413–58438, 2026.

A LLM STATEMENT

In this work, we used large language models (LLMs) for three purposes. First, `gpt-4.1-mini` was integrated into our experimental framework as an LLM component during system execution. Second, LLMs were used sparingly to identify and retrieve relevant literature; all cited literature was independently reviewed and verified by the authors. Finally, we used LLMs for language polishing, limited to correcting grammar and improving clarity and fluency. The research questions, methodology, experimental design, analysis, and conclusions were developed by the authors. All AI-assisted work was reviewed by the authors, who take full responsibility for the final content of this work. No confidential or personal data were provided to external LLM services.

B EXPERIMENTAL DETAILS

B.1 BENCHMARKS AND EVALUATION METRICS

Benchmarks. Our LoCoMo evaluation (Maharana et al., 2024) covers 1,540 questions across ten multi-session conversations, including single-hop (841), multi-hop (282), temporal (321), and open-domain (96) reasoning. LongMemEval-S (Wu et al., 2025) comprises 500 questions over substantially longer interaction histories, spanning single-session user information (70), single-session assistant information (56), single-session preferences (30), multi-session reasoning (133), knowledge updates (78), and temporal reasoning (133).

Reader-context tokens. Avg. Tokens denotes the average number of context tokens provided to the answer reader per question, counted using `o200k_base`. Lower values indicate a more compact context. Notably, this metric measures only the reader-context size rather than the total system token consumption.

Answer accuracy. LLM-as-a-judge accuracy is the percentage of generated answers judged correct by an LLM evaluator. For KITE, we use `gpt-4.1-mini` as the judge and report accuracy overall and by question category.

Evidence recall. Evidence recall measures the fraction of annotated supporting evidence recovered by retrieval. For a question q, recall is computed as

$$r_q = \frac{|G_q \cap R_q|}{|G_q|}, \qquad |G_q| > 0, \tag{9}$$

where G_q and R_q denote the sets of annotated and retrieved evidence units, respectively. Evidence units correspond to dialogue turns on LoCoMo and conversation sessions on LongMemEval-S. The reported recall is the average r_q across questions with annotated supporting evidence.

B.2 BASELINE EVALUATION SETTINGS

Memory-system comparison. For consistency with recent published evaluations, we base our comparison with Mem0, Zep, MemOS, and EverMemOS on the results consolidated by EverMemOS (Hu et al., 2026a). We follow EverMemOS in using `gpt-4.1-mini` for final answer generation, maintaining alignment with the reference model setting.

Full context. Full Context bypasses memory construction and retrieval by providing the complete conversation history directly to the answer reader. Sessions are presented chronologically, with session dates and speaker labels preserved. No retrieval-specific filtering, summarization, or reranking is applied.

Naive RAG. We implement Naive RAG using the standard vector-store pipeline provided by LlamaIndex[2]. Each dated conversation session is treated as a retrieval unit and is embedded independently using `text-embedding-3-small`. At query time, the question is embedded with the same model, and the highest-scoring sessions by cosine similarity are supplied to the answer reader in chronological order, with session dates and speaker labels preserved. No memory extraction, query expansion, temporal filtering, or neural reranking is applied.

B.3 KITE Implementation Settings

Models and decoding. We use `gpt-4.1-mini` throughout all model-based stages of KITE, including memory extraction, vocabulary consolidation, query compilation, and final answer generation. All model calls use a temperature of zero, except query compilation, which draws one plan at temperature zero and two further candidates at temperature 0.8. The same model and decoding settings are used for both benchmarks.

Initial vocabularies and governed expansion. KITE starts from a small set of broad topic roots for each benchmark. Table 6 lists these roots, the available memory-entry types, and the event facets shared by both benchmarks. Entities, participants, objects, and places are extracted from the conversations rather than predefined.

Table 6: Complete initial vocabularies used to construct the knowledge index.

Vocabulary	Values
LoCoMo topic roots	relationships, family, work and career, education, health, activities and hobbies, arts and creativity, travel, life events, identity and beliefs, possessions, and community
LoCoMo entry types	event, plan, preference, identity, relationship, emotion, opinion, health, possession, and other
LongMemEval-S topic roots	personal life, work and career, health and fitness, travel, shopping and finance, food and cooking, family and relationships, hobbies and entertainment, technical help, writing tasks, education and learning, and home and lifestyle
LongMemEval-S entry types	event, plan, preference, identity, relationship, opinion, task request, knowledge, and other
Shared event facets	trip, move, purchase, sale, injury, illness, recovery, award, competition, performance, graduation, job change, wedding, birth, death, adoption, meeting, celebration, accident, release, rejection, milestone, and first-time event

During memory construction, the extractor reuses the most specific suitable topic and creates a child topic only when needed. A new topic remains provisional until it appears in three sessions across at least two different months, while redundant sibling topics are merged only when repeated reviews agree. LongMemEval-S additionally applies a final pass that moves entries from broad roots to suitable established child topics.

Evidence ranking and channel fusion. A validated retrieval plan first filters the memory entries according to its explicit conditions. The eligible entries are then ordered by a fixed additive score based on their topic, entity, and facet matches, with the contributions summarized in Table 7. The indexed ranking is complemented by a sparse lexical ranking that favors rare matching terms and discounts incidental matches in longer entries. Questions containing an explicit date or month also receive a date-based ranking over entries from the corresponding period. The available rankings are

[2] `https://github.com/run-llama/llama_index`

Table 7: Fixed score contributions used to rank entries selected by a structured retrieval plan.

