SUE-MEM: LOGIC-GUIDED OVERLAPPING EVENT STORYLINES FOR LONG-TERM CONVERSATIONAL MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-term dialogue agents must retrieve facts across sessions while distinguishing event relationships from topical similarity. We introduce SUE-Mem, a memory framework that organizes temporally grounded atomic events into overlapping storylines. For each new event, SUE-Mem retrieves candidate storyline summaries and uses an LLM judge with historical event evidence to assess membership. An event may join multiple independently evolving storylines, with milestone-triggered checks identifying those that should split. At query time, a hybrid retriever protects highly ranked semantic candidates and selects additional storylines using semantic, lexical, and exact-anchor scores. On LoCoMo, SUE-Mem achieves absolute gains of 14.53% in Multi-Hop and 5.59% in Overall Judge score over the strongest reported baselines. On LongMemEval, it achieves Overall F1 and Judge scores of 46.35 and 71.32. Ablations support the contributions of overlapping assignment, event-level evidence, and adaptive splitting to long-term conversational question answering.

1 INTRODUCTION

Large language models (LLMs) underpin autonomous agents (Park et al., 2023), yet their fixed context windows limit access to interaction histories across turns, sessions, and time scales (Liu et al., 2024). Truncating or compressing earlier information can cause factual contradictions, inconsistent user states, and omissions of past events. External memory with persistent writing, organization, and retrieval (Lewis et al., 2020) is therefore central to long-term interaction and personalization.

Prior work has explored long-term memory for LLMs from various perspectives (Lu et al., 2023; Wang et al., 2025; Li et al., 2025): MemoryBank (Zhong et al., 2024) hierarchically summarizes histories; MemGPT (Packer et al., 2023) pages memory between context and storage; Mem0 (Chhikara et al., 2025) extracts and retrieves facts with graph-based representations; and A-MEM (Xu et al., 2025) lets memories self-organize in a Zettelkasten manner. With benchmarks such as LoCoMo (Maharana et al., 2024) and LongMemEval (Wu et al., 2025), the focus has shifted from retaining history to organizing event relationships for reasoning across events and time. Recent approaches—RMM (Tan et al., 2025), THEANINE (Ong et al., 2025), StructMem (Xu et al., 2026), and MRAgent (Ji et al., 2026)—further advance this direction through reflective memory management, timeline construction, event structuring, and active memory reconstruction, respectively.

However, existing methods still fall short in representing how a single atomic event advances several independent processes at once. THEANINE (Ong et al., 2025) and StructMem (Xu et al., 2026) organize events into timelines or event-level structures, but they place each event in a single structure, so an event that simultaneously concludes one process and satisfies the prerequisite of another must be duplicated or forced into one thread; the same limitation affects single hierarchical structures (Sarthi et al., 2024; Kang et al., 2025). MRAgent (Ji et al., 2026) reconstructs memory at query time but does not maintain overlapping, independently evolving event boundaries at write time. Moreover, without a principled criterion for when a growing thread should split, long-running storylines drift toward broad life or career themes that are hard to retrieve against. We therefore focus on overlapping storylines whose boundaries are determined by logical relations and reassessed at milestones.

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.

We propose SUE-Mem, a logic-guided event memory framework with session overview (S), user profile (U), and event memory (E) layers. SUE-Mem extracts temporally grounded atomic events and organizes them into storylines around specific events or goals. Candidate storylines are retrieved by summary similarity and verified using fine-grained evidence of temporal progression, causality, elaboration, updates, or coreference. Events may join multiple independently updated storylines, while milestone-triggered splitting limits semantic drift. At query time, semantics-preserving hybrid retrieval protects highly relevant semantic candidates and supplements them with fuzzy IDF-based lexical matching and entity, date, and numerical anchors.

The main contributions of this work are as follows:

- We propose overlapping event storylines in which a single atomic event can participate in multiple independently evolving processes, each maintained as a bounded event or goal with its own summary and evidence, rather than forcing exclusive assignment or broad life/career themes.
- We introduce evidence-augmented logical assignment and milestone-triggered adaptive splitting that jointly decide membership and re-check storyline boundaries, so a shared event advances each process without permanently coupling the two threads.
- We introduce a semantics-preserving hybrid retrieval method that combines lexical features and exact anchors with protected semantic candidates, improving evidence completeness in cross-event reasoning.
- On LoCoMo, SUE-Mem outperforms 12 baselines on 8 of 15 task metrics, including all Multi-Hop metrics and the Overall BLEU and Judge scores, and on LongMemEval it achieves Overall F1 and Judge scores of 46.35 and 71.32.

2 Concept Definitions

2.1 Atomic Events

Long-term conversations contain colloquial expressions, repetition, and facts scattered across turns. SUE-Mem converts incoming dialogue into atomic events that can be stored and retrieved independently. The i-th event is represented as

$$e_i = (id_i, c_i, \tau_i, \mathbf{v}_i), \tag{1}$$

where id_i is its unique identifier, c_i its normalized description, τ_i its real-world occurrence time, and $\mathbf{v}_i = \mathrm{Embed}(c_i)$ its semantic representation.

2.2 Event Storylines

Storing atomic events in isolation makes it difficult to capture how an event evolves over time (Ong et al., 2025; Xu et al., 2026). We therefore organize logically related atomic events into event storylines. A storyline is represented as

$$L_j = (\mathcal{E}_j, s_j, \mathbf{v}_{L_j}), \tag{2}$$

where $\mathcal{E}_j$ contains the atomic events ordered by their actual occurrence times, s_j is a storyline summary generated from all events in the storyline, and $\mathbf{v}_{L_j} = \mathrm{Embed}(s_j)$ is its semantic representation.

3 Methodology

As illustrated in Figure 1, SUE-Mem comprises session-level memory S, user-profile memory U, and event memory E. S summarizes the conversation, U maintains stable identities and preferences for both interlocutors, and E stores atomic events and storylines for factual recall and cross-event reasoning. As session summaries and profiles are well studied (Zhong et al., 2024; Packer et al., 2023; Chhikara et al., 2025; Pan et al., 2025), we use S and U as auxiliary context and focus on storyline construction, evolution, and retrieval in E.

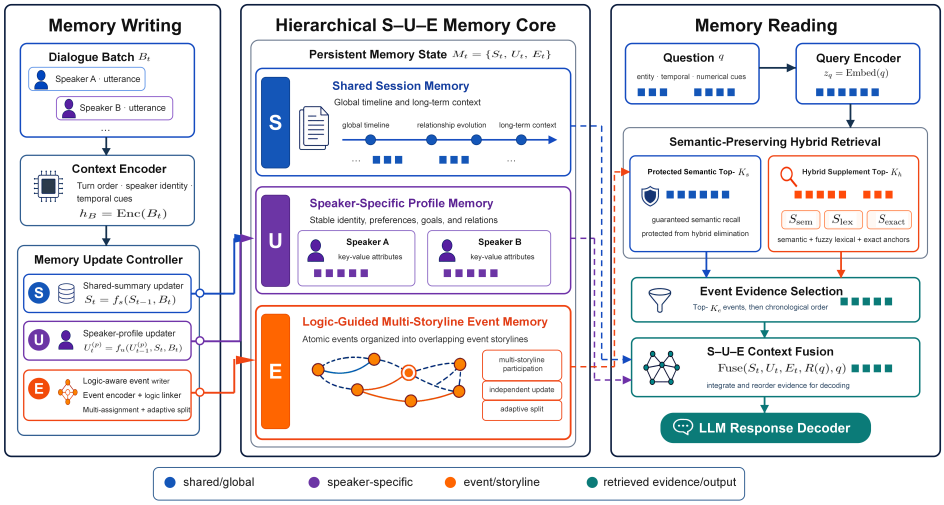


Figure 1: Overview of SUE-Mem. Dialogue batches incrementally update the session, user-profile, and event memories. The event layer organizes atomic events into overlapping storylines through logic-guided assignment and adaptive splitting. During memory reading, the event memory is accessed through semantics-preserving hybrid retrieval, while the session and user memories are directly incorporated during context fusion.

3.1 Incremental Event Memory Construction

The system processes a growing conversation through an overlapping buffer. When $2K_t$ dialogue turns have accumulated, it processes the earliest K_t and retains the rest as overlap for the next batch. This reduces frequent LLM calls while preserving context across batch boundaries.

Given the current batch B, the event extractor generates a set of new atomic events:

$$\mathcal{E}^{\mathrm{new}} = F_{\mathrm{ext}}(B). \tag{3}$$

During extraction, the system jointly performs coreference resolution, relative-time normalization, and factual canonicalization. For example, a conversation may first state that "Alice submitted that oil painting of a harbor last month" and later mention that "it subsequently won first prize." The latter statement is resolved into the shared event $e^\star$: "On August 18, 2024, Alice's *Harbor at Dusk* won the City Art Competition."

The system then assigns the event a unique identifier and computes its semantic embedding. The S and U layers are updated in parallel using the same dialogue batch. Appendices A.1 and A.2 provide the persistent memory state and the complete incremental write procedure, respectively.

3.2 Evidence-Augmented Logical Storyline Assignment

To avoid judging every historical storyline for a new event e, SUE-Mem retrieves the top K_a candidates by cosine similarity between the event and storyline-summary embeddings:

$$\mathcal{L}_e = \mathrm{TopK}_{K_a}\left(\cos(\mathbf{v}_e, \mathbf{v}_{L_j})\right). \tag{4}$$

No fixed similarity threshold is imposed: similarity narrows the candidate space but does not determine membership. For example, the candidate pool for $e^\star$ may contain L_1, "The complete competition workflow for Alice's participation in the City Art Competition"; L_2, "Alice's award-based scholarship eligibility verification"; and L_3, "Alice's Visit to the City Art Museum," which is topically similar but logically unrelated.

