Building Long-Term LLM Memory with Neo-Davidsonian Discourse Units

Sizhe Zhou[1†] **Yanzhen Shen**[2] **Siru Ouyang**[1] **Yuyang Bai**[3]
Yiheng Shu[4] **Yu Zhang**[3†] **Jiawei Han**[1]

[1]University of Illinois Urbana-Champaign [2]Stanford University
[3]Texas A&M University [4]The Ohio State University

{sizhez,siruo2,hanj}@illinois.edu yanzhen4@stanford.edu
{ybai,yuzhang}@tamu.edu shu.251@osu.edu

Abstract

Long-horizon conversational agents need persistent memory beyond a fixed context window, yet memory effectiveness depends on the units stored and retrieved. Existing systems either discard fine-grained details during compression, retrieve units that are too coarse or context-poor, or fragment discourse into entity-relation triples. We propose *Neo-Davidsonian Discourse Units*: enriched Elementary Discourse Units (EDUs) that each express one event or fact with normalized entities, temporal grounding, and enough context to stand alone. The resulting units are non-lossy at the session level, atomic for retrieval, and naturally expose event arguments for graph-based associative recall. We instantiate this representation in EMem, which retrieves EDUs with dense representations and relevance filtering based on large language models, and EMem-G, which further organizes sessions, EDUs, and arguments in an event graph and applies Personalized PageRank. On LoCoMo and LongMemEval$_S$ benchmarks, our methods improve over prior state-of-the-art by up to 21.3% while using up to 7× shorter effective context.

1 Introduction

Large language models (LLMs) have impressive conversational abilities, but fixed context windows still limit coherence and personalization over extended interactions. Even long-context models degrade when relevant information lies in the middle of the input (Liu et al., 2024), and recalling facts from distant sessions remains difficult (Maharana et al., 2024; Wu et al., 2025a). External memory is a natural remedy, but its effectiveness hinges on *memory unit representation*: what is stored, at what granularity, and how units are organized for retrieval and reasoning.

Effective memory units should be both *atomic*, encoding one coherent topic for precise matching, and *self-complete*, containing enough context to be understood in isolation. Existing designs rarely satisfy both. **Coarse-grained units**, such as whole sessions (Chen et al., 2024) or session summaries (Packer et al., 2024; Chhikara et al., 2025; Zhong et al., 2024; Lee et al., 2024), mix many topics, weaken embedding matches, and force downstream models to read unnecessary context. Summarization may also drop details later queried. **Fine-grained units**, such as individual turns (Wu et al., 2025a), improve precision but often leave pronouns, dates, and cross-turn dependencies unresolved. Graph-based memories introduce structure through entity-relation triples (Gutierrez et al., 2024; Gutiérrez et al., 2025; Rasmussen et al., 2025), yet triples fragment coherent events into scattered binary facts, making it hard to recover

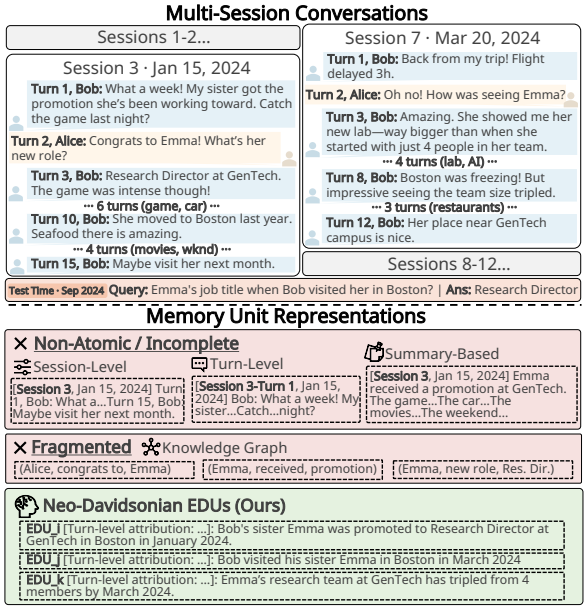

Figure 1: An illustrative multi-session conversation (top) and contrasting memory unit representations (bottom). Existing designs are *non-atomic*, *incomplete* or *fragmented*. Our neo-Davidsonian EDUs are atomic, self-complete, and non-lossy.

[†]Corresponding authors.

what the dialogue originally expressed.

We address these limitations by combining two complementary theories. **Rhetorical structure theory (RST)** treats discourse as decomposable into *Elementary Discourse Units (EDUs)*, minimal clause-level units that convey single propositions (Mann and Thompson, 1988; Carlson et al., 2001; Asher and Lascarides, 2003). Raw EDUs, however, may not stand alone. RST tree structures can encode inter-EDU relations, but are costly to construct and search. We instead enrich each EDU with normalized entities, resolved coreference, temporal grounding, and minimal local context, so the EDUs from a session collectively preserve its expressed information. **Neo-Davidsonian event semantics** further motivates representing meaning as an *event* that binds participants, times, locations, and circumstances into one object rather than scattered binary relations (Davidson, 2001; Parsons, 1990). We call the resulting units **neo-Davidsonian EDUs**.[1] Each EDU expresses one event or fact,[2] is self-complete, and exposes structured arguments for graph-based associative recall without losing event-level coherence.

Concretely, we instruct an LLM to decompose each conversation session into short, self-contained event statements with normalized entity mentions, temporal grounding, and source-turn metadata. Our lightweight system, **EMem**, performs dense retrieval over EDUs followed by LLM-based relevance filtering. **EMem-G** additionally organizes sessions, EDUs, and event arguments into a heterogeneous event-centric memory graph. At query time, it retrieves EDU and argument nodes, filters them for recall, and applies Personalized PageRank to select graph-consistent EDUs. Experiments on LoCoMo (Maharana et al., 2024) and LongMemEval$_S$ (Wu et al., 2025a) show that EMem-G improves over prior state-of-the-art accuracy by up to 4.8% on LoCoMo and 21.3% on LongMemEval$_S$, with up to 7× shorter effective context on LoCoMo.

In summary, our contributions are:

- We introduce neo-Davidsonian EDUs, an event-centric memory representation grounded in rhetorical structure theory and neo-Davidsonian event semantics that yields non-lossy, atomic, and self-complete memory units.
- We propose EMem[3], a strong lightweight retriever with LLM-based EDU filtering, and EMem-G, which adds a heterogeneous event graph with Personalized PageRank for associative recall.
- We show strong results on LoCoMo and LongMemEval$_S$, improving memory-augmented QA while substantially reducing context size.

2 Related Work

2.1 Memory for LLM Agents

Cognitively inspired memory systems such as Nemori (Ma et al., 2026), LightMem (Fang et al., 2026), LiCoMemory (Huang et al., 2026), and A-Mem (Xu et al., 2025) organize interactions into episodes, consolidate memories across stages, or dynamically restructure stored content. Many rely on clustering, summarization, or note-taking to keep the memory store compact, improving efficiency at the cost of potentially discarding fine-grained details. PREMem (Kim et al., 2025) shifts more reasoning to the write phase by storing enriched fragments with inferred cross-session relations. System-level frameworks such as MemOS (Li et al., 2025) and Mem0 (Chhikara et al., 2025) treat memory as an operating-system primitive or service layer, combining scalable storage with summarization and fact extraction. Filesystem-based memory, in which an agent organizes a growing directory of markdown files through generic file tools, has recently been studied as a memory medium in its own right (Zhou et al., 2026). In contrast, our approach avoids lossy compression: it preserves expressed information in event-centric memory units and uses inference-time retrieval and filtering to manage relevance.

2.2 Structured and Graph-Based Memory

Graph-based approaches support associative recall by connecting related memories explicitly. HippoRAG (Gutierrez et al., 2024) and HippoRAG 2 (Gutiérrez et al., 2025) build graphs over entities and passages and use Personalized PageRank to retrieve multi-hop evidence. Zep (Rasmussen et al., 2025) and Mem0's graph extension (Chhikara et al., 2025) maintain temporal knowledge graphs for agents, while ComoRAG (Wang et al., 2026) organizes long narratives into cognitively inspired structures for iterative RAG. SGMem (Wu et al., 2025b)

[1] Throughout the paper, we use "EDUs" to refer to our enriched neo-Davidsonian EDUs unless otherwise specified.

[2] This broad "event" concept includes facts.

[3] Our code is publicly available at https://github.com/KevinSRR/EMem.

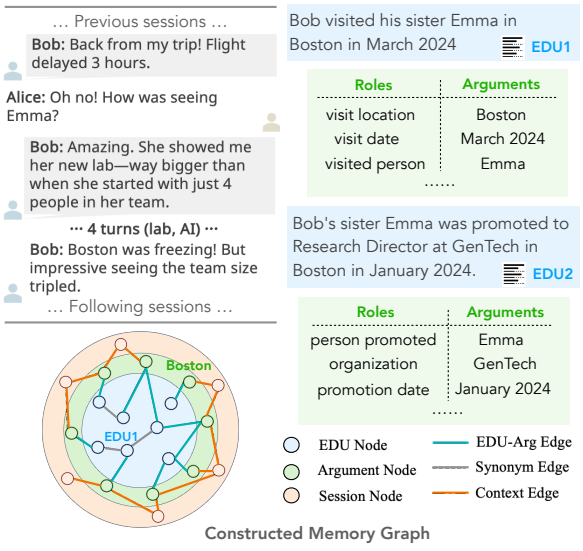

Figure 2: Memory construction. Conversations are decomposed into enriched EDUs, arguments, and an event graph.

represents dialogue as sentence-level graphs connecting sentences within and across sessions. Our work is closest to HippoRAG 2 and SGMem, but differs in two respects. First, enriched EDUs rather than triples or raw sentences serve as the primary memory units, preserving event coherence while still exposing arguments for graph construction. Second, we combine graph retrieval with recall-oriented LLM filtering over both EDU and argument nodes. This is especially useful in conversational memory, where relevant evidence is often referenced implicitly or without explicit names.

3 Method

Task Definition. A conversation consists of timestamped sessions $\mathcal{S} = (s_1, \ldots, s_T)$. Each session $s \in \mathcal{S}$ has timestamp t_s and turns

$$s = \Big((\text{SPK}_1, u_1), (\text{SPK}_2, u_2), \ldots, (\text{SPK}_L, u_L)\Big),$$

where u_ℓ is uttered by speaker SPK_ℓ. Given a question q, the task is to answer it using the full conversation history. We focus on the memory mechanism and assume a text encoder $h(\cdot)$ for retrieval and a QA model f_{QA} that generates an answer from (q, memory). We describe memory construction in Section 3.1 (illustrated in Figure 2) and retrieval in Section 3.2 (illustrated in Figure 3).

3.1 Memory Construction

Building on the design goals in Section 1, we construct long-term memory as enriched event-centric discourse units.

Elementary Discourse Unit. Following Mann and Thompson (1988), we decompose discourse into Elementary Discourse Units (EDUs), each conveying a single proposition and therefore offering a natural level of atomicity. For example, several turns about a trip may yield an EDU such as "`Bob spent five days in Tokyo in March 2024 to attend the Global AI Innovation Symposium 2024 at the University of Tokyo.`" This unit is mildly abstractive but self-complete: it consolidates scattered turns, resolves references, and normalizes entities and time expressions.

Neo-Davidsonian EDU. Triple-based memory graphs (Gutierrez et al., 2024; Gutiérrez et al., 2025; Rasmussen et al., 2025) would split the example into relations such as `(Bob, attend, Global AI Innovation Symposium 2024)`, `(Bob, stay_in, Tokyo)`, and `(Symposium 2024, time, March 2024)`. Such triples support schema-driven association but fragment the source event, requiring later retrieval to recombine participants, times, and circumstances. Motivated by neo-Davidsonian event semantics (Davidson, 2001; Parsons, 1990), we instead treat the event as the primary memory object. Each EDU node is a self-complete, human-readable memory cell that preserves local conversational coherence while remaining fine-grained enough for precise retrieval and argument extraction.

Neo-Davidsonian EDU Extraction. For each session $s \in \mathcal{S}$, an LLM extractor g_{EDU} receives the timestamp, speaker names, turns, and one in-context exemplar, and outputs EDUs $\mathcal{E}_s = \{e_1^{(s)}, \ldots, e_N^{(s)}\}$. Each EDU is a short natural-language description with metadata:

$$e = \Big(\text{text}(e), \text{src}(e), \tau(e)\Big),$$

where $\text{src}(e)$ is the set of supporting turn indices and $\tau(e) = t_s$. For long structured assistant responses in LongMemEval$_S$, we also create structured chunk nodes whose summaries are used for indexing and whose full content is stored for QA. Details of this dataset-specific handling are provided in Appendix B. Aggregating across sessions yields the global EDU set $\mathcal{E} = \bigcup_{s \in \mathcal{S}} \mathcal{E}_s$.

Event Argument Extraction. For each EDU $e \in \mathcal{E}$, a second LLM g_{arg} treats e as one event and returns an event type $t(e)$ and role-argument pairs $\{(r_k, a_k)\}_{k=1}^{K}$. We collect unique argument

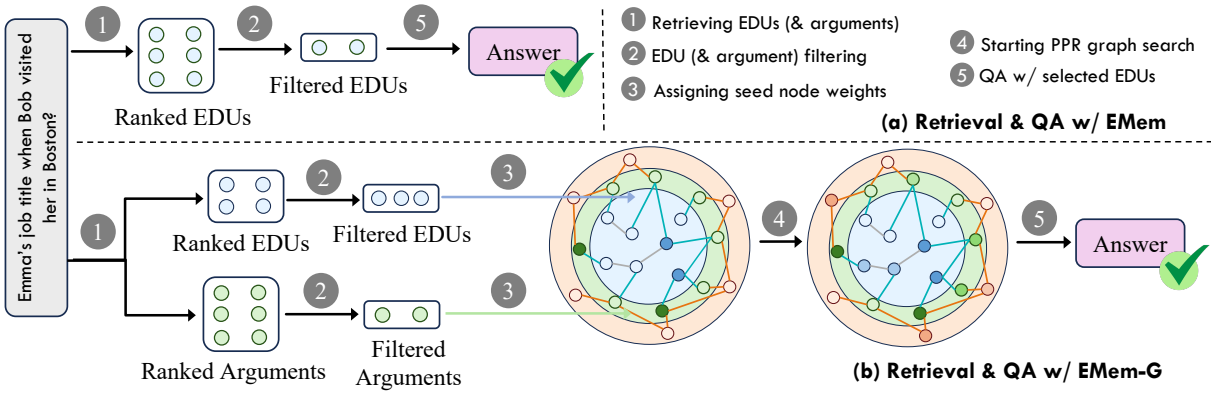

Figure 3: Memory retrieval. EMem retrieves EDUs using retrieval and filtering to provide a compact QA context, while EMem-G additionally retrieves arguments and propagates relevance over the graph.

strings into $\mathcal{A}$ and embed each argument as $h(a)$. Role labels are retained for interpretability but not otherwise used. Instead, argument strings themselves become schema-light graph nodes.

Graph Construction. We construct an event graph $G = (V, E)$ with three types of nodes:

$$V_{\text{session}} = \{v_s \mid s \in \mathcal{S}\},$$
$$V_{\text{EDU}} = \{v_e \mid e \in \mathcal{E}\},$$
$$V_{\text{argument}} = \{v_a \mid a \in \mathcal{A}\},$$

and three types of edges:

$$E_{\text{session-EDU}} = \{(v_s, v_e) \mid e \in \mathcal{E}_s\},$$
$$E_{\text{EDU-argument}} = \{(v_e, v_a) \mid a \text{ is an argument of } e\},$$
$$E_{\text{synonym}} = \{(v_a, v_{a'}) \mid \text{sim}(a, a') \geq \delta\}.$$

Here, $\text{sim}(a, a')$ is cosine similarity between $h(a)$ and $h(a')$, and we cap the number of synonym neighbors for each a (e.g., at 100). As new sessions arrive, we add their sessions, EDUs, arguments, and edges, and cache EDU embeddings $h(e) = h(\text{text}(e))$ for retrieval.

3.2 Memory Retrieval

For the memory retrieval step, we propose two variants. EMem retrieves and filters EDUs to form a compact memory context. EMem-G additionally retrieves argument nodes and propagates relevance through the event graph.

3.2.1 Lightweight Retrieval (EMem)

Given query q, we encode $z_q = h(q)$ and retrieve the top-K_e EDUs by cosine similarity to $h(e)$, forming a candidate set $C^{\text{EDU}}(q)$.

Recall-Oriented LLM Filtering. Embedding similarity is brittle for generic references (e.g., "`that trip`") or highly specific arguments, so we apply a recall-oriented LLM filter. The filter receives q and $\{\text{text}(e) \mid e \in C^{\text{EDU}}(q)\}$ and selects EDUs relevant to answering q. Its discrete selection induces a binary indicator $f_{\text{EDU}}(q, e) \in \{0, 1\}$, and the retained set is

$$R_{\text{lite}}(q) = \{e \in C^{\text{EDU}}(q) \mid f_{\text{EDU}}(q, e) = 1\}.$$

The QA context concatenates retained EDUs with timestamps and source-turn speaker names. For structured chunks, it uses the stored full chunk content.

The QA model then answers

$$\hat{y} = f_{\text{QA}}\Big(q, \{(\text{text}(e), \text{src}(e), \tau(e)) : e \in R_{\text{lite}}(q)\}\Big),$$

with a zero-shot chain-of-thought prompt (Wei et al., 2022).

3.2.2 Graph-Based Retrieval (EMem-G)

EMem-G first retrieves $C^{\text{EDU}}(q)$ as above. In parallel, an LLM mention detector extracts surface mentions $M(q) = \{m_1, \ldots, m_M\}$ covering entities, noun phrases, and salient concepts. For each mention, we retrieve top-K_a argument nodes by similarity between $h(m)$ and $h(a)$, forming a candidate argument set $C^{\text{arg}}(q)$.