Matching signal	Score contribution
Exact topic match	3.0
More specific descendant topic	2.0
Broad hierarchical topic match	0.8
Exact entity match	2.0
Matched facet type	1.5
Memory-entry type match	1.0
Participant match	0.5
Confidence adjustment	0.0–0.4
Rare entity of the requested type	up to 3.5
Lexical condition	up to 6.0

combined using reciprocal rank fusion,

$$F(m) = \sum_{c \in \mathcal{R}(m)} \frac{1}{60 + r_c(m)}, \tag{10}$$

where $\mathcal{R}(m)$ denotes the channels that retrieve m and $r_c(m)$ is its zero-based rank in channel c. The fusion policy is fixed across both benchmarks and uses neither dense embeddings nor neural reranking.

Evidence packing and answer generation. After channel fusion, duplicate entries are removed and the remaining items are added in fused rank order until a fixed token budget is reached. The selected material is arranged chronologically and retains dates, speaker labels, short source quotations, and provenance when available. Given the same index and validated plan, this process produces the same supporting material every time. The answer reader sees only the question and the assembled context, without access to the complete conversation history. A bounded follow-up check handles incomplete or weakly supported responses. If the first reading identifies a specific missing fact, the system may run one additional search over the same index. No external information is introduced, and the final response follows each benchmark's rules for answering or abstaining.

C Prompt Templates

The templates below illustrate the main instructions and output formats used by KITE.[3] Placeholders in braces denote inputs supplied at runtime.

C.1 Memory Extraction Prompt

The extraction prompt converts each dated conversation session into source-linked facts with topic assignments and temporal information. LoCoMo emphasizes coverage of the dialogue, while LongMemEval-S prioritizes the user's statements.

Prompt 1: Memory extraction

```
Encode this conversation session into self-contained, atomic
  facts.
SESSION DATE: {date} ({weekday}), {time}
SPEAKERS: {speakers}
TOPIC TAXONOMY (codes ending in ? are provisional): {tree}
KNOWN ENTITIES: {entities}
DIALOG (id|speaker||text):
{dialog}
```

[3]The full prompts used in our experiments are provided in the released code.

Prompt 1: Memory extraction (continued)

```
Rules:
- Cover substantive details without inferring unstated facts.
  Preserve names, counts, amounts, and other askable details.
- Write one self-contained claim per fact. Resolve month and
  year
  references against the session date, but leave weekday
  references
  relative. Record the event time in "t" when known.
- Set "kind" to one of {kinds}. Use "who" for the person or role
  the fact is about, which may differ from the speaker.
  Set "conf" to high, med, or low.
- Assign one to three of the most specific applicable topic
  codes.
  Reuse existing codes when possible. If only a broad root
  applies,
  propose a specific child under that root.
- Record named entities and the dialogue IDs that directly
  support
  each fact in "entities" and "src".

Return JSON only:
{
  "facts": [{
    "content": str, "kind": str,
    "who": str, "t": str | null, "conf": "high|med|low",
    "topics": [str], "entities": [str], "src": [str]
  }],
  "proposals": [{"code": str, "parent": str}],
  "entity_types": {"name": "type"}
}
```

C.2 Topic Vocabulary Consolidation Prompt

The consolidation prompt reviews potentially redundant sibling topics in the governed topic vocabulary. It identifies equivalent topics and selects a canonical topic for merging, while preserving distinctions useful for retrieval.

Prompt 2: Topic vocabulary consolidation

```
You maintain the controlled topic taxonomy for a long-term
conversational memory. For each pair of topic codes, decide
whether they denote exactly the same conversational subject.

Merge only true synonyms, spelling variants, or equivalent
phrasings. Do not merge a broad topic with one of its subtopics,
a whole with one of its parts, or two subjects that are merely
related. Preserve distinctions that would help a later retrieval
request identify the right conversation. When merging, choose
the clearer, more specific code that remains understandable on
its own.

TOPIC PAIRS:
{pairs}

Return JSON only:
{"decisions": [{"i": 0, "merge": true, "winner": str}]}
```

C.3 INDEX-GROUNDED QUERY COMPILATION PROMPT

The compilation prompt maps a question to a plan using the current topic vocabulary and entity index. They allow independent subqueries or dependent stages in which an earlier result narrows a later search.

Prompt 3: Index-grounded query compilation

```
Translate the question into a retrieval plan (JSON).
TOPIC TAXONOMY: {tree}
ENTITIES: {entities}
FACT KINDS: {kinds}
UNITS (id=date): {units}
TODAY: {today}
QUESTION: {question}

Use one to three simple subqueries and prefer existing topic
and entity codes. Include a broad fact grep query using
distinctive word stems. Use named stages when a later search
depends on an earlier result.

Parallel plan:
{
  "queries": [{
    "select": "facts|lines|units",
    "where": {
      "grep": str, "who": [str],
      "topics": [{"code": str, "closure": true}],
      "entities": [str], "kind": [str], "time": [str, str]
    },
    "pipe": [{"op": "head", "n": 5}]
  }],
  "intent": "factual"
}

Staged plan:
{
  "stages": [
    {"name": "anchor", "select": "units",
     "where": {"grep": str},
     "pipe": [{"op": "head", "n": 1}]},
    {"select": "facts",
     "where": {"units": "$anchor.units"}}
  ],
  "intent": "factual"
}
Return JSON only.
```