A storyline summary is a compressed representation of a long-running event process and may omit specific dates, artwork titles, or state transitions. To provide verifiable evidence for the assignment decision, the system selects the K_v historical events most relevant to the new event from each candidate storyline, $\mathrm{Evidence}(e, L_j) = \mathrm{TopK}_{K_v}(\cos(\mathbf{v}_e, \mathbf{v}_{e_{j,k}}))$. A logical judge then performs a binary

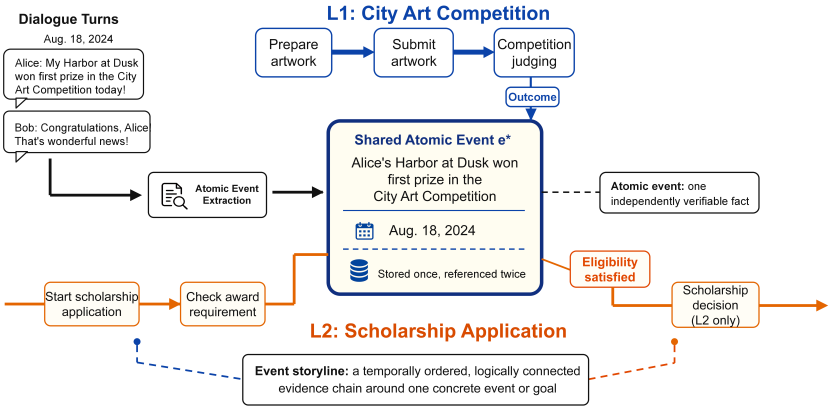


Figure 2: Illustration of an atomic event shared across two independently evolving event storylines. The event $e^\star$ is stored once and referenced by both the painting-competition storyline and the scholarship-application storyline, where it plays different logical roles.

decision using the new event, the storyline summary, and the retrieved fine-grained evidence:

$$J(e, L_j) = F_{\text{judge}}\left(e, s_j, \text{Evidence}(e, L_j)\right) \in \{0, 1\}. \tag{5}$$

The judge returns $J(e, L_j) = 1$ only when the new event and the storyline concern the same concrete event or goal and exhibit at least one explicit logical relation, such as temporal progression, causality, information supplementation, factual updating, state transition, or coreference. Sharing the same person, broad topic, overlapping keywords, or a nearby time period is insufficient on its own to establish membership.

In the running example, $e^\star$ completes the "submission–judging–award" sequence of L_1 (temporal-progression and outcome) and satisfies the award-eligibility requirement of L_2 (prerequisite–result), whereas L_3 is rejected because its shared terms ("Alice," "city," "art") carry no event-identity, causal, or prerequisite relation. Appendix B examines how storyline organization supports cross-session evidence integration and temporal disambiguation in LoCoMo.

3.3 Multi-Storyline Participation and Independent Evolution

The storylines matched by a new event form the set $\mathcal{A}(e) = \{L_j \in \mathcal{L}_e \mid J(e, L_j) = 1\}$. When $\mathcal{A}(e) = \emptyset$, the system initializes a new single-event storyline. When an event matches one or more existing storylines, the same event identifier is appended to every matched storyline without forcing the event into a unique assignment.

In the running example, $\mathcal{A}(e^\star) = \{L_1, L_2\}$: as illustrated in Figure 2, $e^\star$ participates in both the City Art Competition and scholarship-application storylines, updating each to $\mathcal{E}_1' = \text{Unique}(\mathcal{E}_1 \cup \{e^\star\})$ and $\mathcal{E}_2' = \text{Unique}(\mathcal{E}_2 \cup \{e^\star\})$. The system then retrieves all events from each updated storyline, orders them by their actual occurrence times, and independently reconstructs the corresponding storyline summary:

$$s_j' = F_{\text{int}}\left(\text{ChronSort}(\mathcal{E}_j')\right), \qquad \mathbf{v}_{L_j}' = \text{Embed}(s_j'). \tag{6}$$

Consequently, the summaries of L_1 and L_2 may respectively become "Alice entered the City Art Competition with *Harbor at Dusk* and won first prize" and "Alice's competition award satisfied the scholarship eligibility requirement." The same event thus serves different logical roles in different storylines while preserving a consistent underlying fact.

Independent updates keep later developments specific to each storyline: a scholarship decision may extend L_2 without altering the competition outcome in L_1. Each new event is therefore judged against each candidate independently; sharing $e^\star$ does not permanently couple the two storylines.

3.4 MILESTONE-TRIGGERED STORYLINE SPLITTING

As a storyline evolves over an extended period, it may gradually absorb multiple event processes that are only weakly related. To limit this drift, SUE-Mem employs milestone-triggered adaptive splitting rather than imposing a fixed maximum length.

Let R denote the splitting-check interval, and let n_{old} and n_{new} be the numbers of unique events before and after a storyline update, respectively. A splitting check is triggered only when the updated storyline crosses a new milestone:

$$\left\lfloor \frac{n_{\text{new}}}{R} \right\rfloor > \left\lfloor \frac{n_{\text{old}}}{R} \right\rfloor . \tag{7}$$

Crossing a milestone does not necessarily cause the storyline to be split. The splitting judge examines all atomic events in the storyline and creates new storylines only when it identifies two or more logically independent event processes. This strategy preserves genuine long-range event evolution while avoiding the cost of invoking an LLM-based splitting check after every memory update. The deterministic validation invariants, retry policy, and failure-safe update order are detailed in Appendix A.3.

3.5 SEMANTICS-PRESERVING HYBRID RETRIEVAL

After constructing the storylines, the system must recover a compact yet sufficiently complete evidence context for answering a question (Fang et al., 2026). Pure semantic retrieval finds paraphrases but may overlook artwork titles, dates, and numerical details (Karpukhin et al., 2020), while lexical matching (Robertson & Zaragoza, 2009) finds exact facts but misses implicit relations. Global hybrid reranking may also displace semantically relevant storylines by keyword-overlapping ones. We therefore adopt a two-stage strategy of protected semantic results and hybrid supplementary results.

Given a question q, the system first retrieves the top K_c candidate storylines according to the similarity between the question embedding and each storyline-summary embedding:

$$\mathcal{L}_q = \text{TopK}_{K_c} \left(S_{\text{sem}}(q, L_j) \right). \tag{8}$$

For every atomic event in the candidate storylines, the system then computes a fuzzy IDF-based lexical score $S_{\text{lex}}(q, e)$ and an exact-match score $S_{\text{exact}}(q, e)$ based on entity, date, and numerical anchors. The event-level relevance score is defined as

$$S_{\text{event}}(q, e) = w_l S_{\text{lex}}(q, e) + w_x S_{\text{exact}}(q, e). \tag{9}$$

Only the top K_e most relevant atomic events are retained from each storyline. The final hybrid score of a storyline combines summary-level semantic relevance, event-level lexical matching, and exact-anchor matching:

$$S_{\text{hybrid}}(q, L_j) = \frac{w_s S_{\text{sem}}(q, L_j) + w_l S_{\text{lex}}(q, L_j) + w_x S_{\text{exact}}(q, L_j)}{w_s + w_l + w_x}. \tag{10}$$

The top K_s storylines from the initial semantic ranking are retained directly as $\mathcal{P}_s = \mathcal{L}_q[1 : K_s]$, and the top K_h supplementary storylines are selected from the remaining candidates by their hybrid scores, $\mathcal{P}_h = \text{TopK}_{K_h} \left(\{ S_{\text{hybrid}}(q, L) \mid L \in \mathcal{L}_q \setminus \mathcal{P}_s \} \right)$. The final retrieval result is

$$\boxed{\mathcal{R}(q) = \mathcal{P}_s \| \mathcal{P}_h}, \tag{11}$$

where $\|$ denotes ordered concatenation. In other words, the final evidence consists of the protected semantic Top-K_s storylines and the hybrid supplementary Top-K_h storylines.

Within each selected storyline, atomic events are first selected according to relevance and subsequently ordered by their actual occurrence times. The final context combines the shared session summary, the profiles of both interlocutors, and the retrieved storyline evidence:

$$\mathcal{C}_q = S \oplus U^A \oplus U^B \oplus \left\{ (s_j, \widetilde{\mathcal{E}}_j) \mid L_j \in \mathcal{R}(q) \right\}. \tag{12}$$

This design decouples the amount of context supplied to the answer-generating model from the continuously growing size of the complete long-term memory. Appendix A.4 gives the exact fuzzy matching, IDF weighting, anchor extraction, event aggregation, and tie-breaking rules, while Appendix F provides the prompt specifications used throughout memory construction and question answering.

4 EXPERIMENTS

We evaluate SUE-Mem from six perspectives: overall effectiveness on LoCoMo, generalization to LongMemEval, computational cost, component contribution, the effect of semantic protection, and sensitivity to the number of retrieved storylines.

4.1 EXPERIMENTAL SETUP

Dataset We evaluate SUE-Mem on LoCoMo (Maharana et al., 2024), a benchmark for long-term conversational memory across multiple sessions. Its question-answering task contains four categories: Single-Hop, Multi-Hop, Temporal, and Open-Domain.