We apply the same recall-oriented filtering to EDUs and arguments:

$$\tilde{C}^{\text{EDU}}(q) = \{e \in C^{\text{EDU}}(q) \mid f_{\text{EDU}}(q, e) = 1\},$$
$$\tilde{C}^{\text{arg}}(q) = \{a \in C^{\text{arg}}(q) \mid f_{\text{arg}}(q, a) = 1\}.$$

The filters deliberately retain borderline candidates. Graph propagation and final QA can later downweight spurious evidence.

		Temporal Reasoning			Open Domain			Multi-Hop			Single-Hop			Overall		
		LLM Score	F1	BLEU-1	LLM Score	F1	BLEU-1	LLM Score	F1	BLEU-1	LLM Score	F1	BLEU-1	LLM Score	F1	BLEU-1
gpt-4o-mini	FullContext	$0.562_{\pm 0.004}$	0.441	0.361	$0.486_{\pm 0.005}$	0.245	0.172	$0.668_{\pm 0.003}$	0.354	0.261	$0.830_{\pm 0.001}$	0.531	0.447	$0.723_{\pm 0.000}$	0.462	0.378
	LangMem	$0.249_{\pm 0.003}$	0.319	0.262	$0.476_{\pm 0.005}$	**0.294**	0.235	$0.524_{\pm 0.003}$	0.335	0.239	$0.614_{\pm 0.002}$	0.388	0.331	$0.513_{\pm 0.003}$	0.358	0.294
	Mem0	$0.504_{\pm 0.001}$	0.444	0.376	$0.406_{\pm 0.000}$	0.271	0.194	$0.603_{\pm 0.000}$	0.343	0.252	$0.681_{\pm 0.000}$	0.444	0.377	$0.613_{\pm 0.000}$	0.415	0.342
	RAG	$0.237_{\pm 0.000}$	0.195	0.157	$0.326_{\pm 0.005}$	0.190	0.135	$0.313_{\pm 0.003}$	0.186	0.117	$0.320_{\pm 0.001}$	0.222	0.186	$0.302_{\pm 0.000}$	0.208	0.164
	Zep	$0.589_{\pm 0.003}$	0.448	0.381	$0.396_{\pm 0.000}$	0.229	0.157	$0.505_{\pm 0.007}$	0.275	0.193	$0.632_{\pm 0.001}$	0.397	0.337	$0.585_{\pm 0.001}$	0.375	0.309
	Nemori	$0.710_{\pm 0.000}$	0.567	0.466	$0.448_{\pm 0.005}$	0.208	0.151	$0.653_{\pm 0.002}$	0.365	0.256	$0.821_{\pm 0.002}$	**0.544**	**0.432**	$0.744_{\pm 0.001}$	**0.495**	0.385
	EMem	$\mathbf{0.771}_{\pm 0.004}$	0.574	0.461	$\mathbf{0.602}_{\pm 0.009}$	0.285	**0.237**	$0.702_{\pm 0.004}$	**0.406**	**0.307**	$\mathbf{0.830}_{\pm 0.002}$	0.497	0.414	$\mathbf{0.780}_{\pm 0.001}$	0.483	0.393
	EMem-G	$0.760_{\pm 0.003}$	**0.581**	**0.468**	$0.573_{\pm 0.013}$	0.242	0.199	$\mathbf{0.747}_{\pm 0.006}$	**0.406**	0.305	$0.823_{\pm 0.001}$	0.504	0.422	$\mathbf{0.780}_{\pm 0.000}$	0.487	**0.397**
gpt-4.1-mini	FullContext	$0.742_{\pm 0.004}$	0.475	0.400	$0.566_{\pm 0.010}$	0.284	0.222	$0.772_{\pm 0.003}$	0.442	0.337	$0.869_{\pm 0.002}$	0.614	0.534	$0.806_{\pm 0.001}$	0.533	0.450
	LangMem	$0.508_{\pm 0.003}$	0.485	0.409	$0.590_{\pm 0.005}$	**0.328**	**0.264**	$0.710_{\pm 0.002}$	0.415	**0.325**	$0.845_{\pm 0.001}$	0.510	0.436	$0.734_{\pm 0.001}$	0.476	0.400
	Mem0	$0.569_{\pm 0.001}$	0.392	0.332	$0.479_{\pm 0.000}$	0.237	0.177	$0.682_{\pm 0.003}$	0.401	0.303	$0.714_{\pm 0.001}$	0.486	0.420	$0.663_{\pm 0.000}$	0.435	0.365
	RAG	$0.274_{\pm 0.000}$	0.223	0.191	$0.288_{\pm 0.005}$	0.179	0.139	$0.317_{\pm 0.003}$	0.201	0.128	$0.359_{\pm 0.002}$	0.258	0.220	$0.329_{\pm 0.002}$	0.235	0.192
	Zep	$0.602_{\pm 0.001}$	0.239	0.200	$0.438_{\pm 0.000}$	0.242	0.193	$0.537_{\pm 0.003}$	0.305	0.204	$0.669_{\pm 0.001}$	0.455	0.400	$0.616_{\pm 0.000}$	0.369	0.309
	Nemori	$0.776_{\pm 0.003}$	**0.577**	**0.502**	$0.510_{\pm 0.009}$	0.258	0.193	$0.751_{\pm 0.002}$	**0.417**	0.319	$0.849_{\pm 0.002}$	**0.588**	**0.515**	$0.794_{\pm 0.001}$	**0.534**	**0.456**
	EMem	$0.800_{\pm 0.003}$	0.491	0.389	$0.652_{\pm 0.010}$	0.270	0.237	$0.790_{\pm 0.002}$	0.350	0.273	$0.897_{\pm 0.001}$	0.509	0.430	$0.842_{\pm 0.001}$	0.461	0.381
	EMem-G	$\mathbf{0.808}_{\pm 0.001}$	0.513	0.406	$\mathbf{0.717}_{\pm 0.005}$	0.291	0.253	$\mathbf{0.796}_{\pm 0.002}$	0.376	0.308	$\mathbf{0.905}_{\pm 0.001}$	0.510	0.432	$\mathbf{0.853}_{\pm 0.000}$	0.473	0.393

Table 1: Results on LoCoMo (Maharana et al., 2024) by question type. LLM Score is evaluated by a `gpt-4o-mini` judge; its values report the mean over three judge runs, with standard deviation shown as a subscript. Bold and underlining denote the best and second-best results, respectively. Baseline results follow Ma et al. (2026).

Seed Initialization. We initialize a nonnegative weight function $\theta : V \to \mathbb{R}_{\geq 0}$ over graph nodes based on embedding similarities. For filtered EDU nodes,

$$\theta(v_e) = \mathrm{sim}(z_q, h(e)), \quad \forall v_e \in \tilde{C}^{\mathrm{EDU}}(q),$$

and for filtered argument nodes,

$$\theta(v_a) = \mathrm{sim}(h(m), h(a)), \quad \forall v_a \in \tilde{C}^{\mathrm{arg}}(q),$$

where m is the query mention that retrieved a. All other nodes receive zero weight. If more than K argument nodes receive nonzero scores, we keep only the highest-scoring K nodes and set the rest to zero, ensuring that propagation remains focused and the initial mass is not diluted across many weakly related arguments. The resulting vector $\boldsymbol{\theta}$ is then used as the personalization (seed) vector for Personalized PageRank (Haveliwala, 2002).

Personalized PageRank. Let T be the row-stochastic transition matrix of G. We compute

$$\boldsymbol{\pi} = \mathrm{PPR}(G, \boldsymbol{\theta}) \triangleq (1-\alpha)\boldsymbol{\theta} + \alpha T^{\top}\boldsymbol{\pi},$$

where $\alpha \in (0, 1)$. The score $\pi(v)$ measures how strongly node v is connected to the query seeds under random walks that restart at $\boldsymbol{\theta}$.

EDU Selection. We restrict $\boldsymbol{\pi}$ to EDU nodes and select $R_{\mathrm{graph}}(q) = \mathrm{TopK}\{(e, \pi(v_e)) : e \in \mathcal{E}\}$. Structured chunk nodes are expanded to their full stored content before QA.

The final answer is

$$\hat{y} = f_{\mathrm{QA}}\Big(q,\ \{(\mathrm{text}(e), \mathrm{src}(e), \tau(e)) : e \in R_{\mathrm{graph}}(q)\}\Big).$$

4 Experiment

4.1 Experimental Setup

We evaluate EMem and EMem-G on two long-term conversational QA benchmarks: LoCoMo[4] (Maharana et al., 2024) and LongMemEval$_S$ (Wu et al., 2025a). LoCoMo contains 10 multi-session two-speaker dialogues averaging about 24K tokens, while LongMemEval$_S$ contains 500 human-assistant dialogues averaging about 105K tokens. Following prior work, we exclude adversarial question types. Category distributions and LongMemEval$_S$ query preprocessing are described in Appendix A.

We follow the Nemori evaluation framework (Ma et al., 2026),[5] using a `gpt-4o-mini` LLM judge to score final QA accuracy and reporting mean and standard deviation over three runs.[6] For LoCoMo, we also report F1 and BLEU-1. Baselines follow the Nemori setup: full-context prompting, dense retrieval over 4096-token chunks (RAG-4096), and three memory systems: LangMem (LangChain, 2025), Zep (Rasmussen et al., 2025), and Mem0 (Chhikara et al., 2025).

For EMem and EMem-G, we use `text-embedding-3-small` for embeddings and evalu-

[4]We map LoCoMo's raw labels as follows: Category 1 to *Multi-Hop*, Category 2 to *Temporal Reasoning*, Category 3 to *Open-Domain*, and Category 4 to *Single-Hop*, following a public discussion on the LoCoMo GitHub repository.

[5]The Nemori system, its evaluation framework, and the baseline numbers we adopt are those released with arXiv version 3 of Ma et al. (2026); later versions of that preprint use a different title.

[6]See the Nemori repository for implementation details, including the judge prompts.

		Full-context	Nemori	EMem	EMem-G
	Effective Context Length	101K	3.7K-4.8K	0.6K-2.5K	1.0K-3.6K
gpt-4o-mini	single-session-preference	6.7%	**46.7%**	32.2%	32.2%
	single-session-assistant	**89.3%**	83.9%	82.1%	87.5%
	temporal-reasoning	42.1%	61.7%	69.8%	**74.8%**
	multi-session	38.3%	51.1%	**78.0%**	73.6%
	knowledge-update	78.2%	61.5%	87.5%	**94.4%**
	single-session-user	78.6%	**88.6%**	86.5%	87.0%
	Average	55.0%	64.2%	76.0%	**77.9%**
gpt-4.1-mini	single-session-preference	16.7%	**86.7%**	46.7%	50%
	single-session-assistant	**98.2%**	92.9%	82.1%	87.5%
	temporal-reasoning	60.2%	72.2%	80.6%	**83.7%**
	multi-session	51.1%	55.6%	82.1%	**82.6%**
	knowledge-update	76.9%	79.5%	**95.4%**	94.4%
	single-session-user	85.7%	90.0%	93.8%	**94.8%**
	Average	65.6%	74.6%	83.0%	**84.9%**

Table 2: Results on LongMemEval$_S$ (Wu et al., 2025a) by question type. LLM-judged QA accuracy is reported for each category. Bold denotes the best result in each column.

ate with `gpt-4o-mini` and `gpt-4.1-mini` backbones. The synonym-edge threshold is $\delta = 0.9$. We initially retrieve top-$K_e = 30$ EDUs and top-$K_a = 10$ arguments, cap initialized argument nodes at 30, and retain up to $K = 10$ EDUs after propagation. For Personalized PageRank, we set the damping factor to 0.5 and use the PRPACK direct solver.

4.2 Main Results

Tables 1 and 2 show that our methods consistently outperform memory-based baselines while using comparable or fewer tokens. On LoCoMo with `gpt-4o-mini`, both EMem and EMem-G improve the overall LLM-judged score from Nemori's 0.744 to 0.780, with gains concentrated in temporal-reasoning, open-domain, and multi-hop questions. With `gpt-4.1-mini`, EMem-G reaches 0.853, exceeding both full-context prompting (0.806) and Nemori (0.794). These gains require much shorter QA contexts: EMem averages 738.2 tokens and EMem-G 987.8 tokens, compared with Nemori's 2,745 and full-context prompting's 23,653 tokens. Although Nemori remains slightly ahead in F1, our methods match or exceed it on BLEU-1 and achieve higher semantic correctness under the LLM judge. LongMemEval$_S$ further highlights the benefit of our methods in truly long conversations. With `gpt-4o-mini`, EMem and EMem-G reach 76.0% and 77.9% average accuracy, respectively, while reducing effective context from 101K tokens to at most a few thousand. The largest gains appear on temporal-reasoning, multi-session, and knowledge-update questions, where self-contained EDUs and recall-oriented filtering help locate and combine distant evidence. Nemori remains competitive on single-session preference and assistant-related questions, which depend more on local style and user habits than on event structure. This suggests that event-centric memory should eventually be paired with complementary user-profile modeling; a stage-level failure analysis (Appendix P) shows the single-session-preference gap arises at answer generation rather than in memory content or retrieval.

Comparing the two variants, graph propagation provides consistent but task-dependent gains. On LoCoMo, EMem and EMem-G are nearly tied with `gpt-4o-mini`, while EMem-G leads by about 1 point with `gpt-4.1-mini`. On LongMemEval$_S$, EMem-G is stronger on temporal-reasoning and multi-session questions that require dispersed evidence, whereas EMem remains competitive on questions dominated by a few highly relevant, recent EDUs. A case-level analysis of the 178 LoCoMo questions where the two variants disagree (Appendix H) attributes EMem-G's wins primarily to better PPR re-ranking and surfacing dispersed evidence, and EMem's wins most commonly to direct dense retrieval finding critical EDUs that PPR diffuses away from.

EDU Quality. We verify the EDU representation's atomic, self-complete, content-adaptive, and non-lossy properties via four analyses (full results in Appendix D and Appendix M): (i) an LLM-judge (`gpt-4o`) eval on 50 LoCoMo sessions scores coverage 4.36/5, entity normalization 4.68/5, and event-centric atomicity 3.38/5, with blind human annotation on 15 sessions agreeing within ± 1 point on 100 / 93 / 100% of sessions respectively; (ii) automatic gold-evidence recall is **98.0%** at the turn level (97.0% QA-level full coverage); (iii) a blind human granularity-fit annotation on 98 EDUs spanning five content-diverse sessions labels **95.9%** as appropriate with no over-decomposition observed; and (iv) a 569-EDU faithfulness annotation (three independent annotations per EDU, majority vote) finds **96.1%** of EDUs faithful to their attributed turns, with source-attribution bookkeeping rather than content the largest error class.

Efficiency Analysis. Table 3 compares effective context length and search latency on LoCoMo. EMem reaches the highest LLM-judged accuracy with only 223 tokens per query and 151 ms latency, a 93.7% token reduction and 46.8% speedup over Nemori. EMem-G matches this accuracy at 474 tokens and 234 ms. We additionally com-

Method	LLM Score	Tokens	Search (ms)
Mem0	0.613	329	740
Zep	0.585	1,093	491
Nemori	0.744	3,548	284
EMem	0.780	223	151
EMem-G	0.780	474	234

Table 3: Performance and efficiency comparison on LoCoMo. We use `gpt-4o-mini` as the base LLM and OpenAI `text-embedding-3-small` as the embedding model for all methods. Baseline LLM Scores follow Ma et al. (2026); baseline token counts and search latencies were measured in our own runs of each system under matched model settings during submission preparation. Absolute latencies are machine- and API-condition-dependent (Appendix L); Appendix S reports a per-stage breakdown.

pare against three recent SOTA memory systems (LightMem, LiCoMemory, MemOS) under identical settings in Appendix J, where EMem uses 8.2–12.1 $\times$ fewer per-query tokens and the shortest end-to-end wall-clock, and against three graph-memory systems (HippoRAG 2, A-Mem, Mem0's graph variant, reaching 0.741, 0.508, and 0.545 on LoCoMo) under a unified local protocol in Appendix Q. A per-stage latency and token breakdown is provided in Appendix S, and scaling to merged multi-conversation histories in Appendix R. Overall, the EDU representation plus recall-oriented filtering drives most gains; graph propagation adds headroom for relational and temporal integration. A controlled comparison against a Dense X-style propositionizer (Chen et al., 2024) on LoCoMo with an identical pipeline shows EDUs outperform propositions by **+6.6 pp** overall LLM Score (+11.5 pp temporal, +11.0 pp multi-hop) while using **2.1** $\times$ **fewer** units (Appendix E).

Graph statistics. After EDU abstraction, each conversation becomes a manageable graph with roughly 500–1,400 EDU nodes and 1,000–4,000 argument nodes; the graphs are sparse (mean argument degree 1.4–1.9), which suits Personalized PageRank since random walks stay concentrated near relevant clusters. Full statistics are in Appendix C.

4.3 Ablation Study

We ablate the main components of EMem and EMem-G on LoCoMo and LongMemEval$_S$ (Tables 4 and 5). On LoCoMo, removing the LLM-based EDU filter reduces the overall LLM score from 0.780 to 0.748 for EMem and 0.733 for EMem-G, with multi-hop accuracy dropping by 7–15 points. Removing query mention-to-argument seed initialization in EMem-G causes a smaller overall drop (0.780 $\rightarrow$ 0.766) but substantially harms multi-hop reasoning (0.747 $\rightarrow$ 0.675), while a named-entity-based mention detector largely recovers the gap. Dropping QA chain-of-thought has only a minor impact on LoCoMo.

On LongMemEval$_S$, the same components matter more. For EMem-G, removing the EDU filter or QA chain-of-thought lowers average accuracy from 77.9% to 73.1% and 73.4%, respectively, with multi-session accuracy dropping by up to 11.6 points. This likely reflects the many topic-wise similar EDUs produced from long sessions, which make filtering and explicit reasoning especially important. Removing the graph and PPR (EMem) lowers the average to 76.0%: graph propagation helps most on temporal-reasoning and knowledge-update questions, while EMem remains strong when dense EDU retrieval already surfaces the key evidence. Together, the ablations show that event-centric EDUs and recall-oriented filtering carry the main gains, with graph propagation and QA reasoning adding dataset-dependent benefits on harder long-term reasoning cases. To further isolate the EDU *representation* itself, we re-extract EDUs while removing either the self-completeness enrichment or the atomicity constraint, holding retrieval and QA fixed. Removing self-completeness drops LLM Score to 0.691 (an 11.4% relative drop from EMem in Table 1); removing atomicity drops it to 0.689 (11.7% relative). Graph component ablations are in Appendix G; single-property ablations that explicitly invert entity normalization, temporal grounding, or cross-turn synthesis are in Appendix N, and a chunk-node ablation on LongMemEval$_S$ is in Appendix O.

4.4 Hyperparameter Analysis

Retrieval. We vary the number of candidate EDUs retrieved before filtering (linking Top-K_e) for both EMem and EMem-G (Figure 4). Across LoCoMo and LongMemEval$_S$, performance is stable once K_e falls in the range of 20–40, with a mild optimum around 20–30 depending on the dataset. This indicates that dense retrieval plus recall-oriented filtering is robust: a moderately sized candidate pool is enough for the LLM filter to remove noisy EDUs while retaining relevant evidence. For EMem-G, increasing K_e also increases the possible number of argument seeds. Accuracy peaks around 30 on

	Temporal Reasoning			Open Domain			Multi-Hop			Single-Hop			Overall		
	LLM Score	F1	BLEU-1	LLM Score	F1	BLEU-1	LLM Score	F1	BLEU-1	LLM Score	F1	BLEU-1	LLM Score	F1	BLEU-1
EMem-G	$0.760_{\pm 0.003}$	0.581	0.468	$0.573_{\pm 0.013}$	0.242	0.199	$0.747_{\pm 0.006}$	0.406	0.305	$0.823_{\pm 0.001}$	0.504	0.422	$0.780_{\pm 0.000}$	0.487	0.397
w/o Query MD to node	$0.754_{\pm 0.006}$	0.581	0.468	$0.559_{\pm 0.009}$	0.276	0.236	$0.675_{\pm 0.007}$	0.405	0.305	$0.823_{\pm 0.002}$	0.507	0.419	$0.766_{\pm 0.001}$	0.490	0.397
w/ Query NED to node	$0.760_{\pm 0.001}$	0.580	0.462	$0.559_{\pm 0.009}$	0.275	0.226	$0.695_{\pm 0.002}$	0.407	0.302	$0.820_{\pm 0.000}$	0.505	0.420	$0.769_{\pm 0.001}$	0.489	0.395
w/o EDU Filter	$0.747_{\pm 0.003}$	0.565	0.458	$0.516_{\pm 0.000}$	0.252	0.201	$0.602_{\pm 0.007}$	0.357	0.244	$0.796_{\pm 0.001}$	0.502	0.420	$0.733_{\pm 0.002}$	0.473	0.382
w/o QA zero-shot CoT	$0.765_{\pm 0.004}$	0.595	0.485	$0.595_{\pm 0.005}$	0.258	0.208	$0.819_{\pm 0.001}$	0.413	0.308	$0.819_{\pm 0.001}$	0.529	0.443	$0.775_{\pm 0.003}$	0.505	0.413
w/o Graph & PPR (**EMem**)	$0.771_{\pm 0.004}$	0.574	0.461	$0.602_{\pm 0.009}$	0.285	0.237	$0.702_{\pm 0.004}$	0.406	0.307	$0.830_{\pm 0.002}$	0.497	0.414	$0.780_{\pm 0.001}$	0.483	0.393
w/o EDU Filter	$0.748_{\pm 0.005}$	0.567	0.462	$0.588_{\pm 0.005}$	0.268	0.221	$0.633_{\pm 0.010}$	0.363	0.252	$0.805_{\pm 0.002}$	0.502	0.417	$0.748_{\pm 0.003}$	0.476	0.384
w/o QA zero-shot CoT	$0.768_{\pm 0.004}$	0.596	0.476	$0.595_{\pm 0.005}$	0.291	0.244	$0.701_{\pm 0.003}$	0.415	0.307	$0.819_{\pm 0.001}$	0.520	0.431	$0.773_{\pm 0.001}$	0.503	0.407

Table 4: Ablation study on LoCoMo by question type. We use `gpt-4o-mini` as the base LLM and OpenAI `text-embedding-3-small` as the embedding model. LLM Score reports the mean over three judge runs, with standard deviation shown as a subscript.

	single-session-preference	single-session-assistant	temporal-reasoning	multi-session	knowledge-update	single-session-user	Average
EMem-G	32.2%	87.5%	74.8%	73.6%	94.4%	87.0%	77.9%
w/o Query MD to node	26.7%	83.3%	72.7%	73.3%	88.9%	88.5%	75.8%
w/ Query NED to node	26.7%	86.3%	76.1%	77.7%	91.7%	89.6%	78.8%
w/o QA zero-shot CoT	3.3%	87.5%	72.4%	68.3%	90.3%	86.5%	73.4%
w/o EDU Filter	10.0%	85.1%	74.5%	62.0%	90.3%	91.1%	73.1%
w/o Graph & PPR (**EMem**)	32.2%	82.1%	69.8%	78.0%	87.5%	86.5%	76.0%
w/o EDU Filter	26.7%	87.5%	70.1%	64.7%	84.7%	85.9%	72.4%
w/o QA zero-shot CoT	15.5%	85.7%	56.7%	66.1%	94.4%	92.2%	70.6%

Table 5: Ablation study on LongMemEval$_S$ by question type. We use `gpt-4o-mini` as the base LLM and OpenAI `text-embedding-3-small` as the embedding model. LLM-judged QA accuracy is reported.

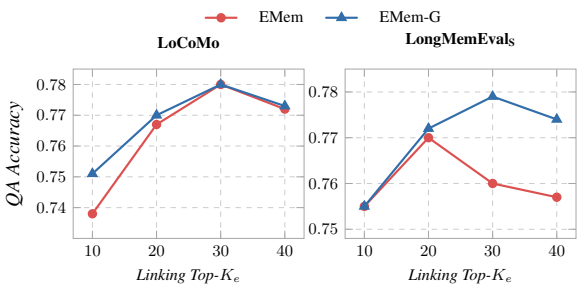

Figure 4: LLM-judged QA accuracy of EMem and EMem-G under different linking Top-K_e values. We use `gpt-4o-mini` as the base LLM and OpenAI `text-embedding-3-small` as the embedding model.

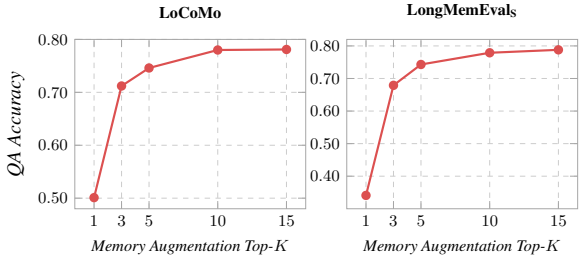

Figure 5: LLM-judged QA accuracy of EMem-G under different memory augmentation Top-K values. We use `gpt-4o-mini` as the base LLM and OpenAI `text-embedding-3-small` as the embedding model.

both datasets and degrades only slightly beyond that point, suggesting that graph propagation benefits from rich seeds but becomes mildly more sensitive to low-relevance noise.

A sweep of the synonym-edge threshold $\delta \in \{0.7, 0.8, 0.9, 0.95\}$ on LoCoMo shows $<2.5\%$ variance in LLM Score, indicating robustness to this hyperparameter (Appendix I).

Memory-Augmented Generation. We vary the QA Top-K, the number of EDUs passed to the QA model, for EMem-G while fixing $K_e = 30$ (Figure 5). Performance improves sharply from very small contexts (1–3 EDUs) to moderate ones (5–10 EDUs), then saturates around 10–15 EDUs. This confirms that strong performance can be achieved with small, fixed context budgets.

5 Conclusion

We introduced neo-Davidsonian EDUs, an event-centric memory representation for long-term conversational agents. By enriching EDUs with normalized entities, temporal grounding, and source metadata, the representation preserves session information in atomic, self-complete units that can be retrieved directly or organized into an event graph. Built on this representation, EMem combines dense EDU retrieval with recall-oriented LLM filtering, while EMem-G adds argument nodes and Personalized PageRank for associative recall. Experiments on LoCoMo and LongMemEval$_S$ show that both variants outperform memory-augmented base-

lines with substantially smaller context budgets and lower retrieval latency, with the largest gains on temporal-reasoning, multi-hop, and knowledge-update questions. The results suggest that event-level memory provides a practical foundation for long-horizon conversational QA, while leaving room for complementary profile or preference memories in less event-centric settings.

Limitations

First, our work focuses on the *representation* and *retrieval* of long-term conversational memory, and does not address dynamic memory-management operations such as merging redundant memories, deleting stale or low-value memories, or updating memories in place when facts change. This scope follows from our non-lossy design goal: instead of compressing or forgetting past content, we retain every extracted EDU with its session timestamp and source-turn attribution, and handle redundancy, conflicts, and knowledge updates at retrieval and QA time. Because each EDU is timestamped and self-contained, the QA model can reason over multiple, possibly conflicting, time-ordered EDUs to answer knowledge-update questions, consistent with the strong knowledge-update accuracy observed in our experiments. We therefore view explicit consolidation, such as deduplication, forgetting, and destructive updates for evolving facts, as complementary to our contribution and leave it to future work.

Second, our quality validation is deepest at the extraction stage: EDU quality and faithfulness are validated directly (Appendices D and M), whereas the intermediate LLM stages, mention detection and relevance filtering, are validated functionally through ablations (Tables 4 and 5) rather than through intrinsic per-stage accuracy measurements. Direct validation of these intermediate stages remains future work.

Third, evaluation on very long *coherent* histories is limited: the merged-history stress test (Appendix R) scales to thousands of sessions by combining independent conversations. Extending the evaluation to more coherent, extremely long-horizon benchmarks, including controlled comparisons with other memory systems at that scale, is left to future work due to the cost of the required API calls.

Ethics Statement

The non-lossy design indefinitely retains conversational content, which raises privacy and data-retention concerns in deployment. Deployments should ensure explicit user consent, bounded retention policies, and deletion rights. The EDU source-turn and timestamp metadata allow affected memory units to be located and removed, after which derived argument and synonym structures can be rebuilt from the remaining store.

Acknowledgements

Research was supported in part by the Institute for Geospatial Understanding through an Integrative Discovery Environment (I-GUIDE) by NSF under Award No. 2118329. The research has used the Delta/DeltaAI advanced computing and data resource, supported in part by the University of Illinois Urbana-Champaign and through allocation #250851 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by National Science Foundation grants OAC 2320345, #2138259, #2138286, #2138307, #2137603, and #2138296. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of the National Science Foundation.

We used general-purpose AI assistants (large language model based coding and writing tools) to support code refactoring, draft polishing, and literature search. All scientific content, experimental design, evaluation, and final wording were authored, verified, and endorsed by the human authors. AI assistants are not listed as authors. Note that the LLM components used inside our proposed system (`gpt-4o-mini`, `gpt-4.1-mini`, `text-embedding-3-small`) are the object of study and are documented in Section 4.1.

References

Nicholas Asher and Alex Lascarides. 2003. *Logics of conversation*. Cambridge University Press.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurovsky. 2001. Building a discourse-tagged corpus in the framework of Rhetorical Structure Theory. In *Proceedings of the Second SIGdial Workshop on Discourse and Dialogue*.

Tong Chen, Hongwei Wang, Sihao Chen, Wenhao Yu, Kaixin Ma, Xinran Zhao, Hongming Zhang, and

Dong Yu. 2024. Dense X retrieval: What retrieval granularity should we use? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15159–15177, Miami, Florida, USA. Association for Computational Linguistics.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. 2025. Mem0: Building production-ready ai agents with scalable long-term memory. In *European Conference on Artificial Intelligence*.

Donald Davidson. 2001. The logical form of action sentences. In *Essays on Actions and Events*. Oxford University Press.

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. 2026. Lightmem: Lightweight and efficient memory-augmented generation. In *The Fourteenth International Conference on Learning Representations*.

Bernal Jimenez Gutierrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. HippoRAG: Neurobiologically inspired long-term memory for large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From RAG to memory: Non-parametric continual learning for large language models. In *Forty-second International Conference on Machine Learning*.

Taher H. Haveliwala. 2002. Topic-sensitive pagerank. In *Proceedings of the 11th International Conference on World Wide Web*, WWW '02, page 517–526, New York, NY, USA. Association for Computing Machinery.

Zhengjun Huang, Zhoujin Tian, Qintian Guo, Fangyuan Zhang, Yingli Zhou, Di Jiang, Zeying Xie, and Xiaofang Zhou. 2026. LiCoMemory: Lightweight and cognitive agentic memory for efficient long-term reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 36842–36858, San Diego, California, United States. Association for Computational Linguistics.

Sangyeop Kim, Yohan Lee, Sanghwa Kim, Hyunjong Kim, and Sungzoon Cho. 2025. Pre-storage reasoning for episodic memory: Shifting inference burden to memory for personalized dialogue. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 22096–22113, Suzhou, China. Association for Computational Linguistics.

LangChain. 2025. LangMem SDK for agent long-term memory. `https://www.langchain.com/blog/langmem-sdk-launch`. Documentation: `https://langchain-ai.github.io/langmem/`.

Kuang-Huei Lee, Xinyun Chen, Hiroki Furuta, John Canny, and Ian Fischer. 2024. A human-inspired reading agent with gist memory of very long contexts. In *Forty-first International Conference on Machine Learning*.

Zhiyu Li, Shichao Song, Hanyu Wang, Simin Niu, Ding Chen, Jiawei Yang, Chenyang Xi, Huayi Lai, Jihao Zhao, Yezhaohui Wang, Junpeng Ren, Zehao Lin, Jiahao Huo, Tianyi Chen, Kai Chen, Kehang Li, Zhiqiang Yin, Qingchen Yu, Bo Tang, and 3 others. 2025. Memos: An operating system for memory-augmented generation (mag) in large language models. *Preprint*, arXiv:2505.22101.

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

Wenquan Ma, Jiayan Nan, Wenlong Wu, and Yize Chen. 2026. What deserves memory: Adaptive memory distillation for llm agents. *Preprint*, arXiv:2508.03341.

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

William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text - Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281.

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

Terence Parsons. 1990. *Events in the Semantics of English*. MIT Press.

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

Juyuan Wang, Rongchen Zhao, Wei Wei, Yufeng Wang, Mo Yu, Jie Zhou, Jin Xu, and Liyan Xu. 2026. Comorag: a cognitive-inspired memory-organized rag for stateful long narrative reasoning. In *Proceedings of the Fortieth AAAI Conference on Artificial Intelligence and Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence and Sixteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'26/IAAI'26/EAAI'26. AAAI Press.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V Le,

and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

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

Yaxiong Wu, Yongyue Zhang, Sheng Liang, and Yong Liu. 2025b. Sgmem: Sentence graph memory for long-term conversational agents. *Preprint*, arXiv:2509.21212.

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

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2024. Memorybank: enhancing large language models with long-term memory. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press.

Sizhe Zhou, Sheldon Yu, Hui Wei, Junda Wu, Siru Ouyang, Yizhu Jiao, Shijia Pan, Julian McAuley, Yu Zhang, Tong Yu, and Jiawei Han. 2026. Filesystem-based memory for llm agents: Organization, evolution, and sustainability. *Preprint*, arXiv:2607.26637.

A Dataset Statistics

We conduct experiments on two long-term conversational QA benchmarks. LoCoMo (Maharana et al., 2024) contains 10 multi-session dialogues between two speakers and covers Multi-Hop, Temporal Reasoning, Open-Domain, and Single-Hop questions. LongMemEval$_S$ (Wu et al., 2025a) contains 500 multi-session user-assistant dialogues spanning six question categories. Following prior work, we exclude adversarial question types. Table 6 summarizes the evaluated category distributions. The LoCoMo counts in Table 6 are the canonical scored set: of the raw file's 1,540 category 1–4 questions, 9 whose evidence pointers are broken in the source data are excluded at loading (1,531 processed), and 11 duplicated copies of questions within one conversation are collapsed at scoring, yielding 1,520 unique questions. We use 1,520 as the canonical QA denominator throughout; analysis-specific denominators are footnoted where they appear (Appendices D and F). For LongMemEval$_S$, excluding the 30 abstention questions leaves 470 evaluated questions out of 500. Its questions are assumed to be asked by the user. During QA, we prepend each question with its timestamp and the prefix "`User: `". For example: "`Date of user query: 2023/05/30 (Tue) 21:54\nUser: How old was I when my grandma gave me the silver necklace?`".

LoCoMo		**LongMemEval$_S$**	
Category	**Count**	**Category**	**Count**
Multi-Hop	278	knowledge-update	72
Temporal	320	multi-session	121
Open Domain	93	single-session-assistant	56
Single-Hop	829	single-session-preference	30
–	–	single-session-user	64
–	–	temporal-reasoning	127
Total	**1,520**	**Total**	**470**

Table 6: Category distributions of the LoCoMo (Maharana et al., 2024) and LongMemEval$_S$ (Wu et al., 2025a) benchmarks.