Metrics We report token-level F1, unigram BLEU (BLEU-1) (Papineni et al., 2002), and an LLM-as-a-Judge score (Zheng et al., 2023) denoted by J. All metrics are scaled to $[0, 100]$, and higher values indicate better performance. A dash indicates that the corresponding result was not reported by the baseline.

Baselines We compare against 12 representative systems: LoCoMo (Maharana et al., 2024), ReadAgent (Lee et al., 2024), MemoryBank (Zhong et al., 2024), MemGPT (Packer et al., 2023), LangMem (LangChain Team, 2025), Zep (Rasmussen et al., 2025), OpenAI Memory (OpenAI, 2024b), Mem0 (Chhikara et al., 2025), Mem0g (Chhikara et al., 2025), A-MEM (Xu et al., 2025), StructMem (Xu et al., 2026), and MRAgent (Ji et al., 2026). Together, they cover full-history processing, flat retrieval, hierarchical summarization, graph memory, evolving memory, and active memory reconstruction.

Implementation details All LLM-based components use GPT-4o-mini (OpenAI, 2024a), and text-embedding-3-small is used for all semantic representations. The temperature is set to 0. Unless otherwise specified, we use $K_t = 10$, $K_a = 10$, $K_v = 5$, $R = 10$, $K_c = 20$, $K_s = 3$, $K_h = 9$, and $K_e = 5$, with $(w_s, w_l, w_x) = (0.55, 0.30, 0.15)$. Appendix C reports the complete data-processing and evaluation protocol, parameter-to-configuration mapping, and persistence and audit settings required for reproduction.

4.2 MAIN RESULTS

Table 1: Main results on LoCoMo. All values are percentages. The best result in each column is shown in bold, and the second-best result is underlined. A dash denotes an unavailable result.

	Multi-Hop			Temporal			Open-Domain			Single-Hop			Overall		
Method	F1	B-1	J	F1	B-1	J	F1	B-1	J	F1	B-1	J	F1	B-1	J
LoCoMo	12.04	11.16	–	18.41	14.77	–	40.36	29.05	–	25.02	19.75	–	22.22	17.72	–
ReadAgent	5.31	5.12	–	12.60	8.87	–	9.67	7.66	–	9.15	6.48	–	9.20	6.80	–
MemoryBank	5.31	5.12	–	9.68	6.99	–	6.61	5.16	–	5.00	4.77	–	6.13	5.32	–
MemGPT	9.15	7.44	–	25.52	19.44	–	41.04	34.34	–	26.65	17.72	–	24.11	17.23	–
LangMem	26.04	22.32	47.91	30.75	25.84	23.43	40.91	33.63	71.12	35.41	26.86	62.23	33.07	26.24	52.07
Zep	19.37	14.82	41.35	42.00	34.53	49.31	**49.56**	38.92	**76.60**	35.74	23.30	61.70	34.91	25.05	56.32
OpenAI	20.09	15.42	42.92	14.04	11.25	21.71	39.31	31.16	62.29	34.30	23.72	63.79	27.79	20.06	51.10
Mem0	28.64	21.58	51.15	48.93	40.51	55.51	47.64	38.72	72.93	38.72	27.13	67.13	39.56	29.63	62.14
Mem0g	24.32	18.82	47.19	51.55	40.28	58.13	49.27	**40.30**	75.71	38.09	26.03	65.71	39.07	28.57	61.36
A-MEM	27.02	20.09	–	45.85	36.67	–	12.14	12.00	–	44.65	37.06	–	39.65	32.31	–
StructMem	43.71	40.34	68.09	61.47	61.73	**78.82**	24.78	29.90	46.88	56.31	54.44	80.83	**53.11**	51.84	75.97
MRAgent	36.67	36.43	62.41	55.39	57.69	65.42	28.97	34.85	42.70	**58.60**	**57.86**	80.12	45.11	46.29	71.42
SUE-Mem	**44.21**	**42.30**	**82.62**	**61.88**	**63.77**	75.39	21.57	29.91	57.29	54.13	53.67	**86.32**	51.90	**52.21**	**81.56**

As shown in Table 1, SUE-Mem leads 8 of the 15 category–metric combinations, with the largest gains on Multi-Hop questions (44.21 F1, 42.30 BLEU-1, and 82.62 J, exceeding the strongest baseline by up to 14.53 points). It also ranks first on Temporal F1 and BLEU-1 (61.88 and 63.77) and on Overall BLEU-1 and J (52.21 and 81.56), and attains the best Single-Hop J (86.32). Its weaker Open-Domain results suggest that the main benefits concern questions grounded in conversation history. Appendix B compares SUE-Mem, StructMem, and MRAgent on cross-session and temporal cases.

4.3 RESULTS ON LONGMEMEVAL

We further evaluate SUE-Mem on LongMemEval (Wu et al., 2025), a benchmark for long-term interactive memory that stresses cross-session factual recall and consistency over extended chat histories. Table 2 reports the overall results.

Table 2: Overall results on LongMemEval. All values are percentages. The best result in each column is shown in bold, and the second-best result is underlined.

Method	Overall F1	Overall B-1	Overall J
SUE-Mem	**46.35**	<u>42.47</u>	**71.32**
MRAgent	<u>45.15</u>	**43.28**	65.87
StructMem	42.26	40.98	<u>67.42</u>
A-MEM	30.78	32.45	46.29
Mem0	33.56	34.89	47.27

SUE-Mem achieves the highest Overall F1 (46.35) and Judge (71.32) scores (Table 2), exceeding MRAgent by 1.20 and 5.45 points. MRAgent leads BLEU-1 (43.28). These results support the effectiveness of logic-guided overlapping storylines on a second long-term memory benchmark.

4.4 COMPUTATIONAL COST

We compare the computational cost of SUE-Mem with MRAgent and StructMem under the same model and API configuration, reporting the average tokens per question, the P50/P95 retrieval and end-to-end latencies, and the Overall Judge (J) score. For this comparison, SUE-Mem uses a reduced retrieval budget of $K_s = 1$ and $K_h = 3$. A dash indicates that the corresponding result is unavailable.

Table 3: Computational cost and answer quality on LoCoMo. Tokens denote the average consumption per question, latency is measured in seconds, and Overall J is the Judge score under each system's efficiency configuration. The best result in each column is shown in bold.

		Retrieval latency (s)		End-to-end latency (s)		
Method	Tokens/query	P50	P95	P50	P95	Overall J
SUE-Mem	**1,869.52**	**0.99**	**1.62**	**2.43**	**3.31**	**77.60**
MRAgent	14,017.38	–	–	17.21	107.34	71.42
StructMem	4,177.41	2.04	2.61	3.72	4.89	75.97

As shown in Table 3, SUE-Mem consumes 1,869.52 tokens per question (86.66% and 55.25% less than MRAgent and StructMem) and attains the lowest P50/P95 latencies for both retrieval and the end-to-end pipeline. Even with the reduced budget $K_s = 1, K_h = 3$, SUE-Mem still achieves the highest Overall J (77.60 vs. 75.97 for StructMem and 71.42 for MRAgent), so the reduced query-time cost is accompanied by competitive answer quality.

4.5 ABLATION STUDY

We conduct ablation experiments to evaluate three core components of the event memory layer: multi-storyline assignment, internal event evidence, and adaptive storyline splitting. Each variant removes only the indicated component while keeping all other settings unchanged.

Table 4: Ablation results on LoCoMo. B-1 denotes BLEU-1. The best result in each column is shown in bold.

Variant	Overall F1	Overall B-1	Overall J
SUE-Mem	**51.90**	**52.21**	**81.56**
w/o multi-storyline assignment	45.91	47.55	73.98
w/o internal event evidence	48.87	50.49	78.30
w/o adaptive splitting	49.18	50.08	77.27

Removing any component reduces all Overall metrics (Table 4). Removing multi-storyline assignment has the largest effect: F1, BLEU-1, and J fall by 5.99, 4.66, and 7.58 points. The decreases without internal event evidence or adaptive splitting also support their contributions. Category-level results appear in Appendix D.1.

4.6 Effect of Semantics-Preserving Hybrid Retrieval

To investigate how semantic protection interacts with hybrid retrieval, we fix the final retrieval budget to $K_r = 12$ and vary the proportion of storylines selected by hybrid ranking from 0% (pure semantic) to 100% (no semantic protection). Intermediate settings preserve the highest-ranked semantic candidates and fill the remaining budget with hybrid results.

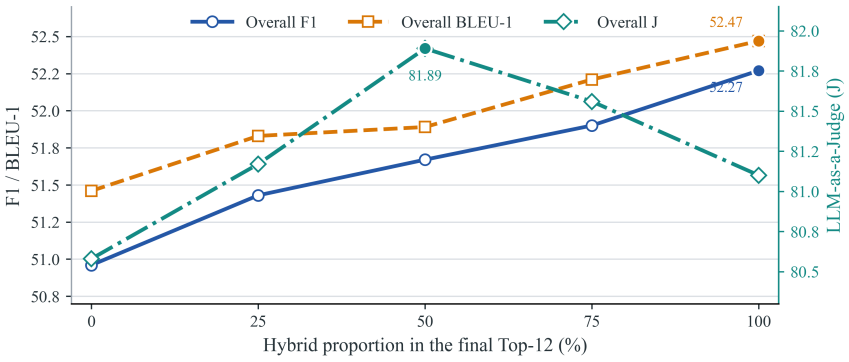


Figure 3: Effect of the hybrid-retrieval proportion in the final Top-12 storylines. Increasing the hybrid proportion improves F1 and BLEU-1, while the Judge score exhibits a non-monotonic trend. The 75% setting provides the best overall balance across the three metrics.

As shown in Figure 3, pure semantic retrieval attains 50.96 F1, 51.46 BLEU-1, and 80.58 J, and adding hybrid-retrieved storylines improves all three. Fully hybrid retrieval reaches the highest F1 and BLEU-1 (52.27 and 52.47) but lowers J, indicating that lexical and exact signals should supplement, not replace, semantic retrieval.

We adopt the 75% setting (three protected semantic storylines and nine hybrid results), balancing F1 (51.90), BLEU-1 (52.21), and J (81.56). Additional sensitivity analyses appear in Appendices E.1 and E.2.

4.7 Effect of the Number of Retrieved Storylines

We investigate the effect of the final retrieval budget $K_r = K_s + K_h$, where K_s and K_h denote the numbers of protected semantic and hybrid supplementary storylines, respectively. We vary K_r over $\{4, 8, 12, 16, 20, 30, 40, 50\}$. Since the final result is drawn from the K_c first-stage candidates, we keep $K_c \geq K_r$, raising K_c to match K_r whenever $K_r > 20$; all other settings are unchanged.

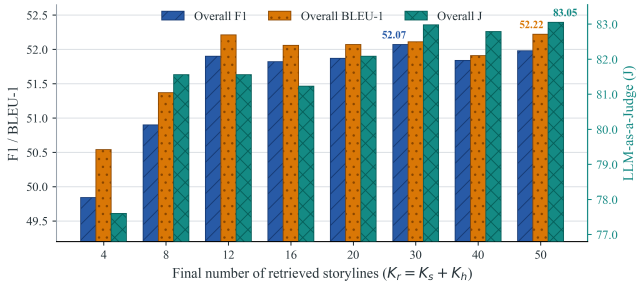


Figure 4: Effect of the final number of retrieved storylines $K_r = K_s + K_h$ on LoCoMo. Performance improves rapidly as the retrieval budget initially increases, and then exhibits diminishing gains and minor fluctuations at larger values of K_r.

Increasing K_r from 4 to 8 raises F1 from 49.84 to 50.90, BLEU-1 from 50.54 to 51.37, and J from 77.60 to 81.56 (Figure 4), consistent with improved coverage of complementary evidence. Gains largely saturate beyond $K_r = 12$, although J peaks at 83.05 for $K_r = 50$. Dips at $K_r = 16$ and 40 suggest that additional storylines can introduce redundancy. We use $K_r = 12$ to balance answer quality and context size. Appendix D.2 reports the category-level results.

5 Conclusions

We presented SUE-Mem, which organizes atomic events into overlapping, independently evolving storylines through explicit logical relations. Evidence-augmented assignment and milestone-triggered splitting maintain storyline boundaries, while protected semantic retrieval and lexical and exact-anchor matching supply query-time evidence. SUE-Mem leads 8 of 15 LoCoMo metrics, including all Multi-Hop metrics and Overall BLEU-1 and Judge, and reduces question-time tokens and latency in the efficiency comparison. Ablation and sensitivity analyses support the value of these components for cross-session reasoning.

6 Limitations

First, evaluation covers LoCoMo and LongMemEval but remains limited to two-party conversations. Second, event extraction, logical assignment, summary updating, and splitting depend on LLM judgments, so errors may accumulate and vary across models. Third, logic-based construction adds write-time LLM calls despite reducing query-time cost. Finally, performance remains weaker on Open-Domain questions, motivating controlled external retrieval.

Reproducibility statement

Appendix C documents the data-processing and evaluation protocol, model and embedding configuration, hyperparameters, persistent state, and audit settings. Appendix A provides the complete memory write and retrieval procedures. Appendix F summarizes the interfaces of all LLM-based components and presents representative templates for atomic-event extraction, storyline splitting, and answer generation. Anonymous source code containing the complete literal prompts, configuration files, and reproduction instructions is provided in the supplementary material.

AI use statement

In this work, we used generative AI tools to assist with the literature survey and with polishing and editing the manuscript. The proposed method, experimental design, implementation, and all reported results were produced by the authors, and no generative AI tools were used to generate or alter experimental results. The authors reviewed all AI-assisted content and take full responsibility for the final text, claims, and artifacts of this paper.

References

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready AI agents with scalable long-term memory. In *Proceedings of the 28th European Conference on Artificial Intelligence*, volume 413 of *Frontiers in Artificial Intelligence and Applications*, pp. 2993–3000. IOS Press, 2025. doi: 10.3233/FAIA251160. URL https://doi.org/10.3233/FAIA251160.

Jizhan Fang, Xinle Deng, Haoming Xu, Ziyan Jiang, Yuqi Tang, Ziwen Xu, Shumin Deng, Yunzhi Yao, Mengru Wang, Shuofei Qiao, Huajun Chen, and Ningyu Zhang. LightMem: Lightweight and efficient memory-augmented generation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/a05b72653ec5b473732129829ae04195-Abstract-Conference.html.

Shuo Ji, Yibo Li, and Bryan Hooi. Memory is reconstructed, not retrieved: Graph memory for LLM agents. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306

of *Proceedings of Machine Learning Research*. PMLR, 2026. URL https://openreview.net/forum?id=xRVWftS3ES.

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. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.1318. URL https://aclanthology.org/2025.emnlp-main.1318/.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pp. 6769–6781. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.550. URL https://aclanthology.org/2020.emnlp-main.550/.

LangChain Team. LangMem SDK for agent long-term memory, 2025. URL https://www.langchain.com/blog/langmem-sdk-launch.

Kuang-Huei Lee, Xinyun Chen, Hiroki Furuta, John Canny, and Ian Fischer. A human-inspired reading agent with gist memory of very long contexts. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 26396–26415. PMLR, 2024. URL https://proceedings.mlr.press/v235/lee24c.html.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474, 2020. URL https://papers.nips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

Hao Li, Chenghao Yang, An Zhang, Yang Deng, Xiang Wang, and Tat-Seng Chua. Hello again! LLM-powered personalized agent for long-term dialogue. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 5259–5276. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.naacl-long.272. URL https://aclanthology.org/2025.naacl-long.272/.

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. doi: 10.1162/tacl_a_00638. URL https://doi.org/10.1162/tacl_a_00638.

Junru Lu, Siyu An, Mingbao Lin, Gabriele Pergola, Yulan He, Di Yin, Xing Sun, and Yunsheng Wu. MemoChat: Tuning LLMs to use memos for consistent long-range open-domain conversation, 2023. URL https://arxiv.org/abs/2308.08239.

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. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.747. URL https://aclanthology.org/2024.acl-long.747/.

Kai Tzu-iunn Ong, Namyoung Kim, Minju Gwak, Hyungjoo Chae, Taeyoon Kwon, Yohan Jo, Seung-won Hwang, Dongha Lee, and Jinyoung Yeo. Towards lifelong dialogue agents via timeline-based memory management. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 8631–8661. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.naacl-long.435. URL https://aclanthology.org/2025.naacl-long.435/.

OpenAI. GPT-4o mini: Advancing cost-efficient intelligence, 2024a. URL https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/.

OpenAI. Memory and new controls for ChatGPT, 2024b. URL https://openai.com/index/memory-and-new-controls-for-chatgpt/.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems, 2023. URL https://arxiv.org/abs/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. SeCom: On memory construction and retrieval for personalized conversational agents. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/e56f394bbd4f0ec81393d767caa5a31b-Abstract-Conference.html.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318. Association for Computational Linguistics, 2002. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040/.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pp. 1–22. Association for Computing Machinery, 2023. doi: 10.1145/3586183.3606763. URL https://doi.org/10.1145/3586183.3606763.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. Zep: A temporal knowledge graph architecture for agent memory, 2025. URL https://arxiv.org/abs/2501.13956.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009. doi: 10.1561/1500000019. URL https://doi.org/10.1561/1500000019.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. RAPTOR: Recursive abstractive processing for tree-organized retrieval. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/8a2acd174940dbca361a6398a4f9df91-Abstract-Conference.html.

Zhen Tan, Jun Yan, I-Hung Hsu, Rujun Han, Zifeng Wang, Long T. Le, Yiwen Song, Yanfei Chen, Hamid Palangi, George Lee, Anand Iyer, Tianlong Chen, Huan Liu, Chen-Yu Lee, and Tomas Pfister. In prospect and retrospect: Reflective memory management for long-term personalized dialogue agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8416–8439. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.413. URL https://aclanthology.org/2025.acl-long.413/.

Qingyue Wang, Yanhe Fu, Yanan Cao, Shuai Wang, Zhiliang Tian, and Liang Ding. Recursively summarizing enables long-term dialogue memory in large language models. *Neurocomputing*, 639:130193, 2025. doi: 10.1016/j.neucom.2025.130193. URL https://doi.org/10.1016/j.neucom.2025.130193.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. LongMemEval: Benchmarking chat assistants on long-term interactive memory. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/d813d324dbf0598bbdc9c8e79740ed01-Abstract-Conference.html.

Buqiang Xu, Yijun Chen, Jizhan Fang, Ruobin Zhong, Yunzhi Yao, Yuqi Zhu, Lun Du, and Shumin Deng. StructMem: Structured memory for long-horizon behavior in LLMs. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 122–146. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-short.12. URL https://aclanthology.org/2026.acl-short.12/.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-MEM: Agentic memory for LLM agents. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-0593. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/19909c36f51abc4856b4560aff3d36d6-Abstract-Conference.html.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems*, volume 36, pp. 46595–46623, 2023. doi: 10.52202/075280-2020.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. MemoryBank: Enhancing large language models with long-term memory. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence*, volume 38, pp. 19724–19731. AAAI Press, 2024. doi: 10.1609/aaai.v38i17.29946. URL https://ojs.aaai.org/index.php/AAAI/article/view/29946.