B Dataset-Specific EDU Extraction for LongMemEval$_S$

LongMemEval$_S$ consists of user-assistant dialogues with asymmetric turn styles: user turns are usually short and focused, while assistant turns are often long structured responses, such as enumerated lists, step-by-step plans, comparisons, and recommendations. When relatively small LLMs such as `gpt-4o-mini` extract EDUs from these long assistant turns, they tend to omit details that appear locally unimportant but may later be queried. We therefore apply a speaker-specific adaptation without changing the core EMem/EMem-G retrieval algorithms.

User Utterances. For user turns, we directly apply the generic EDU extraction procedure from Section 3.1. Each session is passed to g_{EDU} with timestamps and speaker tags, and the model emits EDUs with source-turn attributions.

Assistant Utterances. Each assistant turn is processed in two parallel views:

- **Atomic EDUs.** The extractor produces fine-grained EDUs, each with its source turn index, to capture localized facts and events.
- **Structured Chunks.** The extractor also identifies cohesive information blocks and generates a

2–3 sentence summary $s(c)$ stating the user request addressed, information categories covered, and salient entities.

We treat each chunk as an additional EDU node. Its summary $s(c)$ is used as text(e) for argument extraction, indexing, and retrieval. The full chunk is stored separately and revealed only to the QA model when the chunk node is retrieved. Atomic EDUs and chunks share the same metadata schema, including source turns and session timestamps. This adaptation improves recall on LongMemEval$_S$ while keeping the memory graph short and information-dense.

C Graph Statistics

Table 7 reports detailed graph statistics for EMem-G on both LoCoMo and LongMemEval$_S$. Both benchmarks are structurally challenging: conversations span dozens of sessions and include long turns, especially in LongMemEval$_S$ where assistant responses are often lengthy. Moving from `gpt-4o-mini` to `gpt-4.1-mini` substantially increases the EDU and argument counts on LongMemEval$_S$ but only modestly changes them on LoCoMo, suggesting that stronger extractors matter most when turns are long and heterogeneous. The graphs remain sparse: each EDU connects to a small number of arguments, argument nodes typically have degree 1 or 2, and synonym edges are relatively rare. This locality suits Personalized PageRank because random walks stay concentrated near relevant clusters rather than diffusing through dense neighborhoods, while also suggesting that more normalized and atomic argument extraction could yield additional useful connectivity for graph-based retrieval. Speaker-wise statistics show that memory units are manageable but role-imbalanced: LongMemEval$_S$ is dominated by assistant EDUs and chunks, while LoCoMo is dominated by the speaker who contributes most factual content.

D EDU Quality Analysis

We supplement the in-text summary with three complementary analyses verifying the atomic, self-complete, content-adaptive, and non-lossy properties of extracted EDUs.

Annotation protocol. All human annotations in this paper (this appendix and Appendices H, M, and P) were carried out by the authors; no crowdworkers or other external participants were recruited or paid. All experiments use the publicly released LoCoMo and LongMemEval$_S$ benchmarks, and no data was collected from human participants. The rubrics and label definitions are stated where each annotation is reported.

LLM-judge with human validation. An LLM-judge (`gpt-4o`) scores 50 randomly sampled LoCoMo sessions (five per conversation; backbone for extraction is `gpt-4o-mini`) on three rubrics scored 1–5: *coverage* (do EDUs collectively capture all factual information from the session?), *entity normalization* (are entities consistently named, with pronouns resolved?), and *atomicity* in the neo-Davidsonian sense (does each EDU express exactly one main event together with its participants, time, location, and circumstances? multi-attribute single events are not penalized). We also collect blind human annotations on 15 of these sessions to validate the LLM-judge. Results are shown in Table 8.

All 50 sessions score ≥ 4 on coverage, supporting the event-level non-lossy claim. Entity normalization is high (70% of sessions perfect 5/5, 28% at 4/5). Atomicity averages 3.38/5 under the event-centric rubric; remaining penalties primarily come from cases where two adjacent happenings are grouped into one unit. Human means slightly exceed LLM-judge means on all three rubrics, suggesting the LLM-judge is a conservative estimator rather than over-generous.

Gold-evidence recall. For every LoCoMo QA pair, we check whether each gold evidence turn appears in the source-turn IDs of at least one extracted EDU. Across 1,536 QA pairs[7] and 2,349 gold evidence turns, the turn-level recall is **98.0%** (2,301 / 2,349) and QA-level full coverage (all gold turns covered by at least one EDU) is **97.0%** (1,490 / 1,536). Per-category recall is 97.7%, 97.6%, 98.5%, and 98.2% for Multi-Hop, Temporal, Open-Domain, and Single-Hop, respectively. This is structural coverage (turn attribution), not semantic coverage; together with the LLM-judge coverage rubric above, it provides strong evidence for event-level non-lossy extraction.

Content-adaptive granularity. To test whether EDU granularity is content-adaptive (rather than uniformly finer-grained), we sample 98 EDUs

[7] This analysis covers the 1,540 category 1–4 QA pairs minus the 4 with empty evidence fields, independent of the loader-level exclusions behind the canonical 1,520 (Appendix A).

	LongMemEval$_S$ (GPT-4o-mini)	LongMemEval$_S$ (GPT-4.1-mini)	LoCoMo (GPT-4o-mini)	LoCoMo (GPT-4.1-mini)
Number of Conversations	470	470	10	10
Avg Sessions/Conv	47.7	47.7	27.2	27.2
Avg Session Length (words)	1,644.4	1,644.4	536.9	536.9
Avg Turns/Session	10.3	10.3	21.6	21.6
Avg EDU Nodes/Conv	861.1	1391.5	552.8	570.2
Avg Arg Nodes/Conv	2,780.9	3,786.2	1,144.7	1,184.9
Avg Total Nodes/Conv	3,689.7	5,225.4	1,724.7	1,782.3
Avg Session Node Degree	18.1	29.2	20.3	21.0
Avg EDU Node Degree	5.5	4.7	4.9	4.7
Avg Arg Node Degree	1.5	1.4	1.9	1.8
Avg Session-EDU Edges/Conv	861.3	1391.8	552.9	570.3
Avg EDU-Arg Edges/Conv	3,908.3	5,106.4	2,149.9	2,100.3
Avg Synonym Edges/Conv	105.8	189.0	24.5	38.7
Avg Total Edges/Conv	4,875.4	6,687.1	2,727.3	2,709.3
Avg User EDUs/Conv	314.6	469.2	–	–
Avg Asst EDUs/Conv	421.6	725.6	–	–
Avg Asst Chunks/Conv	125.9	197.7	–	–
Avg Speaker EDU or Chunk Dist/Conv	314.6:421.6:125.9	469.2:725.6:197.7	98.1:16.7	105.9:18.5
Avg Speaker EDU or Chunk Len (words) Dist	23.2:21.3:193.8	20.3:21.1:121.3	15.0:15.4	16.9:17.9
Avg Asst Chunk Summary Len (words)	43.7	40.1	–	–

Table 7: Statistics of the graphs constructed by EMem-G. All metrics except the first row are averaged per conversation. For LongMemEval$_S$, "Avg Speaker EDU or Chunk Dist/Conv" and "Avg Speaker EDU or Chunk Len (words) Dist" report the average count and average word length, respectively, in the format "User EDUs:Assistant EDUs:Assistant Chunks". For LoCoMo, the two speakers in each conversation are ordered by EDU count, and results are reported as "higher-EDU speaker:lower-EDU speaker".

Rubric	LLM mean	Human mean	Agreement
Coverage	4.36	4.53	100%
Entity normalization	4.68	4.73	93%
Atomicity (event-centric)	3.38	4.00	100%

Table 8: EDU quality on LoCoMo. LLM-judge means are over 50 sessions; human means are blind annotations on 15 sessions. Agreement is the fraction of sessions where the LLM-judge and human differ by at most one point.

across five LoCoMo sessions chosen for content diversity (multi-hop, temporal, open-domain, single-hop, and one mixed session) and ask a blind human annotator to label each EDU as *appropriate*, *should-split* (bundles independent events), *should-merge* (over-decomposed), or *unclear*. The annotator labels **95.9%** (94/98) as *appropriate*, with the four errors all being *should-split* cases (the multi-hop, temporal, and open-domain sessions are at 100%); no *should-merge* cases are observed in the sample. The absence of over-decomposition supports that the extractor does not uniformly produce finer-grained units; granularity adapts to the local content rather than following a fixed rule.

Across 272 LoCoMo sessions our extractor produces an average of 20.2 EDUs per session (range 9–46, standard deviation 5.7), a 5 $\times$ span confirming content-adaptive variability.

E Comparison with Dense X Propositions

To isolate the contribution of our event-centric, content-adaptive EDU representation over a generic atomic-proposition representation, we run a controlled comparison against a Dense X–style propositionizer (Chen et al., 2024) on LoCoMo, using the same retrieval and QA pipeline for both (dense cosine retrieval over text-embedding-3-small embeddings, top-K=10, no LLM filter to avoid confounds, `gpt-4o-mini` for both extraction and QA, same generous LLM-judge prompt). The two methods differ only in the unit-construction prompt: Dense X uses the standard "decompose into independent, minimal, self-contained, decontextualized propositions" prompt, while EDUs use the standard EMem extraction prompt that additionally enforces event-centric atomicity, normalized entities, temporal grounding, and cross-turn synthesis. Results are summarized in Table 9.

Category	Dense X	EDU	Δ
Multi-Hop	0.578	**0.688**	+11.0pp
Temporal	0.355	**0.470**	+11.5pp
Open-Domain	0.406	**0.427**	+2.1pp
Single-Hop	0.713	**0.752**	+3.8pp
Overall LLM Score	0.595	**0.661**	**+6.6pp**
Total units	16,567	**7,723**	2.1× fewer
Avg context tokens / query	**121**	160	+39

Table 9: Dense X propositions vs our EDUs on LoCoMo with an identical retrieval+QA pipeline. LLM Score is the same gpt-4o-mini judge used in Table 1.

EDUs outperform Dense X propositions by **+6.6 pp** overall, with the largest gains on temporal reasoning (+11.5 pp) and multi-hop (+11.0 pp) – exactly where the event-centric binding of participants, time, and location should matter most. EDUs achieve this with **2.1 × fewer units** (7,723 vs 16,567), so the gain does not come from finer slicing. The slightly larger per-query context (160 vs 121 tokens) reflects EDUs' enriched, self-complete form. Dense X propositions, despite their self-contained design goal, fragment cross-turn content into too many small pieces; without event-binding or temporal grounding, retrieval struggles on questions that require integrating who-did-what-when-where across turns.

F F1 vs LLM-Judge Reliability

Standard token-overlap metrics (F1, BLEU-1) systematically underestimate performance on LoCoMo because gold answers vary widely in form (date format, paraphrase, level of detail). We quantify this on our LoCoMo EMem run: of 1,531 evaluated QA pairs[8], the LLM-judge marks 453 as semantically CORRECT but F1 is below 0.5 on those same pairs (**29.6%** of all questions). The reverse case (LLM=WRONG but F1>0.5) occurs only 23 times (1.5%). Concrete examples are shown in Table 10.

These failure modes – date paraphrasing, semantic equivalence with different word choice, and informative answers being penalized via lower precision – are inherent to token-overlap metrics on free-form QA. We therefore follow Ma et al. (2026) and report LLM Score as our primary metric while still showing F1 and BLEU-1 for comparability with prior work.

[8] 1,531 is the loader-processed set (1,540 minus 9 broken evidence pointers); the canonical scored set collapses 11 duplicates to the 1,520 unique questions of Table 6 (Appendix A).

G Representation and Graph Component Ablations

Representation ablations. To test whether the EDU *representation* itself drives performance (rather than the retrieval pipeline), we hold retrieval, filtering, and QA fixed and re-extract EDUs with two perturbations: (1) *no atomicity*, allowing each EDU to span multiple events and turns; (2) *no self-completeness*, removing entity normalization, coreference resolution, temporal grounding, and cross-turn synthesis. Results on LoCoMo with `gpt-4o-mini` are in Table 11.

Both perturbations cause similarly large drops (more than 11% relative), confirming that the joint combination of atomicity and self-completeness, rather than either constraint alone, drives EMem's advantage.

Graph component ablations. To isolate each graph component's contribution to EMem-G, we remove (a) synonym edges between similar argument nodes, (b) all argument nodes and their incident edges, leaving session-EDU edges only, and (c) session nodes and session-EDU edges. Removing each component independently changes LLM Score by less than 1 pp (Table 12), consistent with the observation that self-complete EDUs already encode most local context, so the graph adds value primarily when evidence is genuinely dispersed across sessions (e.g., temporal reasoning and multi-session questions on LongMemEval$_S$, where PPR contributes meaningful gains as reported in the main results).

H Case-Level Analysis: EMem vs EMem-G

To understand *when* graph propagation helps versus hurts on LoCoMo, we manually inspect the 178 LoCoMo questions on which EMem and EMem-G disagree (91 where EMem-G is correct and EMem is not, 87 where EMem is correct and EMem-G is not). For each case we examine both systems' retrieved EDUs and predictions and assign a primary reason.

For EMem-G's wins, the dominant patterns are **better PPR re-ranking** of evidence already retrieved by both systems (42.9%), **surfacing dispersed evidence** that dense retrieval missed entirely (33.0%), and **noise reduction** where graph propagation yields a cleaner final context (12.1%). For EMem's wins, the dominant patterns are PPR

Question	Gold	Prediction	F1	LLM
When did Caroline have a picnic?	The week before 6 July 2023	29 June 2023	0.222	CORRECT
What fields would Caroline pursue?	Psychology, counseling certification	Counseling and mental health	0.286	CORRECT
Would Caroline pursue writing?	Likely no; though she likes reading, she wants to be a counselor	No, Caroline is focused on counseling.	0.000	CORRECT
What is Caroline's identity?	Transgender woman	Caroline is transgender.	0.400	CORRECT

Table 10: Examples of LLM-judge CORRECT with low F1 on LoCoMo. F1 uses the standard simple-tokenize, set-based comparison used by prior memory-system evaluations.

Variant	LLM Score	F1	Δ_{rel}
EMem w/o atomicity	0.689	0.441	-11.7%
EMem w/o self-completeness	0.691	0.437	-11.4%

Table 11: Representation ablations of EMem on LoCoMo (`gpt-4o-mini`). Δ_{rel} is the relative change in LLM Score with respect to EMem in Table 1 (0.780).

Variant	LLM Score	Δ
EMem-G w/o synonym edges	0.776	-0.4 pp
EMem-G w/o argument nodes	0.774	-0.6 pp
EMem-G w/o session nodes	0.772	-0.8 pp

Table 12: Graph component ablations of EMem-G on LoCoMo (`gpt-4o-mini`). Δ is the change in LLM Score with respect to EMem-G in Table 1 (0.780).

missing a critical EDU that dense retrieval directly finds (32.2%), graph drift or context dilution (combined 12.6%), and the LLM filter producing a cleaner candidate set than PPR (9.2%). The remaining cases are mostly heterogeneous and not informative on their own.

Two implications: (i) the graph's main value is *re-ranking and dispersed-evidence retrieval*, not multi-hop reasoning per se – both systems usually retrieve enough evidence; (ii) PPR can occasionally diffuse away from the relevant cluster, so self-complete EDUs and an LLM filter remain important even with the graph.

I Synonym-Edge Threshold Sensitivity

The paper uses $\delta = 0.9$. We additionally sweep $\delta \in \{0.7, 0.8, 0.95\}$ on LoCoMo with `gpt-4o-mini`. All three settings produce LLM Scores within 1.5 pp of the paper baseline (Table 13), with the widest gap being 1.4 pp at $\delta = 0.7$. This indicates that the design is not overfitted to the specific choice of δ.

δ	LLM Score	F1	Δ
0.7	0.766	0.485	-1.4 pp
0.8	0.785	0.495	$+0.5$ pp
0.95	0.773	0.491	-0.7 pp

Table 13: Sensitivity of EMem-G to the synonym-edge threshold δ on LoCoMo (`gpt-4o-mini`). Δ is the change in LLM Score with respect to the paper's $\delta = 0.9$ baseline in Table 1 (0.780).

J Comparison with Recent SOTA Memory Systems

To complement Table 3, we compare EMem and EMem-G against three recent SOTA memory systems on LoCoMo under identical settings (`gpt-4o-mini` backbone, `text-embedding-3-small` embeddings, the same machine, and the same LLM-judge evaluation framework). Results are in Table 14.

EMem and EMem-G achieve the highest LLM Score (0.780) while using 8.2–15.9 $\times$ fewer per-query tokens than the four baselines, the shortest indexing wall-clock (5–6 min vs 22–203 min), and the shortest end-to-end wall-clock (17–22 min vs 77–349 min). The combination of low per-query tokens, short indexing time, and short end-to-end wall makes our approach practical for production deployment, even on smaller-context backbones.

K Storage and Scalability

Per-session storage. On LoCoMo, the persistent storage footprint of EMem is approximately 245 KB per session, dominated by 1,536-dimensional EDU text embeddings (text-embedding-3-small) – comparable in order of magnitude to a simple per-turn embedding store. EMem-G adds argument embeddings and a small graph file, for a total of approximately 780 KB per session. Re-measurement on independently repro-

Method	LLM Score	Tokens/query	Indexing (min)	End-to-end (min)
Nemori (Ma et al., 2026)	0.744	3,548	92	99
LightMem (Fang et al., 2026)	0.722	2,691	30	77
LiCoMemory (Huang et al., 2026)	0.685	1,868	22	135
MemOS (Li et al., 2025)	0.732	1,824	203	349
EMem	**0.780**	**223**	**5**	**17**
EMem-G	**0.780**	474	6	22

Table 14: EMem and EMem-G vs recent SOTA memory systems on LoCoMo with identical settings. Indexing wall is the total wall-clock for processing all 10 LoCoMo conversations; end-to-end wall additionally includes the QA phase over the 1,540 asked questions (scored on the canonical 1,520 questions; Appendix A). LLM Scores and token counts for Nemori, EMem, and EMem-G are quoted from Tables 1 and 3; all other entries were measured in our runs on a single machine (Appendix L), with the three re-run systems scored by the same three-run judge.

duced stores confirms both figures within 1 KB (246 and 783 KB per session), corresponding to approximately 2.3 and 7.5 GB per 10,000 sessions; because embeddings dominate, storage grows approximately linearly with the number of extracted units.