A Extended Algorithmic Details

This appendix supplements the method in Section 3 with execution details that are required for reproducibility but omitted from the main text for space. We describe the implemented memory state, the complete write path, the deterministic safeguards used during storyline splitting, and the exact collection-level aggregation used by hybrid retrieval. The notation follows the main paper; an event collection in the implementation corresponds to an event storyline L_j.

A.1 Incremental Memory State

For each two-party conversation, SUE-Mem maintains the state $\mathcal{M} = (S, U^A, U^B, \mathcal{E}, \mathcal{L})$. The raw dialogue is ingested in chronological order. Once $2K_t$ unprocessed turns are available, the earliest K_t turns form the next update batch and the remaining K_t turns provide overlap with the following batch. A final flush processes the residual turns at the end of a conversation. The three memory layers are updated from the same batch, but only the event layer is structurally retrieved at question time.

Table 5: Persistent state used by SUE-Mem. The Indexed column indicates the object stored in the first-stage vector index.

State	Persistent content	Update unit	Indexed
S	shared session summary	dialogue batch	no
U^A, U^B	stable speaker attributes	dialogue batch	no
e_i	identifier, content, occurrence time	atomic event	no
L_j	event identifiers, summary, embedding	storyline	yes

Atomic events are stored once. Multi-storyline participation is implemented by placing the same event identifier in multiple storyline membership lists, rather than duplicating its content. Consequently, updating or splitting one storyline does not alter the underlying event or the membership of any other storyline.

A.2 Logic-Guided Storyline Writer

The procedure in Table 6 gives the complete write path for one dialogue batch. Semantic similarity is used only to obtain K_a candidates. Every returned candidate is passed to the logical judge; no similarity threshold determines membership. The judge sees both the storyline summary and the K_v event-level evidence items most similar to the new event. All positive decisions are retained, which realizes multi-storyline participation.

A positive link requires (i) a specific shared anchor, (ii) an explicit relationship around that anchor, and (iii) preservation of one coherent storyline. The allowed relationship types are chronological

Table 6: Logic-guided incremental storyline construction.

Input: Dialogue batch B, memory state $\mathcal{M}$, budgets K_a and K_v
Output: Updated memory state $\mathcal{M}$

$S \leftarrow \textsc{UpdateSessionSummary}(S, B)$
$U^A, U^B \leftarrow \textsc{UpdateProfiles}(U^A, U^B, S, B)$
$\mathcal{E}^{\text{new}} \leftarrow \textsc{ExtractAtomicEvents}(B)$
for all $e \in \mathcal{E}^{\text{new}}$ **do**
    $\mathbf{v}_e \leftarrow \textsc{Embed}(e.\text{content})$
    $\mathcal{C} \leftarrow \textsc{SemanticTopK}(\mathbf{v}_e, \{\mathbf{v}_{L_j}\}, K_a)$
    matched $\leftarrow [\,]$
    for all $L_j \in \mathcal{C}$ **do**
        $V_j \leftarrow \textsc{EventTopK}(\mathbf{v}_e, \text{Events}(L_j), K_v)$
        $d \leftarrow \textsc{LogicJudge}(e, \text{Summary}(L_j), V_j)$
        if $d.\texttt{has_logic_link} = \text{true}$ **then**
            Append L_j to matched
    if matched is empty **then**
        Create a single-event storyline containing $\text{id}(e)$
    else
        Store e once
        for all $L_j \in$ matched **do**
            $\mathcal{E}_j \leftarrow \text{Unique}(\mathcal{E}_j \cup \{\text{id}(e)\})$
            $\textsc{RebuildSummaryAndEmbedding}(L_j)$
            if a new split milestone is crossed **then**
                $\textsc{ValidateAndPossiblySplit}(L_j)$

progression, causality, prerequisite–result, complementary information, corrective update, same-episode derivation, same-entity state transition, and explicit coreference. A shared speaker, broad topic, keyword, emotion, or nearby timestamp is insufficient. If no candidate passes this test, a new single-event storyline is created. After each accepted assignment, all events in that storyline are sorted by `occurred_at`; its summary and embedding are regenerated from the complete ordered sequence.

A.3 Milestone Checks and Deterministic Split Validation

A split check is triggered when an update moves a storyline from one interval of R unique events to the next. The interval is a check schedule, not a maximum storyline length. The LLM receives all child events because a Top-K subset cannot establish complete partition coverage. Its output is accepted only after the deterministic checks in Table 7.

Table 7: Deterministic invariants applied to a proposed storyline split.

Invariant	Enforced behavior
Schema	The response contains exactly `should_split` and `groups`; each group contains only a nonempty `eventsId` list.
Identifier validity	Every returned identifier must be copied from the original storyline; event text and invented identifiers are rejected.
Nondegeneracy	A valid split has at least two distinct groups; no group may reproduce the complete original storyline.
Coverage	Every original event must be represented. An omitted valid event is conservatively retained as a singleton storyline.
Controlled overlap	The same event may occur in several groups only when it has a direct factual role in each; duplicate groups are rejected.
Failure safety	Invalid output is retried at most twice with the validation error. After three invalid responses, the original storyline is preserved and processing continues.

New storyline summaries and embeddings are prepared before the original storyline is removed. The replacement is committed only after every proposed group has been successfully integrated. Atomic-event records remain intact, which preserves shared events referenced by storylines not involved in the split.

A.4 Hybrid Retrieval Scoring

Let $\mathcal{D}_q$ be all atomic events contained in the K_c semantically retrieved candidate storylines. Stop words are removed from lexical matching, and the two interlocutor names are excluded from third-party entity anchors. Tokens are case-normalized and lightly stemmed. For query token t and event token u, the fuzzy match function is

$$m(t,u) = \begin{cases} 1, & \mathrm{canon}(t) = \mathrm{canon}(u), \\ 0.90, & d_{\mathrm{DL}}(\mathrm{canon}(t), \mathrm{canon}(u)) \le 1, \\ 0.86, & \text{one canonical token is a length-}\ge 4 \text{ prefix of the other}, \\ 0, & \text{otherwise}, \end{cases} \tag{13}$$

where d_{DL} is Damerau–Levenshtein distance and the edit-distance rule is applied only when the longer token has at least five characters. With $N = |\mathcal{D}_q|$ and event-level document frequency $\mathrm{df}(t)$, the lexical score is

$$S_{\mathrm{lex}}(q,e) = \frac{\sum_{t \in \mathrm{uniq}(q)} \left[\log \frac{N+1}{\mathrm{df}(t)+1} + 1\right] \max_{u \in e} m(t,u)}{\sum_{t \in \mathrm{uniq}(q)} \left[\log \frac{N+1}{\mathrm{df}(t)+1} + 1\right]}. \tag{14}$$

Exact anchors include ISO dates, year–month expressions, years, month names, non-year numbers, and third-party named entities. Date and number anchors use fixed type weights, whereas entity anchors use event-level IDF. If $\mathcal{A}_q$ is the extracted anchor set and $\mathcal{A}_e$ the anchors matched by event e, the implemented exact score is

$$S_{\mathrm{exact}}(q,e) = \frac{\sum_{a \in \mathcal{A}_q \cap \mathcal{A}_e} w_a}{\sum_{a \in \mathcal{A}_q} w_a} \left(1 - \exp\left[-\frac{\sum_{a \in \mathcal{A}_q \cap \mathcal{A}_e} w_a}{3}\right]\right). \tag{15}$$

This saturation term prevents a single common exact match from immediately receiving the maximum score.

Within each candidate storyline, events are ranked by $w_l S_{\mathrm{lex}} + w_x S_{\mathrm{exact}}$, and only the top K_e are retained. For either event-level signal z, the storyline aggregation is $0.7\max(z) + 0.3\,\mathrm{mean}(z)$ over the retained events. Chroma cosine distance d_j is converted to $S_{\mathrm{sem}}(q, L_j) = \mathrm{clip}(1 - d_j/2, 0, 1)$ before applying the hybrid score in Section 3.5. The original semantic Top-K_s is copied to the output unchanged; only the remaining candidates compete for the K_h supplementary positions. Ties are resolved by smaller semantic distance and then by stable storyline identifier. Selected events are finally restored to chronological order before prompt construction.

B Case Studies of Storyline Memory

B.1 Overview

We examine two LoCoMo questions that SUE-Mem answers correctly but StructMem and MRAgent answer incorrectly. Both come from `conv-50`: Q59 is a Multi-Hop question and Q49 is a Temporal question, with question numbers indexed from one in the original QA array. They illustrate cross-session evidence integration and the use of time information to distinguish similar activities.

All three systems use GPT-4o-mini for answer generation. SUE-Mem uses the full S, U, and E layers with $K_c = 20$, $K_s = 3$, $K_h = 9$, and $K_e = 5$. We compare the recorded answers under each system's memory, retrieval, and prompt configurations. Judge denotes the original binary per-question evaluation result. These selected cases illustrate differences in evidence organization and use; they do not isolate individual components or establish an aggregate performance advantage.

B.2 Case 1: Integrating the Origins of an Interest in Automotive Engineering

Question and required evidence. The question (`conv-50`, Q59, Multi-Hop) asks:

> Which events in Dave's life inspired him to take up auto engineering?

The reference answer is:

> attending a car show with Dad, working on an old car in a neighbor's garage when he was young, spent a summer restoring an old car with Dad

Answering requires three complementary experiences: a first car show with his father, repairing a neighbor's old car, and a childhood summer spent restoring a car with his father. These experiences appear in separate sessions (Table 8). A single relevant fact about Dave's interest in cars cannot cover the requested experiences.

Table 8: Source evidence for the three experiences in Q59. Dates identify the sessions in which the experiences were recounted, not when they occurred.

Experience	Dialogue / session date	Original dialogue excerpt
Restoring a car with his father	D12:4 / 2023-08-03	"I have fond memories of working on cars with my dad as a kid. We spent one summer restoring an old car."
Repairing a neighbor's old car	D25:15 / 2023-10-23	"I found an old car in a neighbor's garage and asked if I could fix it. That's when my love for car engineering began!"
Attending his first car show	D26:6 / 2023-10-25	"My dad took me to my first car show when I was 10, and I was wow-ed! I was hooked and wanted to learn more."

From atomic events to a shared retrieval unit. SUE-Mem represents the experiences as separate atomic events, preserving their specific content (Table 9). The neighbor's-car event joins six storylines, each of which updates its own summary. This experience can thus contribute both to the account of Dave's childhood interest and to the background of his later goal of building a custom car.

Table 9: Atomic-event content corresponding to Q59. Short identifiers are introduced only for readability.

Identifier	Atomic-event content
e_f	Dave has fond memories of restoring an old car with his dad during one summer when he was a kid.
e_n	Dave's love for car engineering began when he found an old car in a neighbor's garage and asked to fix it.
e_s	Dave attended his first car show at the age of 10, which sparked his interest in automotive engineering.

One storyline containing the childhood experiences reaches ten child events and splits into groups of six and four. The first retains all three required experiences; the second covers his longstanding mechanical interest, tinkering with engines in his father's garage, and helping neighbors repair cars. The first later incorporates current career information, reaching seven events. This organization keeps complementary experiences from different sessions together in a unit that can be retrieved for questions about how his interest developed.

Complementary summaries and child events. For Q59, this storyline ranks first in the initial semantic retrieval. Its `overAll` summary supplied to the answer model includes:

> "… at the tender age of 10 when he attended his first car show …"
>
> "… during the summer of his childhood … restoring an old car with his dad."
>
> "… discovered an old car in a neighbor's garage and eagerly sought permission to fix it …"

The summary therefore covers all three reference experiences. The selected child events provide finer-grained evidence alongside this account. Notably, the event about restoring a car with his father is absent from this storyline's selected Top-5 child events, but remains available in its summary. The

answer context can thus retain a required experience even when its individual event falls outside the local selection budget.

Comparison of recorded answers. Table 10 shows that SUE-Mem mentions all three experiences. StructMem identifies the first car show but omits both car-repair experiences. MRAgent focuses on working with Dave's father and the resulting sense of fulfillment. Its supporting material also concerns a recent car show and automotive workshops, which cannot substitute for the childhood first show and the neighbor's car. After three rounds of agent interaction, its answer still does not integrate all three required experiences.

Table 10: Recorded answers to Q59 and their original Judge results.

System	Generated answer	Coverage	Judge
SUE-Mem	Attended first car show at age 10, discovered old car in neighbor's garage, restored old car with dad.	All three experiences	1
StructMem	First car show at age 10.	First car show only	0
MRAgent	working on cars with my dad, refurbishing them gives me a sense of fulfillment	Work with his father only	0

Both baselines return relevant information but omit other required experiences. In this case, the storyline summary and atomic events jointly give SUE-Mem a more complete context. The comparison illustrates the difference between finding a relevant fragment and using complementary evidence together.

B.3 Case 2: Distinguishing Similar Activities Using Time Information

The question (`conv-50`, Q49, Temporal) asks:

> What was Dave doing in the first weekend of October 2023?

The reference answer is "attending a car show." Two activities in the history are easy to confuse. In D19:1, a session dated September 15, 2023, Dave mentions attending a rock concert the previous weekend. In D22:1, dated October 8, 2023, he says, "Last Friday I went to the car show," corresponding to October 6. We follow the dataset's reference answer, which associates the question's weekend description with that car show.

Table 11: Recorded answers to Q49 and their original Judge results.

System	Generated answer	Judge
SUE-Mem	Attended a car show.	1
StructMem	Attending a rock concert.	0
MRAgent	rock concert	0

SUE-Mem's context preserves the association between each activity and its time. The concert storyline places the event on the weekend before September 15, whereas the car-show storyline includes two child events dated `2023-10-06`. Although the concert ranks first in the semantic results and the car show ranks third, the answer model selects the car show using the available time information (Table 11).

Both baselines instead answer with the rock concert from another month. MRAgent ultimately cites D19:1, the September concert account. Although it queries the activity's session time during its interaction, its final answer does not satisfy the requested month. The case illustrates how providing event content together with its time can help distinguish similar activities beyond their semantic relevance.

B.4 Discussion

The two cases highlight complementary uses of storyline memory. In Q59, storylines bring together experiences from separate sessions, and the summary retains required information missing from the selected child events. In Q49, storylines preserve activity–time associations that help distinguish a

car show from a concert in another month. These examples show how related facts, specific events, and temporal information jointly support answer generation in the observed cases.

C Reproducibility and Experimental Protocol

C.1 LoCoMo Processing and Evaluation

Each LoCoMo conversation is parsed session by session in its original order. The session timestamp is used as the base time, with a one-second offset per turn to obtain a deterministic within-session ordering. Memory construction is completed before any question is answered, and the final incomplete dialogue window is explicitly flushed. Generated answers are not written back into memory, so questions from the same conversation cannot influence one another.

All LLM-based components, including the answer generator and the Judge evaluator, use GPT-4o-mini with temperature 0. All semantic representations use `text-embedding-3-small`. F1 computes token overlap. BLEU-1 uses unigram precision with add-one smoothing and the standard brevity penalty. J is the percentage of answers labeled `CORRECT` by an LLM-as-a-Judge evaluator (Zheng et al., 2023). Category and Overall scores are macro-averaged over questions and multiplied by 100. All reported F1, BLEU-1, and J values are averaged over five runs. The LoCoMo evaluation covers the Single-Hop, Multi-Hop, Temporal, and Open-Domain question categories; the adversarial category of LoCoMo is not included in this work.

C.2 LongMemEval Processing

LongMemEval experiments use the `longmemeval_s` benchmark, which contains 500 question-answering samples. Each sample spans roughly 40–48 sessions and about 115k tokens and fits within a 128k context window. As on LoCoMo, F1, BLEU-1, and J are averaged over five runs. The abstention task of LongMemEval is not included in this work.

C.3 Hyperparameters

Table 12 maps the notation used in the paper to the implementation configuration. The main experiments use $K_s + K_h = 12$ retrieved storylines; the efficiency experiment in Section 4.4 instead uses $K_s = 1$ and $K_h = 3$, as stated there.

Table 12: Default configuration used for the main LoCoMo experiment.

Symbol	Configuration key	Value	Meaning
K_t	`K_TURNS`	10	processed turns; pending window is $2K_t$
K_a	`EVENT_AGGREGATION_TOP_K`	10	assignment candidates
K_v	`EVENT_EVIDENCE_TOP_K`	5	evidence events per candidate
R	`EVENT_COLLECTION_SPLIT_INTERVAL`	10	split-check interval
K_c	`ANSWER_CANDIDATE_TOP_K`	20	first-stage storyline candidates
K_s	`ANSWER_V2_SEMANTIC_TOP_K`	3	protected semantic storylines
K_h	`ANSWER_V2_HYBRID_TOP_K`	9	hybrid supplementary storylines
K_e	`ANSWER_V2_CHILD_EVENT_TOP_K`	5	events retained per storyline
w_s	`ANSWER_SEMANTIC_WEIGHT`	0.55	semantic score weight
w_l	`ANSWER_LEXICAL_WEIGHT`	0.30	lexical score weight
w_x	`ANSWER_EXACT_MATCH_WEIGHT`	0.15	exact-anchor score weight
–	`LLM_MAX_RETRIES`	3	provider-call attempts

C.4 Persistence, Isolation, and Auditability

SQLite stores turns, summaries, profiles, atomic events, storyline membership, ownership, event embeddings, and audit records. Chroma stores only storyline summary embeddings and applies a conversation-identifier filter to every query. Therefore, first-stage retrieval never searches raw atomic events and cannot cross conversation boundaries. Each LLM input/output, update action, split decision, validation failure, and retrieval configuration is recorded for post-hoc inspection. The implementation targets Python 3.11 and fixes the dependency ranges in its project configuration.

D Additional Analysis

D.1 Category-Level Ablations

The main paper reports Overall ablations. Table 13 reports the corresponding category-level Judge results, which help distinguish failures in cross-event, temporal, and direct factual retrieval.

Table 13: Category-level ablation measured by Judge score. MH, T, OD, and SH denote Multi-Hop, Temporal, Open-Domain, and Single-Hop, respectively.

Variant	MH J	T J	OD J	SH J	Overall J
SUE-Mem	82.62	75.39	57.29	86.32	81.56
w/o multi-storyline assignment	69.29	72.70	49.41	78.85	73.98
w/o internal event evidence	79.08	72.24	53.45	83.19	78.30
w/o adaptive splitting	70.92	75.70	50.00	83.11	77.27

D.2 Detailed Category-Level Results for the Retrieval Budget

To complement the Overall results in Section 4.7, we report the effect of the final storyline budget $K_r = K_s + K_h$ separately for Multi-Hop, Temporal, Open-Domain, and Single-Hop questions. Each panel uses the same budgets $K_r \in \{4, 8, 12, 16, 20, 30, 40, 50\}$ and presents grouped bars for F1, BLEU-1, and J. This disaggregation reveals whether the aggregate trend is shared across question types or is dominated by a particular reasoning regime.