PPR scaling. We report PPR cost at two granularities, measured on the stored per-conversation graphs with the production solver configuration (PRPACK, damping 0.5, 30-seed personalization vectors, 20 repetitions per graph; measured on the MacBook Pro described in Appendix L, and absolute times are machine-dependent). On LoCoMo, where graphs average roughly 1.7K nodes, a single PPR pass takes about 0.26 ms per query, accumulating to about 0.39 s over the 1,520 evaluated questions. On LongMemEval$_S$ with the `gpt-4o-mini` extractor, graphs average roughly 3.7K nodes (at most about 4.5K); a pass takes about 0.52 ms (at most 0.64 ms), accumulating to about 0.24 s over the 470 evaluated questions. Graphs built with `gpt-4.1-mini` average 5.2K nodes (Table 7), and per-pass cost grows near-linearly with graph size: on merged multi-conversation graphs it is about 1.2 ms at 5.9K nodes and 2.6 ms at 12.5K nodes (see also Appendix R).

L Implementation and Configuration Details

Graph construction and PPR. The event graph is undirected. Session–EDU and EDU–argument edges have weight 1.0; synonym edges connect argument nodes whose embedding cosine similarity is at least $\delta = 0.9$, with at most 100 synonym neighbors per argument. The weighted adjacency matrix is normalized by weighted degree. Personalized PageRank uses igraph's PRPACK solver with damping factor 0.5. The personalization vector places the filtered EDU scores $\mathrm{sim}(z_q, h(e))$ and filtered argument scores $\mathrm{sim}(h(m), h(a))$ on the corresponding seed nodes, capping nonzero argument seeds at the top 30 by score, and zero elsewhere.

Prompts, exemplars, and decoding. Extraction, mention detection, and filtering use fixed, hand-written one-shot exemplars that are identical for every session; there is no dynamic exemplar selection. All decoding uses temperature 0, seed 42, and at most 8,192 output tokens, and every pipeline call requests its JSON response schema through the API's structured-output support. The exact model snapshots are `gpt-4o-mini-2024-07-18` and `gpt-4.1-mini-2025-04-14`, with `text-embedding-3-small` for all embeddings. All prompts are reproduced verbatim in Appendix T, and the released implementation includes every prompt and configuration.

LLM judging. The judge is `gpt-4o-mini` at temperature 0, run three times, with mean and standard deviation reported. LoCoMo uses the standard judge prompt employed verbatim in the public Nemori and Mem0 evaluations; LongMemEval$_S$ uses the benchmark's category-specific judge prompts (temporal reasoning, knowledge update, single-session preference, and the default case) as implemented in the Nemori evaluation framework, which keeps the official instruction text verbatim and wraps the question and response in tagged fields with a structured yes/no output. Judge-run variation reflects API nondeterminism.

Alias handling across sessions. Entity normalization occurs within each session, where the extractor sees the full context, emits canonical names, resolves pronouns, and grounds dates. Across sessions, argument nodes are hard-unified only on

exact normalized-string matches; aliases, partial names, and paraphrases are instead connected by embedding-based synonym edges, over which PPR propagates relevance. We avoid hard cross-session alias merging because false merges irreversibly corrupt indexed memory, whereas soft links let retrieval, filtering, and QA arbitrate identity at query time. Consistently, removing entity/coreference normalization at extraction lowers the overall LoCoMo score from 0.772 to 0.601 in the matched run of Appendix N, whereas removing synonym edges costs only 0.4 points (Table 12); most of the benefit comes from within-session normalization, with soft cross-session links adding a smaller, safer increment. A persistent cross-session entity registry remains future work.

Compute environments. Experiments were run at different times on four machines. (i) The main evaluations behind Tables 1 and 2 ran on a 144-core aarch64 Linux server with 238 GiB of memory. (ii) The efficiency figures in Table 3 were measured during submission preparation on a second Linux server (two 32-core AMD EPYC 7502 processors, 1 TiB of memory): EMem and EMem-G with the released efficiency-evaluation harness, and the baseline token counts and search latencies in our own runs of each system under matched model settings, while the baseline LLM Scores follow Ma et al. (2026) as in Table 1. (iii) The cross-system comparison in Appendix J was measured end to end on a third Linux server (Intel Xeon E5-2650 v4, 24 logical CPUs, 251 GiB of memory, NVIDIA GTX 1080 and 1080 Ti GPUs; our own runs disabled CUDA and used API-hosted models, whereas LightMem's local compression model ran on the GPUs). (iv) The PPR timings in Appendix K and the analyses in Appendices M–S were measured on an Apple MacBook Pro (Apple M5 Pro, 18 cores, 64 GB of memory). Within each table, machine-dependent quantities (latency and wall-clock) come from a single machine; absolute times depend on hardware and API conditions and are not comparable across tables, whereas accuracy metrics, token counts, and annotation statistics are machine-independent. Unmodified re-runs of EMem and EMem-G performed alongside the later analyses reproduce Table 1 within 0.8–1.6 LLM Score points (three-run and single-run judging, respectively), with F1 and BLEU-1 matching within 0.006; we therefore quote Tables 1–3 values for EMem and EMem-G as printed and label analysis-specific reference rows explicitly.

M EDU Faithfulness and Downstream Effects of Atomicity Imperfections

Appendix D measures recall-oriented quality; this section measures per-EDU precision (faithfulness) and traces the downstream effect of atomicity imperfections. The analyses run on an independently reproduced extraction of the paper configuration, which re-derives Appendix D's recall within 0.5 points (97.9% turn-level recall and 96.5% QA-level coverage, vs the printed 98.0% and 97.0%).

Faithfulness annotation. We annotated 569 unique EDUs: 150 random LoCoMo EDUs, 150 random LongMemEval$_S$ user-side EDUs, and 275 LoCoMo EDUs covering gold-evidence turns, with six overlaps across strata. Each EDU was judged by three independent annotators against its attributed turns with the full session visible (majority vote; 98.4% of verdicts unanimous). The annotation schema covers omitted qualifications, merged distinct facts, unsupported details, misstated content, and incorrect source attribution. Results are in Table 15.

Population	EDUs	Faithful
LoCoMo, random	150	96.7%
LongMemEval$_S$ user-side, random	150	92.7%
LoCoMo, covering gold evidence	275	97.8%
All unique EDUs	**569**	**96.1%**

Table 15: EDU faithfulness by population (majority vote of three independent annotations per EDU).

Among the 22 majority-unfaithful EDUs, 10 contain correct content but cite the wrong turns, 9 misstate a detail, and 5 exhibit an abstraction-risk mode: omitted qualification (2), merged distinct facts (2), or unsupported detail (1); two EDUs carry two labels. Content-level defects therefore occur in 12/569 EDUs (2.1%), and source-attribution bookkeeping is the largest single error class. This complements the recall evidence of Appendix D.

Bundling prevalence. Under a strict criterion of independent facts, 2/569 EDUs (0.4%) received a majority bundling flag and 5/569 (0.9%) were flagged by at least one annotator; the blind should-split annotation of Appendix D identified 4/98. Bundling is thus uncommon under either criterion, consistent with the observation in Appendix D that the LLM judge's 3.38/5 atomicity score is conser-

vative (blind human annotation on the 15-session subset averages 4.00/5).

Downstream propagation. For the 275 gold-evidence EDUs, we joined the annotation with three downstream outcomes under `gpt-4o-mini`: membership in the dense top 30, membership in the final QA context, and answer correctness (Table 16).

Only one majority-bundled EDU covers gold evidence; it is linked to two questions, giving two EDU–question pairs per variant. It reached the dense top 30 and the final context for both questions under both variants, and 3 of the 4 question–variant cases were answered correctly, so in the observed cases bundling did not suppress recall. On the precision side, this EDU appeared in only 5 of 153 final contexts per variant, and the three non-linked questions among those contexts were answered correctly under both variants. At the measured prevalence (at most 0.9% of EDUs), even if every question whose gold evidence lies in a bundled EDU were answered wrongly, the impact on overall accuracy would be below one point; the observed cases show no failure concentration. This analysis is descriptive rather than causal: the affected sample is small and bundling is not randomly assigned.

Attribution coverage vs answer accuracy. The 97.0% QA-level figure of Appendix D measures source-turn citation coverage, not lost content. On the reproduction, the 45 questions with an uncited gold turn are answered at least as accurately as fully covered ones (0.800 vs 0.769 for EMem-G; 0.778 vs 0.762 for EMem; small-sample and descriptive). Manual inspection shows that the same content typically appears in another EDU with different attribution, consistent with cross-turn synthesis.

N Single-Property Extraction Ablations

Table 11 removes bundled instruction sets from the extraction prompt. To attribute the gains to individual representation properties, we ran three stronger interventions, each explicitly *inverting* exactly one property in the extraction prompt and its one-shot exemplar (e.g., instructing the extractor *not* to resolve pronouns), while holding retrieval, filtering, QA, and the three-run judge at the paper configuration (EMem mode; run-to-run std below 0.3 points; 1,520 questions). Entity naming and pronoun resolution are ablated jointly because the extractor implements them as one normalization operation. Results are in Table 17.

Each property is individually load-bearing, with overall drops of 9.7 to 17.1 points within the run. The damage localizes where each property should matter: removing temporal grounding collapses temporal reasoning from 0.746 to 0.208 while barely affecting multi-hop; removing entity/coreference normalization harms every category, consistent with unresolved references weakening embedding matches; removing cross-turn synthesis primarily harms temporal and multi-hop reasoning, the categories that need integrated facts. These interventions are not directly comparable to Table 11: that ablation *omits* instructions, which the base model may still follow implicitly, whereas these explicitly invert one property each.

O Chunk-Node Ablation on LongMemEval$_S$

Because LongMemEval$_S$ contains long structured assistant responses represented by both atomic EDUs and structured chunks (Appendix B), we ablate retrievable node types with identical dense retrieval, LLM filtering, and QA (EMem mode; extraction shared via cache; official category-specific judge prompts, three runs, row std at most 0.2 points; 470 questions). User turns have no chunk representation, so the chunk-only condition retains user EDUs and removes assistant EDUs. Results are in Table 18.

The two representations play complementary roles. Chunks are load-bearing for exactly one category: single-session-assistant collapses from 79.8 to 42.9 without them, since they preserve long assistant responses verbatim. Assistant EDUs carry the rest: without them, multi-session, knowledge-update, and single-session-user all dip. Retrieving larger structured blocks alone does not explain the gains: chunk-only reaches 75.2 and EDU-only 73.2, while their combination reaches 77.3. Because the three node populations compete for the same top-30 pool and context slots, these effects are marginal rather than additive; temporal accuracy, for instance, moves by at most 1.8 points under either ablation. The single-session-preference cells are small-sample (30 questions) and volatile (Appendix P).

Group	n	Top 30	Final ctx.	Correct
Clean gold EDUs, EMem-G	469	72.1%	61.4%	0.733
Clean gold EDUs, EMem	469	72.1%	53.9%	0.719
Bundled gold EDUs, EMem-G	2	100%	100%	1 of 2
Bundled gold EDUs, EMem	2	100%	100%	2 of 2

Table 16: Downstream outcomes of gold-evidence EDUs (EDU–question pairs) on the reproduced runs.

Variant	Overall	Temp.	M-Hop	Open-D.	S-Hop
EMem (reproduction reference)	0.772	0.746	0.724	0.595	0.817
w/o entity/coref. normalization	0.601	0.308	0.604	0.462	0.729
w/o temporal grounding	0.634	0.208	0.692	0.530	0.791
w/o cross-turn synthesis	0.675	0.400	0.681	0.591	0.789

Table 17: Single-property extraction ablations on LoCoMo (`gpt-4o-mini`). All rows are measured within one reproduction run; the unmodified reference (0.772) matches Table 1's EMem (0.780) within 0.8 points.

P Single-Session-Preference Failure Analysis

The largest category gap in Table 2 is single-session-preference. We audited this category end to end on reproduced runs: every question failed by at least one variant (judged wrong in at least two of the three judge runs; 26 questions) was traced through extraction, dense retrieval, LLM filtering, final-context construction, and answer generation, and annotated for both variants, giving 52 question–variant judgments, each labeled by three independent annotators over full trace sheets (majority vote; no ties; 88% unanimous). Table 19 locates the failures (25 per variant; 24 questions fail under both).

In 98% of failures, the preference evidence was extracted, retrieved, and present in the answer model's context, so the bottleneck is answer generation rather than memory. Our QA prompt requests concise factual answers: failed predictions average six words, whereas the category's reference rubrics average 58 words and often describe a desired personalized recommendation. As an oracle control, we bypassed memory and supplied the gold-evidence sessions directly to the same QA model: even with perfect retrieval by construction, accuracy is only 20.0%, and full-context prompting reaches 6.7% (Table 2). As a causal probe, we re-answered the same failed cases over unchanged retrieved contexts while modifying only the answer-prompt guidance: adding personalized-recommendation guidance (dropping the concision requirement) recovers 64% of the failures, and additionally instructing the model to name the specific items the user expressed preferences about recovers 80% (official judge, three runs), again without changing memory contents. Annotators also flagged 5 of the 26 failed questions (19%) as requiring details only weakly supported by the gold session; excluding them does not change the stage-level conclusion.

This audit was performed on reproduced runs. Over the full 470-question set, the reproduced pipelines reproduce the Table 2 averages within 1.3 points (77.2% for EMem-G and 77.3% for EMem against the printed 77.9% and 76.0%), and the other five categories reproduce within 6 points in either direction. The exception is this 30-question category, where the reproduced pipelines score 17.8% for EMem-G and 16.7% for EMem under the official three-run judge against the printed 32.2%. The category combines a small sample with rubric-style grading, which makes it uniquely sensitive to variation in `gpt-4o-mini` answer generation between the original runs and this later analysis; the oracle ceiling above (20.0%) brackets the reproduced numbers. The census conclusion concerns *where* failures occur and is unaffected by the rate. These findings align with Section 4: event-centric memory should be complemented by profile-style answer generation for preference-style questions.

Q Controlled Graph-Memory Baselines on LoCoMo

We evaluated the closest available graph-memory systems under a unified local protocol: one memory store per conversation; `gpt-4o-mini` and `text-embedding-3-small` configured through each system's released interface; the released indexing, retrieval, filtering, and QA pipelines otherwise unchanged; the same three-run judge; and the canonical 1,520 questions. HippoRAG 2 (Gutiérrez et al.,

Variant	ss-pref	ss-assist	temp.	m-sess	k-upd	ss-user	Avg.
Full: user EDUs + asst. EDUs + chunks	16.7	79.8	75.6	79.3	88.9	90.1	77.3
EDU-only: chunks hidden	13.3	42.9	74.8	81.0	88.9	92.2	73.2
User EDUs + chunks: asst. EDUs hidden	10.0	82.1	73.8	77.7	86.1	85.9	75.2

Table 18: Chunk-node ablation on LongMemEval$_S$ (`gpt-4o-mini`). All rows are measured within one reproduction run; the full-system reference (77.3) matches Table 2's EMem (76.0) within 1.3 points.

Failure location	EMem-G	EMem
Preference information not extracted	0%	0%
Extracted but absent from the dense top 30	0%	0%
Retrieved but absent from the final QA context	0%	4%
Present in context; answer misses the rubric	68%	60%
Present in context and reflected in the answer; judged strictly	32%	36%

Table 19: Failure locations for single-session-preference questions on the reproduced runs (share of each variant's 25 failures).

2025) is designed for passage-based document QA rather than conversational memory, so each dated, speaker-labeled session is represented as one passage (split at 1,200 tokens when needed) without changing its internal pipeline. A-Mem (Xu et al., 2025) runs via its released LoCoMo evaluation. Mem0's graph variant runs as `mem0ai` 0.1.118 with its embedded graph store. Results are in Table 20.

HippoRAG 2 is a strong comparator: its 0.741 is at Nemori's level (0.744), and it exceeds EMem-G on open-domain and single-hop questions. EMem and EMem-G are 3.9 points higher overall while EMem uses 14.9 times fewer context tokens; the differences concentrate in temporal reasoning (+14.9 points for EMem) and multi-hop reasoning (+8.3 points for EMem-G), while EMem ties HippoRAG 2 on single-hop questions. This pattern is consistent with our design claim: passage-level graph retrieval transfers well to factual lookup, whereas conversational temporal and multi-hop questions benefit from atomic, self-complete, temporally grounded EDUs and recall-oriented filtering. A-Mem reaches 0.508 under this protocol; published re-runs of A-Mem vary widely, and ours is on the favorable side. Mem0's graph variant reaches 0.545; for validation, Mem0's vector variant re-run under the same protocol reaches 0.594, within 1.9 points of Table 1's 0.613.

LongMemEval$_S$. A controlled LongMemEval$_S$ run of the closest graph-memory systems is left to future work because of its cost: the released Zep and Mem0 evaluation harnesses call hosted services rather than the open-source cores, and faithful ingestion through the open-source cores of Zep's engine, Mem0, and Mem0's graph variant is estimated at approximately 1.55M LLM calls over roughly 3.6B prompt tokens (about $680 at the prices used in our experiments), plus multi-day wall-clock at high rate-limit tiers; for scale, our entire two-variant evaluation used approximately 447K LLM calls. As contextual evidence only, re-scoring our cached LongMemEval$_S$ predictions with the judge setup of Rasmussen et al. (2025) (a GPT-4o judge with the official LongMemEval prompts; we pin the snapshot `gpt-4o-2024-08-06` and average three runs) yields 80.9% for EMem and 80.7% for EMem-G, against Zep's reported 63.8% with a `gpt-4o-mini` backbone and 71.2% with `gpt-4o`; these figures share the judge choice but not the full system protocol, so we do not present them as a main-table comparison.