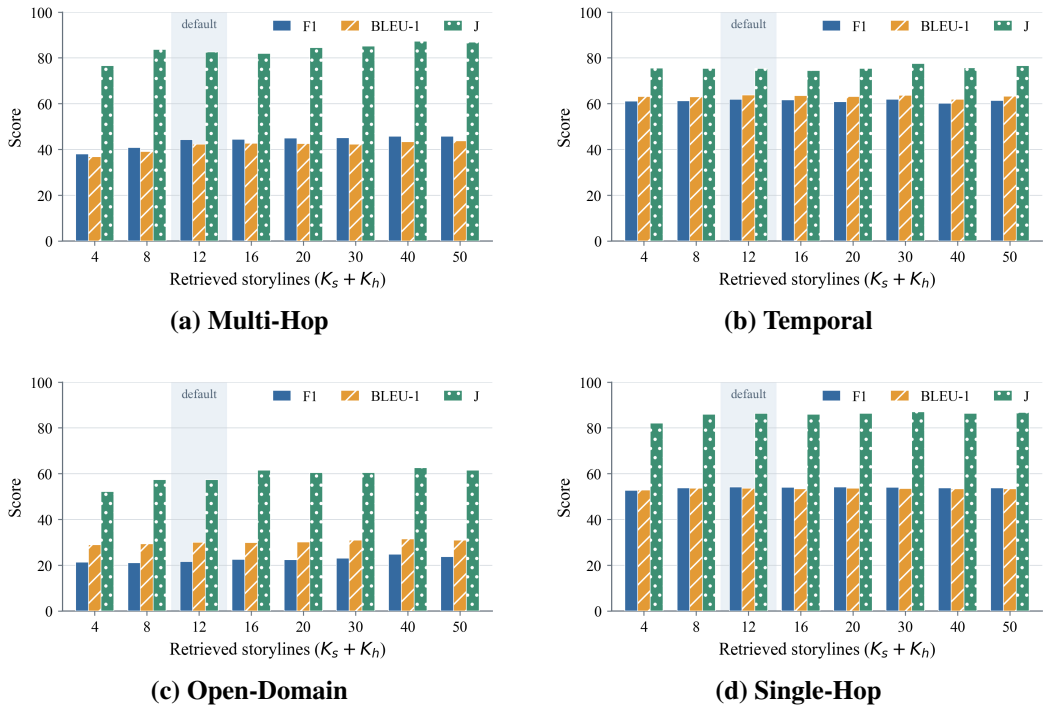


Figure 5: Category-level results for the retrieval-budget experiment in Section 4.7. Each panel reports F1, BLEU-1, and J as the total number of retrieved storylines $K_r = K_s + K_h$ varies; the shaded group marks the default setting $K_r = 12$.

Figure 5 shows that the benefit of a larger retrieval budget depends on the reasoning type. Multi-Hop performance improves substantially as additional storylines become available: relative to $K_r = 4$, $K_r = 40$ raises F1 from 38.00 to 45.79 and J from 76.59 to 87.23, while BLEU-1 reaches its maximum of 43.78 at $K_r = 50$. Open-Domain questions exhibit a similar but less monotonic pattern,

with all three metrics peaking at $K_r = 40$ (24.82 F1, 31.40 BLEU-1, and 62.50 J) before declining at 50, suggesting that broader evidence coverage eventually competes with retrieval noise. In contrast, Temporal results remain within a narrow range across all budgets, and Single-Hop performance largely saturates by $K_r = 8$–12. These results indicate that larger budgets primarily help evidence-intensive Multi-Hop and Open-Domain questions, whereas $K_r = 12$ captures most of the benefit for Temporal and Single-Hop questions at substantially lower inference cost.

E Additional Retrieval Sensitivity Experiments

The main paper studies semantic protection and the total number of retrieved storylines. We further conduct two controlled experiments on the internal hybrid-score weights and the number of child events retained from each selected storyline. Unless explicitly varied below, all remaining parameters use their default values and the final storyline budget remains $K_s + K_h = 12$.

E.1 Effect of Hybrid-Score Weights

The hybrid score combines semantic, lexical, and exact-anchor evidence using w_s, w_l, and w_x, respectively. Because the weighted sum is normalized by $w_s + w_l + w_x$, multiplying all three weights by the same constant leaves the ranking unchanged. We therefore change one weight at a time while fixing the other two at their default values. This design isolates whether the final answers benefit more from semantic coverage, fuzzy lexical matching, or exact entity/date/number anchors.

Table 14: Sensitivity analysis of the hybrid-score weights. Each non-default row varies one weight only. All reported metrics are Overall scores on LoCoMo.

Setting	w_s	w_l	w_x	F1	B-1	J
Default	0.55	0.30	0.15	**51.90**	**52.21**	81.56
Lower w_s	0.35	0.30	0.15	51.37	51.77	81.17
Higher w_s	0.75	0.30	0.15	51.63	51.90	**81.69**
Lower w_l	0.55	0.10	0.15	51.40	51.80	81.56
Higher w_l	0.55	0.50	0.15	51.44	51.79	80.97
Lower w_x	0.55	0.30	0.05	51.52	51.90	81.17
Higher w_x	0.55	0.30	0.30	51.77	52.09	81.10

The default weights achieve the best F1 and BLEU-1 scores, while increasing w_s to 0.75 yields the highest J score (81.69). This 0.13-point gain in J, however, is accompanied by decreases of 0.27 F1 and 0.31 BLEU-1, indicating that excessive reliance on summary-level semantics can weaken answer-level lexical fidelity. Reducing w_s degrades all three metrics, supporting the role of semantic similarity as the primary retrieval signal. Increasing w_l causes the largest J reduction (0.59 points), suggesting that strong lexical matching can promote surface-level distractors. Likewise, increasing w_x yields F1 and BLEU-1 scores close to the defaults but reduces J by 0.46 points. Overall, $(w_s, w_l, w_x) = (0.55, 0.30, 0.15)$ provides the most balanced trade-off between semantic coverage and fact-specific matching.

E.2 Effect of Child Events per Storyline

After storyline selection, SUE-Mem retains the top K_e child events from each storyline and restores them to chronological order. A small K_e reduces the answer context but may omit intermediate evidence required by multi-hop or temporal questions. Conversely, a large K_e increases evidence coverage but can introduce redundant facts and distract the answer generator. We vary $K_e \in \{1, 3, 5, 7, 10\}$ while fixing $K_c = 20$, $K_s = 3$, $K_h = 9$, and all score weights.

Table 15: Effect of the number of retained child events per selected storyline. All reported metrics are Overall scores on LoCoMo.

K_e	Overall F1	Overall B-1	Overall J
1	50.98	51.35	81.69
3	51.64	51.95	**81.82**
5	**51.90**	**52.21**	81.56
7	51.37	51.77	81.43
10	51.53	51.78	81.49

Increasing K_e from 1 to 5 steadily improves F1 and BLEU-1, showing that a single event often cannot provide enough context for cross-event reasoning. The J score peaks earlier at $K_e = 3$ (81.82), whereas $K_e = 5$ obtains the best F1 (51.90) and BLEU-1 (52.21). Retaining more than five events provides no further benefit: at $K_e = 7$, all three metrics decline, and $K_e = 10$ recovers only marginally. These results indicate that evidence coverage saturates after a small number of high-ranked child events, after which redundant or weakly related details can distract answer generation. We therefore use $K_e = 5$ as the default because it offers the strongest lexical accuracy while maintaining a competitive Judge score and a compact question-time context.

F Prompt Specifications

The anonymous supplementary source code contains the complete literal prompts used in the experiments. For readability and auditability, Table 16 summarizes the interfaces of all seven LLM-based components, and the following subsections present three representative templates for atomic-event extraction, storyline splitting, and answer generation. These templates preserve their roles, input variables, decision rules, and output schemas. Braced fields are replaced at runtime by a deterministic renderer. Every call is sent as a single system message; JSON-producing components are parsed as objects and rejected when the required schema is not satisfied.

F.1 Prompt Interface Summary

Table 16: Interfaces of the LLM-based components.