R Scaling to Merged Conversation Histories

We stress-tested retrieval quality beyond single conversations by merging LoCoMo conversations into one shared EMem-G memory and re-asking each member conversation's questions, so every added conversation acts as distractor history. We used the eight conversations with globally unique speaker names (the other two share a speaker named John). Extraction is reused from cache; retrieval, filtering, and QA run fresh; the curve uses an in-loop single-run judge validated against the three-run protocol on the unmerged store. Results with `gpt-4o-mini` are in Table 21.

From one to eight merged conversations, accuracy stays within 1.5 points despite up to seven distractor histories per question, with no monotonic

System	Overall	F1	BLEU-1	Multi-Hop	Temporal	Open-Domain	Single-Hop	Tokens/query
HippoRAG 2	0.741	0.497	0.356	0.664	0.622	0.588	0.830	3,327
A-Mem	0.508	0.329	0.273	0.446	0.464	0.290	0.570	6,302
Mem0-graph	0.545	0.398	0.316	0.445	0.482	0.355	0.623	1,731
EMem (Table 1)	0.780	0.483	0.393	0.702	0.771	0.602	0.830	223
EMem-G (Table 1)	0.780	0.487	0.397	0.747	0.760	0.573	0.823	474

Table 20: Controlled runs of released graph-memory systems on LoCoMo (`gpt-4o-mini`, `text-embedding-3-small`, same three-run judge, canonical 1,520 questions; judge run-to-run std at most 0.3 points). The EMem and EMem-G rows are quoted from Tables 1 and 3, and a matched reproduction of EMem under this judge scores 0.772; see Appendix L. Context tokens per query are measured from each system's answer calls.

Merged scale	Sessions	Nodes	Questions	LLM Score	PPR ms/query
1 conversation	19	1,251	151	0.762	0.2
2 conversations	38	2,349	232	0.754	0.4
4 conversations	99	5,943	581	0.750	1.2
8 conversations	212	12,490	1,205	0.765	2.6

Table 21: Merged-history stress test (`gpt-4o-mini`). PPR latency measured on the MacBook Pro described in Appendix L.

degradation; the 8× graph (12,490 nodes) is 2.4 times the largest graph in Table 7. A matched control on the first conversation's 151 questions scores 0.762, 0.755, 0.788, and 0.755 from the unmerged store through the 2×, 4×, and 8× merges. We separately tested computational scaling by tiling replicas of the real 8× topology up to 199,840 nodes (approximately 3,400 sessions): PPR latency grows near-linearly, from 2.1 ms at 12.5K nodes (an isolated solver benchmark; the in-pipeline figure in Table 21 is 2.6 ms) to 19.6 ms when seeds lie in one cluster and 31.8 ms when seeds span clusters at 200K nodes (20 repetitions per point, std below 1 ms). The tiling experiment tests PPR cost only, not retrieval quality over thousands of sessions. Two caveats: merged independent conversations are a distractor-heavy stress test yet more separable than one coherent history, and absolute latency is machine-dependent.

S Per-Stage Efficiency Breakdown

We instrumented every query-time stage over a full LoCoMo pass for each variant with response caching disabled, so LLM-stage latencies reflect real API latency. Per-query times are batched wall-clock averages under the run's concurrency, the same amortized convention as Table 3. We define *search latency* as query encoding through final EDU selection and *end-to-end latency* as search plus answer generation. Because API conditions vary across runs, within-run shares are the comparable quantity; absolute times are machine-specific (Appendix L). Results are in Table 22.

Stage	EMem-G	EMem
Query encoding (embedding call)	9.2 (4.0%)	63.7 (39.3%)
Mention detection (LLM call)	43.1 (18.8%)	–
EDU filtering (LLM call)	81.9 (35.7%)	91.1 (56.2%)
Argument filtering (LLM call)	86.3 (37.6%)	–
Personalized PageRank	0.3 (0.1%)	–
Local bookkeeping	8.8 (3.8%)	7.3 (4.5%)
Search total	229.7	162.1
Answer generation (LLM call)	56.7	57.7
End-to-end per query	**286.4**	**219.8**

Table 22: Per-stage query-time latency on LoCoMo (`gpt-4o-mini`); ms per query and share of search latency. Mention detection, argument filtering, and PPR exist only in EMem-G.

LLM calls dominate: the two filters account for 73% of EMem-G's search latency, and EDU filtering accounts for 56% of EMem's; PPR is negligible at this scale (Appendix K). Per query, EMem-G uses 283 tokens for mention detection, 3,571 across the two filters, and 989 for answer generation, approximately $0.97 per 1,000 queries at the pricing used in our experiments. Table 3's search figures (234 and 151 ms) come from a different run on different hardware; the share pattern reproduces.

T Prompts

This appendix reproduces, verbatim, every prompt used by EMem and EMem-G and by the evaluation judges; the same prompts are included in our released repository. Each pipeline prompt is shown as its full message list inside one box: the system

message, the fixed one-shot exemplar (identical for every input; the exemplar's assistant message shows the structured JSON output), and the final user-message template with its placeholders (e.g., `{query}`). The colored role markers (e.g., [System message]) are typeset labels separating the messages, not prompt text. JSON outputs are displayed with indentation for readability; at runtime they are serialized compactly. Every pipeline call additionally enforces the corresponding JSON response schema through the API's structured-output support (Appendix L).

T.1 EDU Extraction (LoCoMo)

Applied once per session; response format `ConversationEDUExtractionV1`.

Prompt 1: EDU Extraction (LoCoMo)

[System message]

```
Given a conversation session between
speakers with numbered turns, your task is
to decompose it into Elementary Discourse
Units (EDUs) - short spans of text that
are minimal yet complete in meaning. Each
EDU should express a single fact, event,
or proposition and be atomic (not easily
divisible further while still making
sense). It is important that you preserve
all information from the conversation - no
detail should be lost in the extraction
process.
Requirements for Conversation EDUs with
Turn Attribution:
1. Each EDU should be a self-contained
unit of meaning that can be understood
independently. It should not depend on any
other EDU for understanding, although it
may relate to it
2. Avoid pronouns or ambiguous references
- use specific names and details, and
consistently use the most informative name
for each entity in all EDUs
3. The extracted EDUs must include all the
information conveyed in the current
conversation session. The extracted EDUs
should collectively capture everything
discussed
4. For each EDU, you must provide the
source_turn_ids field containing a list of
turn ID integers from which the EDU was
extracted or referenced (e.g., [1], [3,
5], etc.)
5. EDUs can span multiple turns if they
represent the same factual unit - in such
cases, include all relevant turn IDs
6. Focus on extracting facts, events, and
substantive information rather than
conversational pleasantries
7. Infer and add complete temporal context
where needed for clarity
8. Pay attention to capturing all details,
facts, decisions, concerns, and
substantive information from all speakers
```

[User message]

```
Session conversation:
Date: 2:30 pm on 15 March, 2024

Turn 1:
Alice: Hey Bob! How was your trip to Tokyo?

Turn 2:
Bob: It was amazing! I spent 5 days there
for the Global AI Innovation Symposium
2024. The conference at Tokyo University
was incredible.

Turn 3:
Alice: That sounds exciting! What was the
main focus?

Turn 4:
Bob: They had sessions on large language
models and robotics. I presented our
recent work on multimodal learning.

Turn 5:
Alice: How did it go?

Turn 6:
Bob: Really well! We got great feedback
and Dr. Yamamoto from Sony AI wants to
collaborate on our next project.

Turn 7:
Alice: That's fantastic! When are you
planning to start that collaboration?
Also, I remember our department head
mentioned we should prioritize industry
partnerships this quarter.

Turn 8:
Bob: We're aiming for next month. He said
they have a $2 million budget for joint
research.

Turn 9:
Alice: Wow, that's significant! That would
definitely meet our Q2 funding targets.
What about your flight back?

Turn 10:
Bob: Flight was delayed by 3 hours due to
weather, but I made it back safely on
March 14th evening.

Turn 11:
Alice: At least you're back safely. I've
scheduled a team meeting for Monday to
discuss how we can support this
collaboration.

Speaker names: Alice, Bob
```

[Assistant message]

```
{
  "edus": [
```

```
    {
      "edu_text": "Bob traveled to Tokyo
for 5 days to attend the Global AI
Innovation Symposium 2024 in March 2024",
      "source_turn_ids": [
        2
      ]
    },
    {
      "edu_text": "The Global AI
Innovation Symposium 2024 was held at
Tokyo University in Tokyo",
      "source_turn_ids": [
        2
      ]
    },
    {
      "edu_text": "The Global AI
Innovation Symposium 2024 included
sessions on large language models and
robotics",
      "source_turn_ids": [
        4
      ]
    },
    {
      "edu_text": "Bob presented his
team's recent work on multimodal learning
at the Global AI Innovation Symposium
2024",
      "source_turn_ids": [
        4
      ]
    },
    {
      "edu_text": "Bob's presentation on
multimodal learning at the Global AI
Innovation Symposium 2024 received great
feedback",
      "source_turn_ids": [
        6
      ]
    },
    {
      "edu_text": "Dr. Yamamoto from Sony
AI expressed interest in collaborating
with Bob on a joint research project after
his presentation at the Global AI
Innovation Symposium 2024",
      "source_turn_ids": [
        4,
        6
      ]
    },
    {
      "edu_text": "Alice and Bob's
department head mentioned prioritizing
industry partnerships in Q2 2024",
      "source_turn_ids": [
        7
      ]
    },
    {
      "edu_text": "Bob and Dr. Yamamoto
from Sony AI plan to start the Bob-Sony AI
collaboration project in April 2024",
      "source_turn_ids": [
        8
      ]
    },
    {
      "edu_text": "Dr. Yamamoto from Sony
AI has a $2 million budget for the
Bob-Sony AI collaboration project",
      "source_turn_ids": [
        8
      ]
    },
    {
      "edu_text": "The Bob-Sony AI
collaboration project would meet the Q2
2024 funding targets of Bob and Alice's
department",
      "source_turn_ids": [
        9
      ]
    },
    {
      "edu_text": "Bob's return flight
from Tokyo in March 2024 was delayed by 3
hours due to weather",
      "source_turn_ids": [
        10
      ]
    },
    {
      "edu_text": "Bob arrived safely back
from Tokyo on March 14th, 2024 evening
despite the flight delay",
      "source_turn_ids": [
        10
      ]
    },
    {
      "edu_text": "Alice scheduled a team
meeting for Monday March 18th, 2024 to
discuss supporting the Bob-Sony AI
collaboration project",
      "source_turn_ids": [
        11
      ]
    }
  ]
}
```

[User message]

```
Session conversation:
${session_text}

Speaker names: ${speaker_names}
```

T.2 EDU Extraction, User Side (LongMemEval$_S$)

Applied to user turns of each session (Appendix B).

Prompt 2: EDU Extraction, User Side

[System message]

```
Given a conversation session between
speakers with numbered turns, your task is
to extract Elementary Discourse Units
```

(EDUs) from the User's utterances only and attribute each EDU to the specific turn IDs it was extracted from. Each EDU should be a short span of text that is minimal yet complete in meaning, expressing a single fact, event, or proposition and be atomic (not easily divisible further while still making sense). It is critical that you preserve all information from the User's utterances - no detail should be lost in the extraction process.
Requirements for User-side Conversation EDUs with Turn Attribution:
1. Extract EDUs only from the User's utterances - do not extract information that is solely from the Assistant's responses without the User referencing or engaging with it
2. When the User references or engages with information from the Assistant's utterances (e.g., 'those programs you mentioned', 'the salary ranges you provided'), you must resolve these references by incorporating the specific details from the Assistant's utterances to make the EDU self-contained
3. Each EDU should be a self-contained unit of meaning that can be understood independently, without requiring context from other EDUs or the conversation
4. Avoid pronouns or ambiguous references - use specific names and details, and consistently use the most informative name for each entity in all EDUs
5. The extracted EDUs must include all the information and all the details expressed by the User, regardless of how minor the detail may seem. The extracted EDUs should be able to completely replace all of the User's utterances.
6. For each EDU, you must provide the source_turn_ids field containing a list of turn ID integers from which the EDU was extracted or referenced (e.g., [1], [3, 5], etc.)
7. EDUs can span multiple User turns if they represent the same factual unit or evolving state - in such cases, include all relevant turn IDs
8. Focus on extracting facts, events, preferences, decisions, concerns, and substantive information rather than conversational pleasantries
9. Infer and add complete temporal context where needed for clarity
10. Pay special attention to capturing all details, facts, preferences, background information, personal attributes, and states mentioned by the User - these are critical and must never be omitted

[User message]

Session conversation:
Date: 2023/05/29 (Mon) 04:50

Turn 1:
User: I'm considering pursuing a Master's degree in Data Science and I've narrowed down my options to Stanford and Carnegie Mellon. Can you give me an overview of the programs and their strengths?

Turn 2:
Assistant: You've shortlisted two of the most prestigious institutions for Data Science programs. Here's a brief overview:

Stanford University - Master of Science in Data Science (MS-DS)
* Duration: 12-18 months
* Strengths: Located in Silicon Valley with access to top tech companies, renowned faculty in AI/ML, strong emphasis on practical applications and industry collaboration, entrepreneurial resources
* Considerations: Highly competitive admissions (5-6% acceptance rate), one of the most expensive universities in the US

Carnegie Mellon University - Master of Science in Data Science (MS-DS)
* Duration: 16-24 months
* Strengths: Leader in AI/ML research, strong industry partnerships with Google and Uber, interdisciplinary approach integrating technical skills with policy/management, Pittsburgh's lower cost of living
* Considerations: Location may be less desirable for West Coast/Silicon Valley seekers, emphasis on policy aspects may not appeal to purely technical students

Both programs are excellent choices. Research thoroughly, reach out to current students and alumni, and weigh pros and cons based on your goals.

Turn 3:
User: I'm leaning towards Stanford, but I'm concerned about the cost. Can you tell me about the average salary of a software engineer in San Francisco? Also, I need to finalize which program to accept by this Friday since that's the admission offer deadline. By the way, I completed my undergraduate degree in Computer Science from UCLA in 2021, which I believe has a strong reputation in the tech industry.

Turn 4:
Assistant: Great that you have a CS background from UCLA - it's well-regarded! For SF software engineer salaries: the average is around $144,000-$164,000 per year, but ranges from $110,000 to over $200,000 depending on experience, company, and role. Entry-level (0-3 years): $110,000-$130,000; Mid-level (4-7 years): $140,000-$160,000; Senior (8-12 years): $160,000-$190,000; Lead/Manager (13+ years): $190,000-$220,000. These estimates vary by company and industry.

```
Turn 5:
User: Thanks! Those salary ranges you
mentioned would definitely help justify
the investment. I'm particularly
interested in the Silicon Valley location
and the industry connections you described
for Stanford.

Speaker names: User, Assistant
```

[Assistant message]

```
{
  "edus": [
    {
      "edu_text": "In a conversation on
May 29, 2023, the User expressed that the
User is considering pursuing a Master's
degree in Data Science",
      "source_turn_ids": [
        1
      ]
    },
    {
      "edu_text": "The User has narrowed
down Master's degree options to Stanford
University and Carnegie Mellon University,
but is leaning towards Stanford University
after initial consideration",
      "source_turn_ids": [
        1,
        3
      ]
    },
    {
      "edu_text": "The User is concerned
about the cost of attending Stanford
University's Master of Science in Data
Science (MS-DS) program",
      "source_turn_ids": [
        3
      ]
    },
    {
      "edu_text": "The User needs to
decide between accepting the Stanford
University Master of Science in Data
Science (MS-DS) program admission offer or
the Carnegie Mellon University Master of
Science in Data Science (MS-DS) program
admission offer by June 2, 2023 (Friday)
as the deadline for accepting the
admission offers",
      "source_turn_ids": [
        3
      ]
    },
    {
      "edu_text": "The User completed an
undergraduate degree in Computer Science
from UCLA (University of California, Los
Angeles) in 2021",
      "source_turn_ids": [
        3
      ]
    },
    {
      "edu_text": "The User believes UCLA
(University of California, Los Angeles)
has a strong reputation in the tech
industry",
      "source_turn_ids": [
        3
      ]
    },
    {
      "edu_text": "The User believes that
software engineer salary ranges in San
Francisco (entry-level $110,000-$130,000,
mid-level $140,000-$160,000, senior
$160,000-$190,000, lead/manager
$190,000-$220,000) would help justify the
investment in Stanford University's Master
of Science in Data Science (MS-DS)
program",
      "source_turn_ids": [
        5
      ]
    },
    {
      "edu_text": "The User is
particularly interested in Stanford
University's Silicon Valley location with
access to top tech companies",
      "source_turn_ids": [
        5
      ]
    },
    {
      "edu_text": "The User is
particularly interested in Stanford
University's industry connections and
collaboration opportunities",
      "source_turn_ids": [
        5
      ]
    }
  ]
}
```

[User message]

```
Session conversation:
${session_text}

Speaker names: ${speaker_names}
```

T.3 Structured Extraction, Assistant Side (LongMemEval$_S$)

Produces atomic EDUs and structured chunks for assistant turns (Appendix B).

Prompt 3: Structured Extraction

[System message]

```
Given a conversation session between
speakers with numbered turns, your task is
to extract information from the
Assistant's utterances in two forms: (1)
Simple EDUs and (2) Structured Chunks. You
must identify which information should be
```

extracted as simple atomic EDUs and which should be kept together as structured chunks.

Simple EDUs (Elementary Discourse Units):
These are atomic units of information - single facts, statements, opinions, or recommendations that stand alone. Each simple EDU should be minimal yet complete in meaning, expressing a single fact or proposition that cannot be easily divided further while still making sense.

Structured Chunks:
These are cohesive information blocks that contain multiple related details organized in a specific structure (e.g., comparisons, detailed overviews, comprehensive recommendations, step-by-step procedures, lists of related items). Breaking these into individual EDUs would destroy the underlying organizational structure and overall meaning.

Requirements for Simple EDUs:
1. Extract standalone facts, single opinions, brief recommendations, or acknowledgments that don't require the surrounding structure to be understood
2. Each EDU should be self-contained and understandable independently
3. Avoid pronouns or ambiguous references - use specific names and details, and consistently use the most informative name for each entity in all EDUs
4. Include source_turn_ids indicating which turn(s) the EDU came from

Requirements for Structured Chunks:
1. Identify information blocks that have an inherent organizational structure (comparisons, detailed breakdowns, multi-attribute descriptions)
2. Keep the entire structure intact - do not break it into separate EDUs
3. Resolve all coreferences and pronouns in chunk_content to make it fully self-contained (e.g., replace 'it' with the actual entity name)
4. For chunk_summary, provide a brief summary-style description that:
   - States what user request or question this chunk addresses
   - Describes what information categories are covered (e.g., 'program duration, strengths, considerations')
   - Naturally incorporates essential entities and key terms within the summary (e.g., institution names like 'Stanford University' and 'Carnegie Mellon University', category labels like 'Entry-level' or 'Senior', key concepts like 'AI/ML' or 'industry collaboration')
   Keep the summary concise (2-3 sentences) and flowing naturally while ensuring key entities and concepts appear for retrieval purposes. Avoid listing out specific numbers or exhaustive details - those belong in chunk_content
5. Include source_turn_ids indicating which turn(s) the chunk came from (can span multiple turns if they form one coherent structure)

General Guidelines:
1. Process ALL information from Assistant's utterances - nothing should be lost
2. Use your judgment to decide whether information should be a simple EDU or part of a structured chunk
3. If multiple pieces of information naturally belong together in a structure, group them as a structured chunk
4. If a fact or statement stands alone and doesn't need the surrounding structure, extract it as a simple EDU
5. When chunks span multiple turns or reference user context, include all relevant turn IDs
6. Focus on substantive information - skip conversational pleasantries unless they contain meaningful opinions or recommendations

[User message]

Session conversation:
Date: 2023/05/29 (Mon) 04:50

Turn 1:
User: I'm considering pursuing a Master's degree in Data Science and I've narrowed down my options to Stanford and Carnegie Mellon. Can you give me an overview of the programs and their strengths?

Turn 2:
Assistant: You've shortlisted two of the most prestigious institutions for Data Science programs. Here's a brief overview:

Stanford University - Master of Science in Data Science (MS-DS)
* Duration: 12-18 months
* Strengths: Located in Silicon Valley with access to top tech companies, renowned faculty in AI/ML, strong emphasis on practical applications and industry collaboration, entrepreneurial resources
* Considerations: Highly competitive admissions (5-6% acceptance rate), one of the most expensive universities in the US

Carnegie Mellon University - Master of Science in Data Science (MS-DS)
* Duration: 16-24 months
* Strengths: Leader in AI/ML research, strong industry partnerships with Google and Uber, interdisciplinary approach

integrating technical skills with policy/management, Pittsburgh's lower cost of living
* Considerations: Location may be less desirable for West Coast/Silicon Valley seekers, emphasis on policy aspects may not appeal to purely technical students

Both programs are excellent choices. Research thoroughly, reach out to current students and alumni, and weigh pros and cons based on your goals.

Turn 3:
User: I'm leaning towards Stanford, but I'm concerned about the cost. Can you tell me about the average salary of a software engineer in San Francisco? Also, I need to finalize which program to accept by this Friday since that's the admission offer deadline. By the way, I completed my undergraduate degree in Computer Science from UCLA in 2021, which I believe has a strong reputation in the tech industry.

Turn 4:
Assistant: Great that you have a CS background from UCLA - it's well-regarded! For SF software engineer salaries: the average is around $144,000-$164,000 per year, but ranges from $110,000 to over $200,000 depending on experience, company, and role. Entry-level (0-3 years): $110,000-$130,000; Mid-level (4-7 years): $140,000-$160,000; Senior (8-12 years): $160,000-$190,000; Lead/Manager (13+ years): $190,000-$220,000. These estimates vary by company and industry.

Turn 5:
User: Thanks! Those salary ranges you mentioned would definitely help justify the investment. I'm particularly interested in the Silicon Valley location and the industry connections you described for Stanford.

Speaker names: User, Assistant

[Assistant message]


```
{
  "simple_edus": [
    {
      "edu_text": "The Assistant's opinion is that Stanford University and Carnegie Mellon University are two of the most prestigious institutions for Data Science programs and both Stanford University's Master of Science in Data Science (MS-DS) program and Carnegie Mellon University's Master of Science in Data Science (MS-DS) program are excellent choices",
      "source_turn_ids": [
        2
      ]
    },
    {
      "edu_text": "The Assistant recommends that prospective students should research the Data Science programs thoroughly, reach out to current students and alumni, and weigh the pros and cons based on individual goals",
      "source_turn_ids": [
        2
      ]
    },
    {
      "edu_text": "The Assistant's opinion is that UCLA (University of California, Los Angeles) is well-regarded and a Computer Science background from UCLA serves as a solid foundation for graduate studies in Data Science",
      "source_turn_ids": [
        4
      ]
    },
    {
      "edu_text": "The Assistant notes that software engineer salary estimates for San Francisco can vary depending on the company, industry, and specific role",
      "source_turn_ids": [
        4
      ]
    }
  ],
  "structured_chunks": [
    {
      "chunk_content": "**Comparative Overview of Stanford University and Carnegie Mellon University Master of Science in Data Science (MS-DS) Programs:**\n\n**Stanford University - Master of Science in Data Science (MS-DS)**\n* Duration: 12-18 months\n* Strengths: Located in Silicon Valley with access to top tech companies, renowned faculty in AI/ML, strong emphasis on practical applications and industry collaboration, entrepreneurial resources\n* Considerations: Highly competitive admissions (5-6% acceptance rate), one of the most expensive universities in the US\n\n**Carnegie Mellon University - Master of Science in Data Science (MS-DS)**\n* Duration: 16-24 months\n* Strengths: Leader in AI/ML research, strong industry partnerships with Google and Uber, interdisciplinary approach integrating technical skills with policy/management, Pittsburgh's lower cost of living\n* Considerations: Location may be less desirable for West Coast/Silicon Valley seekers, emphasis on policy aspects may not appeal to purely technical students",
      "chunk_summary": "Comparative overview of Stanford University and Carnegie Mellon University MS-DS programs addressing the User's request for program information and strengths. Covers program duration, key strengths including Silicon
```

```
Valley location, AI/ML faculty and
research, industry collaboration with
companies like Google and Uber,
interdisciplinary approach, and important
considerations such as acceptance rates,
costs, and location preferences
(Pittsburgh vs Silicon Valley).",
      "source_turn_ids": [
        1,
        2
      ]
    },
    {
      "chunk_content": "**Software
Engineer Salary Ranges in San
Francisco:**\n\nThe average salary for a
software engineer in San Francisco is
around $144,000 to $164,000 per year, but
the range extends from $110,000 to over
$200,000 depending on experience, company,
and role.\n\nSalary breakdown by
experience level:\n* Entry-level (0-3
years of experience): $110,000 to $130,000
per year\n* Mid-level (4-7 years of
experience): $140,000 to $160,000 per
year\n* Senior (8-12 years of experience):
$160,000 to $190,000 per year\n*
Lead/Manager (13+ years of experience):
$190,000 to $220,000 per year",
      "chunk_summary": "Software engineer
salary information in San Francisco
addressing the User's question about
average salaries. Includes overall salary
ranges and detailed breakdown by
experience levels: Entry-level, Mid-level,
Senior, and Lead/Manager positions with
corresponding years of experience and
salary ranges.",
      "source_turn_ids": [
        3,
        4
      ]
    }
  ]
}
```

[User message]

```
Session conversation:
${session_text}

Speaker names: ${speaker_names}
```

T.4 Event Argument Extraction

Applied once per EDU; returns the event type and role–argument pairs.

Prompt 4: Event Argument Extraction

[System message]

```
Given an Elementary Discourse Unit (EDU)
which describes an event, your task is to:
1. Summarize the event type
2. Extract all of the event argument roles
and corresponding arguments

Requirements:
1. Avoid pronouns or ambiguous references
in the event type, argument roles and
arguments. Make sure they are informative.
2. Make sure the event type concisely and
accurately summarizes the EDU while being
general.
3. The extracted roles and arguments must
include all the information conveyed in
the input EDU.
```

[User message]

```
EDU: Tirzepatide, under the brand name
Mounjaro, received its Food and Drug
Administration (FDA) approval for Type 2
diabetes in May 2022.
```

[Assistant message]

```
{
  "event_type": "Drug Approval for Medical
Condition",
  "role_argument_pairs": [
    {
      "role": "drug generic name",
      "argument": "Tirzepatide"
    },
    {
      "role": "drug brand name",
      "argument": "Mounjaro"
    },
    {
      "role": "approval type",
      "argument": "Food and Drug
Administration (FDA) approval"
    },
    {
      "role": "approval authority",
      "argument": "Food and Drug
Administration (FDA)"
    },
    {
      "role": "approved medical condition",
      "argument": "Type 2 diabetes"
    },
    {
      "role": "approval date",
      "argument": "May 2022"
    }
  ]
}
```

[User message]

```
EDU: ${edu}
```

T.5 Query Mention Detection

Extracts entities, noun phrases, and salient concepts from the query (EMem-G).

Prompt 5: Query Mention Detection

[System message]

```
You're a very effective entity extraction
system for questions. Extract all entities
```

```
and key concepts that are important for
solving the question. Entities include
both named entities (e.g., people,
organizations, locations, events,
products) and non-named but salient
objects or concepts (e.g., "train",
"taxi", "hotel", "airport"). The entities
should be self-complete (no abbreviations,
use full names) to be easy to use for
exact matching or matching with encoded
embeddings. Focus on high precision — only
extract clear, unambiguous entities or
concepts that are central to the question.
```

[User message]

```
Should I visit the Louvre Museum or the
Musée d'Orsay in Paris if I'm interested
in Impressionist paintings and want to
take the metro instead of a taxi?
```

[Assistant message]

```
{
  "entities_and_concepts": [
    "Louvre Museum",
    "Musée d'Orsay",
    "Paris",
    "Impressionist paintings",
    "metro",
    "taxi"
  ]
}
```

[User message]

```
Question: ${query}
```

T.6 Query Named-Entity Detection (ablation variant)

Named-entity-only mention detector used in the "w/ Query NED to node" ablation rows of Tables 4 and 5.

Prompt 6: Query NED (ablation)

[System message]

```
You're a very effective entity extraction
system for questions. Extract all named
entities that are important for solving
the questions. The entities should be
self-complete (no abbreviations, use full
names) to be easy to use for exact
matching or matching with encoded
embeddings. Focus on high precision - only
extract clear, unambiguous entities.
```

[User message]

```
Which magazine was started first Arthur's
Magazine or First for Women?
```

[Assistant message]

```
{
  "named_entities": [
    "Arthur's Magazine",
    "First for Women"
  ]
}
```

[User message]

```
Question: ${query}
```

T.7 EDU Filtering (LoCoMo)

Recall-oriented EDU filter.

Prompt 7: EDU Filtering (LoCoMo)

[System message]

```
You are a conversational memory retrieval
assistant helping to filter candidate
memory units (EDUs - Elementary Discourse
Units) for answering a user's query.

Your task is to select EDUs that are
relevant to answering the query. Be
MAXIMALLY INCLUSIVE - err on the side of
keeping too many rather than too few:

**CRITICAL RULES:**
1. **Keep EDUs with embedded
information**: Even if an EDU contains
extra context or discusses multiple
topics, KEEP IT if ANY part is relevant to
the query. Don't discard EDUs just because
they're long or contain additional
information.

2. **All temporal references**: Keep EVERY
EDU that mentions dates, times, durations,
or events within the query's timeframe,
even if buried in longer descriptions.

3. **All quantitative information**: Keep
EVERY EDU containing numbers, counts,
amounts, or measurements related to the
query domain (e.g., points, items, events,
days).

4. **Direct matches**: Keep ANY EDU that
directly mentions entities, actions, or
topics from the query (e.g., if query asks
about 'items to pick up', keep EDUs
mentioning picking up anything).

5. **Multi-hop reasoning**: Include EDUs
that form chains of information. If asked
'how many X', keep ALL EDUs mentioning
individual instances of X, even if they
don't say 'how many'.

6. **Historical context**: Keep EDUs about
past activities, purchases, visits, or
experiences that relate to the query
domain.

7. **Prefer false positives over false
negatives**: It's MUCH better to include
an irrelevant EDU than to miss a relevant
one. When uncertain, ALWAYS include it.
```

```
8. **Copy exactly**: Selected EDUs must be copied exactly as they appear in the candidate list.

Remember: Your job is NOT to find the perfect answer, but to keep ALL EDUs that MIGHT help answer the query. Be generous!
```

[User message]

```
Query: How many loyalty points does Sarah currently have at her favorite bookstore?

Candidate EDUs:
[
  "On June 10, 2023, Sarah purchased three books at Barnes & Noble and earned 45 loyalty points, bringing the total to 320 points.",
  "Sarah enjoys reading mystery novels and has been collecting books from her favorite authors.",
  "On May 15, 2023, Sarah signed up for the Barnes & Noble membership program and received 50 welcome bonus points.",
  "Sarah is considering purchasing a new bookshelf to organize her growing book collection.",
  "Barnes & Noble offers 1 point for every dollar spent on books and other items.",
  "Sarah's favorite bookstore is Barnes & Noble, where she shops regularly for books and gifts.",
  "Sarah redeemed 75 points for a discount on June 12, 2023, leaving 245 points in her account.",
  "On June 14, 2023, Sarah purchased a journal at Barnes & Noble for $25 and earned 25 points, bringing the total to 270 points.",
  "Sarah has been visiting Barnes & Noble since childhood and has many fond memories of browsing the aisles.",
  "Emma suggests that loyalty programs are great for saving money on frequent purchases."
]
```

[Assistant message]

```
{
  "selected_edus": [
    "On May 15, 2023, Sarah signed up for the Barnes & Noble membership program and received 50 welcome bonus points.",
    "On June 10, 2023, Sarah purchased three books at Barnes & Noble and earned 45 loyalty points, bringing the total to 320 points.",
    "Barnes & Noble offers 1 point for every dollar spent on books and other items.",
    "Sarah's favorite bookstore is Barnes & Noble, where she shops regularly for books and gifts.",
    "Sarah redeemed 75 points for a discount on June 12, 2023, leaving 245 points in her account.",
    "On June 14, 2023, Sarah purchased a journal at Barnes & Noble for $25 and earned 25 points, bringing the total to 270 points."
  ]
}
```

[User message]

```
Query: How many concerts or shows did Marcus attend this summer?

Candidate EDUs:
[
  "Marcus is planning to attend more live music events and is looking for upcoming concerts in the area.",
  "On June 15, 2023, Marcus mentioned going to a jazz concert at the Blue Note last night and really enjoying the saxophone player's performance.",
  "Marcus prefers outdoor venues for concerts because of the better atmosphere and fresh air.",
  "Lisa suggests checking Ticketmaster and local venue websites for upcoming concert schedules.",
  "On July 4, 2023, Marcus asked for restaurant recommendations near the State Theater before attending a Broadway show that evening.",
  "Marcus has been collecting vinyl records from his favorite artists and enjoys listening to them at home.",
  "Marcus is interested in learning to play the guitar and asked about beginner lessons on July 20, 2023, mentioning he was inspired after seeing a live acoustic performance at Central Park the previous weekend.",
  "Lisa recommends following local music venues on social media to stay updated about upcoming shows.",
  "On August 10, 2023, Marcus shared that he went to an outdoor rock festival last Saturday and the headliner was amazing, though it rained during the opening act.",
  "Marcus is considering buying noise-canceling headphones for listening to music while commuting."
]
```

[Assistant message]

```
{
  "selected_edus": [
    "On June 15, 2023, Marcus mentioned going to a jazz concert at the Blue Note last night and really enjoying the saxophone player's performance.",
    "On July 4, 2023, Marcus asked for restaurant recommendations near the State Theater before attending a Broadway show that evening.",
    "Marcus is interested in learning to play the guitar and asked about beginner lessons on July 20, 2023, mentioning he was inspired after seeing a live acoustic performance at Central Park the previous
```

```
weekend.",
    "On August 10, 2023, Marcus shared
that he went to an outdoor rock festival
last Saturday and the headliner was
amazing, though it rained during the
opening act.",
    "Marcus prefers outdoor venues for
concerts because of the better atmosphere
and fresh air.",
    "Marcus is planning to attend more
live music events and is looking for
upcoming concerts in the area."
  ]
}
```

[User message]

```
Query: ${query}

Candidate EDUs:
${candidate_edus}
```

T.8 EDU Filtering (LongMemEval$_S$)

Recall-oriented EDU filter.