Component	Runtime inputs	Output	Main constraint
Session summary	old summary, new turns	JSON text field	retain chronology and unresolved plans
Speaker profile	speaker, old profile, summary, turns	JSON object	stable traits of one speaker only
Event extraction	turns	event JSON array	standalone facts with occurrence times
Logical assignment	new event, summary, evidence events	decision JSON	concrete anchor and direct relation
Storyline integration	complete event list	plain text	chronological, complete, single theme
Storyline splitting	summary, complete event list	grouping JSON	coherent groups using valid event IDs
Answer generation	S, U^A, U^B, retrieved evidence, question	plain text	evidence-only concise answer

F.2 Atomic-Event Extraction Template

Atomic-event extraction

```
You are a Personal Information Extractor.
Your task is to extract all possible facts or information about the
  speakers from a
conversation,
INPUT JSON
{
  "turns": "[YYYY-MM-DD HH:mm][speaker]:text\n[YYYY-MM-DD
    HH:mm][speaker]:text"
}
Important Instructions:
0. For each message, stop and **carefully** evaluate its content
  before moving to the next.
```

Atomic-event extraction *continued*

```
Do NOT reorder, batch-skip, or skip ahead --- treat messages
  one-by-one.
1. You MUST process every user message in order, one by one.
For each message, decide whether it contains any factual information.
- If yes -> extract it and rephrase into a standalone sentence.
- Do NOT skip just because the information looks minor, trivial, or
  unimportant.
Extract ALL meaningful information including:
* Past events and current states
* Future plans and intentions
* Thoughts, opinions, and attitudes
* Wants, hopes, desires, and preferences
2. **CRITICAL - Preserve All Specific Details**:
When extracting facts, you MUST include ALL specific entities and
  details mentioned:
- **Full names with context**: "The Name of the Wind" by Patrick
  Rothfuss (not just "a
book")
- **Complete location names**: Galway, Ireland; The Cliffs of Moher;
  Barcelona (not just
"a city")
- **Specific event names**: benefit basketball game, study abroad
  program (not just "an
event")
- **Product/item details**: vintage camera, brand new fire truck (not
  just "a camera")
- **Numbers and quantities**: 4 years ago, next month, last week
- **Company/organization names**: beverage company, fire-fighting
  brigade
Additionally, **infer implied information** when clearly supported:
- If multiple related items mentioned -> may infer general pattern
- Keep BOTH specific facts AND inferred insights as separate entries
3. Perform light contextual completion so that each fact is a clear
  standalone statement.
4. **Time Handling**:
Note: Distinguish mention time (when said) vs event time (when
  happened).
- For events with relative time (yesterday, last week, X ago, next
  month):
The occurred_at field records the actual time when an event occurs,
  which supports year-level, month-level and day-level granularity,
  and can represent time points in the past, present or future.
5. OUTPUT JSON
{
  "events": [
    {"content": string, "occurred_at": string (ISO-8601 or fuzzy time
      expression)}
  ]
}
Reminder: Be exhaustive and ALWAYS include specific names, titles,
  locations, and details
in every fact.
example:
input:
{
```

Atomic-event extraction *continued*

```
  "turns": "[2024-01-07 17:24][Tim]:John! I'm going to Galway, Ireland
    next month for a one-semester exchange program.\n[2024-01-07
    17:24][John]:That's amazing! Where are you planning to
    stay?\n[2024-01-07 17:24][Tim]:I'll be staying right in Galway. I
    also want to visit the Cliffs of Moher.\n[2024-01-07
    17:24][John]:Sounds great! By the way, I held a charity basketball
    game last week.\n[2024-01-07 17:24][Tim]:That's so cool! I'm
    currently reading The Name of the Wind written by Patrick
    Rothfuss.\n[2024-01-07 17:24][John]:That sounds
    interesting!\n[2024-01-12 13:41][John]:Good news! I secured an
    endorsement partnership with a well-known beverage company last
    week.\n[2024-01-12 13:41][Tim]:Congratulations! That's
    incredible.\n[2024-01-12 13:41][John]:Thanks! Also, Barcelona is
    absolutely worth visiting.\n[2024-01-12 13:41][Tim]:I'll add it to
    my travel list!"
}
output:
{
  "events": [
    {
      "content": "Tim will travel to Galway, Ireland for a
        one-semester exchange program in February 2024.",
      "occurred_at": "2024-02"
    },
    {
      "content": "Tim plans to complete his exchange studies in
        Galway, Ireland in February 2024.",
      "occurred_at": "2024-02"
    },
    {
      "content": "Tim will live in Galway during his exchange
        period.",
      "occurred_at": "2024-02"
    },
    {
      "content": "Tim intends to travel to the Cliffs of Moher for
        sightseeing.",
      "occurred_at": "2024-02"
    },
    {
      "content": "John hosted a charity basketball game in the week
        prior to January 7, 2024.",
      "occurred_at": "the week before 2024-01-07"
    },
    {
      "content": "Tim is currently reading the book The Name of the
        Wind written by Patrick Rothfuss.",
      "occurred_at": "2024-01-07"
    },
    {
      "content": "Tim is reading the novel titled The Name of the
        Wind.",
      "occurred_at": "2024-01-07"
    },
    {
```

Atomic-event extraction *continued*

```
      "content": "John obtained an endorsement cooperation with a
        famous beverage company in the week prior to January 12,
        2024.",
      "occurred_at": "the week before 2024-01-12"
    },
    {
      "content": "John thinks Barcelona is a city worth visiting and
        recommended it to Tim.",
      "occurred_at": "2024-01-12"
    },
    {
      "content": "Tim has a travel list and intends to add Barcelona
        to it.",
      "occurred_at": "2024-01-12"
    }
  ]
}
now input is:
{turns}
```

F.3 Storyline Splitting Template

Storyline splitting

```
Role
You are an event-collection coherence auditor for a long-term memory
  system.
Your task is to determine whether an event collection should remain
  intact or be split into smaller, retrieval-ready event collections.

Input Structure
{
  "collection": {
    "id": "collection identifier",
    "overAll": "existing collection summary",
    "events": [
      {
        "eventId": "event identifier",
        "content": "event content",
        "occurred_at": "ISO-8601 timestamp or fuzzy time expression"
      }
    ]
  }
}
The child events are the primary evidence.
The `overAll` summary is only a reference and may be overly broad or
  inaccurate.

Decision Standard
Return `should_split: false` only when every event belongs to the same
  bounded event thread.
A bounded event thread must satisfy both conditions:
1. All events concern the same identifiable incident, object, goal,
  problem, plan, project, relationship episode, or stateful process.
2. Every event directly describes, updates, explains, causes,
  continues, corrects, or completes that same thread.
```

Storyline splitting *continued*

```
Different stages of the same specific process should remain together
  when they form one continuous timeline or state transition.
Return `should_split: true` when the events contain two or more
  independently understandable threads that do not directly depend on
  each other.
The following are not sufficient reasons to keep events together:
- They involve the same person.
- They share a broad topic or similar keywords.
- They occurred close together in time.
- They express generic support, encouragement, feelings, or opinions.
- They can be combined only under a broad label such as a life
  journey, career, hobby, health, family, or personal development.
Use this retrieval test:
If one subset can answer a coherent group of questions without
  requiring the facts in another subset, they should normally be
  separate collections.

Size Rule
Event count alone does not require splitting, and there is no hard
  maximum collection size.
However, as a collection grows, verify that every event still directly
  contributes to the same bounded thread. Do not keep a large
  collection intact merely because all events fit under one broad
  theme.

Grouping Rules
When splitting is required:
1. Use only the original input `eventId` values.
2. Every input `eventId` must appear in at least one group.
3. Do not invent, modify, or omit any `eventId`.
4. Create the smallest number of groups needed to restore coherence.
5. Each group must be independently understandable and suitable for
  generating a focused summary.
6. An independent event may form a single-event group.
7. Do not split mechanically by event count, timestamp, or input
  order.
8. Near-duplicate descriptions of the same fact should normally remain
  together.
9. An event may appear in multiple groups only when it has a direct
  factual role in every corresponding thread.
10. Every group must contain at least one event not contained in all
  other groups.
11. Do not produce duplicate groups, subset-only groups, or a group
  containing the entire input collection alongside smaller groups.

Before returning the result, verify internally that:
- every eventId is valid;
- every input event is covered;
- every group has a distinct bounded thread;
- no group is connected only by a broad topic;
- the split produces a meaningful partition rather than duplicating
  the original collection.

Mandatory Output Format
Return pure JSON only. Do not output Markdown, explanations, reasons,
  summaries, or additional fields.
```

Storyline splitting *continued*

```
No Split
{
  "should_split": false,
  "groups": []
}

Split
{
  "should_split": true,
  "groups": [
    {
      "eventsId": ["event-id-1", "event-id-2"]
    },
    {
      "eventsId": ["event-id-3", "event-id-4"]
    }
  ]
}

Current Input
Event Collection: {eventCollection}
```

F.4 Answer Generation Template

Answer generation

```
You are an Intelligent Memory Assistant. Your job is to retrieve
  accurate information from conversation memories to respond to
  queries.
Background
You can access a speaker's memory records from the dialogue. Each
  memory has a timestamp and may contain information related to the
  question.
Step-by-Step Procedure
1. Filter all memories relevant to the query.
2. Check timestamps and content of matched memories carefully.
3. If a direct matching answer exists, keep all nouns, names, dates,
  numbers, locations, items, actions and descriptive terms exactly as
  written in the original memory.
4. Never replace specific answers with:
    - Broader categories
    - Synonyms
    - Interpretations
    - Topic summaries
5. Generate precise, concise answers using only evidence from
  memories.
6. Retain all independent items requested by the query.
7. Final answers must be concise. Do not remove critical entity
  information just for length constraints.
    + Yes/No questions: Only output Yes, No, Likely Yes, Likely No.
    + Person queries: Output only the full name.
    + Country queries: Output only the country name; no subordinate
      cities.
    + City queries: Output only the city name.
    + Year queries: Output only the four-digit year.
    + Date queries: Output only the date or original time string.
```

Answer generation *continued*

```
    + Number queries: Output only the value.
    + Object queries: Output only the target object.
    + List queries: Output all entries separated by commas.
    + Do not repeat the subject if it is already present in the
      question.
    + Do not add authors, explanations or background unless explicitly
      requested.

Session summaries:
{session_summaries}
user profile:
{speaker_1_name}:{user1_profile}
{speaker_2_name}:{user2_profile}
Memories for users:
{speaker_memories}
Question: {question}
Answer:
```