Prompt 8: EDU Filtering (LongMemEval)

[System message]

```
You are a conversational memory retrieval
assistant helping to filter candidate
memory units (EDUs - Elementary Discourse
Units) for answering a user's query.

Your task is to select EDUs that are
relevant to answering the query. Be
MAXIMALLY INCLUSIVE - err on the side of
keeping too many rather than too few:

**CRITICAL RULES:**
1. **Keep EDUs with embedded
information**: Even if an EDU contains
extra context or discusses multiple
topics, KEEP IT if ANY part is relevant to
the query. Don't discard EDUs just because
they're long or contain additional
information.

2. **All temporal references**: Keep EVERY
EDU that mentions dates, times, durations,
or events within the query's timeframe,
even if buried in longer descriptions.

3. **All quantitative information**: Keep
EVERY EDU containing numbers, counts,
amounts, or measurements related to the
query domain (e.g., points, items, events,
days).

4. **Direct matches**: Keep ANY EDU that
directly mentions entities, actions, or
topics from the query (e.g., if query asks
about 'items to pick up', keep EDUs
mentioning picking up anything).

5. **Multi-hop reasoning**: Include EDUs
that form chains of information. If asked
'how many X', keep ALL EDUs mentioning
individual instances of X, even if they
don't say 'how many'.

6. **Historical context**: Keep EDUs about
past activities, purchases, visits, or
experiences that relate to the query
domain.

7. **Prefer false positives over false
negatives**: It's MUCH better to include
an irrelevant EDU than to miss a relevant
one. When uncertain, ALWAYS include it.

8. **Copy exactly**: Selected EDUs must be
copied exactly as they appear in the
candidate list.

Remember: Your job is NOT to find the
perfect answer, but to keep ALL EDUs that
MIGHT help answer the query. Be generous!
```

[User message]

```
Query: Date of user query: 2023/06/15
(Thu) 14:30
User: How many loyalty points do I
currently have at my favorite bookstore?

Candidate EDUs:
[
  "On June 10, 2023, the User purchased
three books at Barnes & Noble and earned
45 loyalty points, bringing the total to
320 points.",
  "The User enjoys reading mystery novels
and has been collecting books from their
favorite authors.",
  "On May 15, 2023, the User signed up for
the Barnes & Noble membership program and
received 50 welcome bonus points.",
  "The User is considering purchasing a
new bookshelf to organize their growing
book collection.",
  "Barnes & Noble offers 1 point for every
dollar spent on books and other items.",
  "The User's favorite bookstore is Barnes
& Noble, where they shop regularly for
books and gifts.",
  "The User redeemed 75 points for a
discount on June 12, 2023, leaving 245
points in their account.",
  "On June 14, 2023, the User purchased a
journal at Barnes & Noble for $25 and
earned 25 points, bringing the total to
270 points.",
  "The User has been visiting Barnes &
Noble since childhood and has many fond
memories of browsing the aisles.",
  "The Assistant suggests that loyalty
programs are great for saving money on
frequent purchases."
]
```

[Assistant message]

```
{
```

```
  "selected_edus": [
    "On May 15, 2023, the User signed up
for the Barnes & Noble membership program
and received 50 welcome bonus points.",
    "On June 10, 2023, the User purchased
three books at Barnes & Noble and earned
45 loyalty points, bringing the total to
320 points.",
    "Barnes & Noble offers 1 point for
every dollar spent on books and other
items.",
    "The User's favorite bookstore is
Barnes & Noble, where they shop regularly
for books and gifts.",
    "The User redeemed 75 points for a
discount on June 12, 2023, leaving 245
points in their account.",
    "On June 14, 2023, the User purchased
a journal at Barnes & Noble for $25 and
earned 25 points, bringing the total to
270 points."
  ]
}
```

[User message]

```
Query: Date of user query: 2023/08/20
(Sun) 16:45
User: How many concerts or shows did I
attend this summer?

Candidate EDUs:
[
  "The User is planning to attend more
live music events and is looking for
upcoming concerts in the area.",
  "On June 15, 2023, the User mentioned
going to a jazz concert at the Blue Note
last night and really enjoying the
saxophone player's performance.",
  "The User prefers outdoor venues for
concerts because of the better atmosphere
and fresh air.",
  "The Assistant suggests checking
Ticketmaster and local venue websites for
upcoming concert schedules.",
  "On July 4, 2023, the User asked for
restaurant recommendations near the State
Theater before attending a Broadway show
that evening.",
  "The User has been collecting vinyl
records from their favorite artists and
enjoys listening to them at home.",
  "The User is interested in learning to
play the guitar and asked about beginner
lessons on July 20, 2023, mentioning they
were inspired after seeing a live acoustic
performance at Central Park the previous
weekend.",
  "The Assistant recommends following
local music venues on social media to stay
updated about upcoming shows.",
  "On August 10, 2023, the User shared
that they went to an outdoor rock festival
last Saturday and the headliner was
amazing, though it rained during the
opening act.",
  "The User is considering buying
noise-canceling headphones for listening
to music while commuting."
]
```

[Assistant message]

```
{
  "selected_edus": [
    "On June 15, 2023, the User mentioned
going to a jazz concert at the Blue Note
last night and really enjoying the
saxophone player's performance.",
    "On July 4, 2023, the User asked for
restaurant recommendations near the State
Theater before attending a Broadway show
that evening.",
    "The User is interested in learning to
play the guitar and asked about beginner
lessons on July 20, 2023, mentioning they
were inspired after seeing a live acoustic
performance at Central Park the previous
weekend.",
    "On August 10, 2023, the User shared
that they went to an outdoor rock festival
last Saturday and the headliner was
amazing, though it rained during the
opening act.",
    "The User prefers outdoor venues for
concerts because of the better atmosphere
and fresh air.",
    "The User is planning to attend more
live music events and is looking for
upcoming concerts in the area."
  ]
}
```

[User message]

```
Query: ${query}

Candidate EDUs:
${candidate_edus}
```

T.9 Argument Filtering (LoCoMo)

Recall-oriented argument filter (EMem-G).

Prompt 9: Argument Filtering (LoCoMo)

[System message]

```
You are a conversational memory retrieval
assistant helping to filter candidate
argument nodes (entities and keywords) for
answering a user's query.

**Context**: In our system, conversations
are broken down into Elementary Discourse
Units (EDUs), and each EDU is analyzed to
extract key arguments (entities, keywords,
concepts). These arguments are connected
to EDUs in a knowledge graph. By
identifying relevant arguments, we can
locate relevant EDUs and conversation
sessions that help answer the query.
```

Your task: Intelligently select argument nodes that could lead to information needed to answer the query. Prioritize PRECISION - focus on arguments that are truly relevant to the query intent.

REASONING PROCESS:
1. **Understand the query intent**: What information is needed to answer this query?
2. **Identify key entities and concepts**: What are the core elements mentioned in the query?
3. **Think about information dependencies**: What related entities or attributes would be needed to answer the query?
4. **Select relevant arguments**: Choose arguments that could lead to the target information.

SELECTION RULES:
1. **Direct query entities**: Keep entities explicitly mentioned in the query (people, places, organizations, objects, events).

2. **Query-relevant attributes**: Keep attributes, properties, or modifiers that are directly relevant to what the query asks about. For example:
   - If asking about 'loyalty points', keep 'points', 'loyalty', 'membership', 'earned', 'redeemed'
   - If asking about 'cost comparison', keep cost-related terms for the compared items

3. **Quantitative and temporal terms**: Keep numbers, counts, dates, or time-related terms when the query involves 'how many', 'when', 'how much', or comparison.

4. **Domain-specific terms**: Keep domain-specific terminology that relates to the query topic (e.g., for concert queries: 'venue', 'performance', 'show'; for restaurant queries: 'cuisine', 'dining', 'menu').

5. **Causal/relational terms**: Keep arguments that represent relationships or interactions relevant to the query (e.g., 'purchase', 'attend', 'visit', 'received').

6. **Arguments with partial relevance**: Keep arguments that contain relevant information EVEN IF they also include irrelevant information. Since argument extraction may not be perfect, an argument might be a list or phrase containing multiple entities where only some are relevant. If ANY part of the argument relates to the query, keep it. Examples:
   - 'coffee shop and bookstore' → Keep if query asks about either location
   - 'bus fare, taxi cost, and parking fee' → Keep if query asks about any transportation cost
   - 'Italian restaurant on Main Street' → Keep even if only 'Italian restaurant' or 'Main Street' is relevant

WHAT TO EXCLUDE:
- Arguments that are entirely irrelevant to the query intent with no connection to the information needed
- Generic background terms that don't help locate the target information (e.g., for a loyalty points query, exclude 'childhood memories', 'browsing aisles')
- Context entities with no involvement in the query target (e.g., for 'Sarah's points', exclude other people who don't affect her point transactions)

IMPORTANT: Do NOT exclude an argument just because it contains some irrelevant information. The argument extraction process is imperfect, so arguments may be compound phrases or lists. Focus on whether the argument CONTAINS relevant information, not whether it's PURELY relevant.

BALANCE: Be precise but not overly restrictive. If an argument could plausibly lead to information needed for the answer, include it. When uncertain about an argument's relevance, consider: 'Would conversations containing this argument likely contain information to answer the query?'

Copy exactly: Selected arguments must be copied exactly as they appear in the candidate list.

[User message]

Query: How many loyalty points does Sarah currently have at her favorite bookstore?

Candidate Arguments:
[
  "Sarah",
  "loyalty points",
  "bookstore",
  "Barnes & Noble",
  "points",
  "membership",
  "books",
  "purchase",
  "discount",
  "welcome bonus",
  "favorite",
  "shopping",
  "mystery novels",
  "authors",
  "bookshelf",
  "organize",
  "collection",
  "dollar spent",
  "redeemed",

```
  "journal",
  "childhood",
  "browsing",
  "aisles",
  "Emma",
  "saving money",
  "books, journals, and bookmarks",
  "loyalty points and rewards program"
]
```

[Assistant message]

```
{
  "selected_arguments": [
    "Sarah",
    "loyalty points",
    "bookstore",
    "Barnes & Noble",
    "points",
    "membership",
    "purchase",
    "welcome bonus",
    "redeemed",
    "favorite",
    "books, journals, and bookmarks",
    "loyalty points and rewards program"
  ]
}
```

[User message]

```
Query: How many concerts or shows did
Marcus attend this summer?

Candidate Arguments:
[
  "Marcus",
  "concerts",
  "shows",
  "summer",
  "attend",
  "live music",
  "events",
  "jazz concert",
  "Blue Note",
  "saxophone player",
  "performance",
  "outdoor venues",
  "atmosphere",
  "Lisa",
  "Ticketmaster",
  "venue websites",
  "State Theater",
  "Broadway show",
  "restaurant",
  "vinyl records",
  "artists",
  "listening",
  "guitar",
  "lessons",
  "acoustic performance",
  "Central Park",
  "rock festival",
  "headliner",
  "opening act",
  "noise-canceling headphones",
  "commuting",
  "concerts, festivals, and theater shows",
  "tickets and venue information"
]
```

[Assistant message]

```
{
  "selected_arguments": [
    "Marcus",
    "concerts",
    "shows",
    "summer",
    "attend",
    "live music",
    "events",
    "jazz concert",
    "Blue Note",
    "performance",
    "State Theater",
    "Broadway show",
    "acoustic performance",
    "Central Park",
    "rock festival",
    "concerts, festivals, and theater
shows",
    "tickets and venue information"
  ]
}
```

[User message]

```
Query: ${query}

Candidate Arguments:
${candidate_arguments}
```

T.10 Argument Filtering (LongMemEval$_S$)

Recall-oriented argument filter (EMem-G).

Prompt 10: Argument Filtering (LongMemEval)

[System message]

```
You are a conversational memory retrieval
assistant helping to filter candidate
argument nodes (entities and keywords) for
answering a user's query.

**Context**: In our system, conversations
are broken down into Elementary Discourse
Units (EDUs), and each EDU is analyzed to
extract key arguments (entities, keywords,
concepts). These arguments are connected
to EDUs in a knowledge graph. By
identifying relevant arguments, we can
locate relevant EDUs and conversation
sessions that help answer the query.

**Your task**: Select argument nodes that
are semantically related to the query and
could help locate relevant information. Be
MAXIMALLY INCLUSIVE - err on the side of
keeping too many rather than too few:

**CRITICAL RULES:**
1. **Semantic relevance**: Keep arguments
that are semantically related to the query
entities, topics, or intent, even if not
exact matches. Include synonyms, related
```

```
concepts, and contextually relevant terms.

2. **Domain entities**: Keep ALL entities
mentioned in the query domain (people,
places, organizations, products, events,
etc.). If the query asks about 'bookstore
loyalty points', keep arguments like
'Barnes & Noble', 'loyalty program',
'points', 'books', etc.

3. **Temporal and quantitative keywords**:
Keep arguments related to time expressions
(dates, months, durations) and quantities
(numbers, counts, measurements) if the
query involves temporal or quantitative
reasoning.

4. **Action and concept terms**: Keep
verbs, actions, and abstract concepts that
relate to the query intent. For example,
if asking 'how many concerts attended',
keep 'concert', 'show', 'performance',
'attended', 'music', 'venue', etc.

5. **Multi-hop reasoning support**:
Include arguments that might form chains
of reasoning. For example, if asking about
'favorite restaurant', keep not just
'restaurant' but also 'cuisine type',
'location', 'dining', 'food', etc.

6. **Entity attributes and modifiers**:
Keep arguments that are attributes or
modifiers of query entities (e.g., for
'outdoor concert', keep both 'outdoor' and
'concert').

7. **Prefer false positives over false
negatives**: It's MUCH better to include a
marginally relevant argument than to miss
an important one. When uncertain, ALWAYS
include it.

8. **Copy exactly**: Selected arguments
must be copied exactly as they appear in
the candidate list.

Remember: These arguments act as bridges
to find relevant conversation memory. Your
job is to identify ALL arguments that
might lead to useful information, not to
find the perfect answer. Be generous!
```

[User message]

```
Query: 2023-06-15: How many loyalty points
do I currently have at my favorite
bookstore?

Candidate Arguments:
[
  "I",
  "loyalty points",
  "bookstore",
  "Barnes & Noble",
  "points",
  "membership",
  "books",
  "purchase",
  "discount",
  "welcome bonus",
  "favorite",
  "shopping",
  "mystery novels",
  "authors",
  "bookshelf",
  "organize",
  "collection",
  "dollar spent",
  "redeemed",
  "journal",
  "childhood",
  "browsing",
  "aisles",
  "friend",
  "saving money"
]
```

[Assistant message]

```
{
  "selected_arguments": [
    "I",
    "loyalty points",
    "bookstore",
    "Barnes & Noble",
    "points",
    "membership",
    "books",
    "purchase",
    "discount",
    "welcome bonus",
    "favorite",
    "shopping",
    "redeemed",
    "journal"
  ]
}
```

[User message]

```
Query: 2023-08-15: How many concerts or
shows did I attend this summer?

Candidate Arguments:
[
  "I",
  "concerts",
  "shows",
  "summer",
  "attend",
  "live music",
  "events",
  "jazz concert",
  "Blue Note",
  "saxophone player",
  "performance",
  "outdoor venues",
  "atmosphere",
  "friend",
  "Ticketmaster",
  "venue websites",
  "State Theater",
  "Broadway show",
  "restaurant",
  "vinyl records",
  "artists",
```

```
  "listening",
  "guitar",
  "lessons",
  "acoustic performance",
  "Central Park",
  "rock festival",
  "headliner",
  "opening act",
  "noise-canceling headphones",
  "commuting"
]
```

[Assistant message]

```
{
  "selected_arguments": [
    "I",
    "concerts",
    "shows",
    "summer",
    "attend",
    "live music",
    "events",
    "jazz concert",
    "Blue Note",
    "performance",
    "outdoor venues",
    "State Theater",
    "Broadway show",
    "acoustic performance",
    "Central Park",
    "rock festival",
    "venue websites"
  ]
}
```

[User message]

```
Query: ${query}

Candidate Arguments:
${candidate_arguments}
```

T.11 Question Answering

Zero-shot chain-of-thought QA over the retrieved memory context.

Prompt 11: Question Answering

[System message]

```
You are an intelligent memory assistant tasked with
answering the given question by leveraging the
retrieved conversation memories. Think step-by-step
through your reasoning process before providing your
final answer.
```

[User message]

```
# CONTEXT:

You will be provided with memories extracted from a
conversation${date_span_info}${speaker_info}. These
memories contain timestamped information that may be
relevant to answering the question.

# INSTRUCTIONS:

Before answering, you must first reason through the
problem step-by-step in the "thought" field, then
provide your final answer.

**In your reasoning (thought field), explicitly:**
1. Identify which memories are relevant to the question
2. Extract key information from each relevant memory
(facts, dates, numbers, details)
3. If multiple memories are needed, explain how they
connect
4. If calculations are required (arithmetic or
temporal), show your work step-by-step
5. For temporal questions, convert any relative time
references (e.g., "last year", "two months ago") to
absolute dates using the memory's timestamp
6. Resolve any contradictions (prioritize more recent
memories)
7. Explain your reasoning before stating the answer

**Guidelines:**
- Carefully analyze all provided memories for relevant
information
- Pay attention to timestamps, speakers, and
contextual details
- For factual questions, look for direct evidence in
the provided memories
- For open-domain questions requiring reasoning or
external knowledge, integrate provided information
with your knowledge of related topics
- Verify arithmetic calculations carefully
- Convert relative time references to specific dates,
months, or years. For example, if a memory from 4 May
2022 mentions "went to India last year," then the trip
occurred in 2021
- The final answer should typically be concise (e.g.
less than 5-6 words), but you may provide slightly
longer answers when necessary
- Avoid vague time references in your final answer

Memories:

${context}

Question: ${question}
```

T.12 LLM Judge (LoCoMo)

The standard LoCoMo judge prompt, verbatim in the public Nemori and Mem0 evaluations (verified against both repositories). Single user-message template.

Prompt 12: LoCoMo Judge

```
Your task is to label an answer to a
question as 'CORRECT' or 'WRONG'. You will
be given the following data:
    (1) a question (posed by one user to
another user),
    (2) a 'gold' (ground truth) answer,
    (3) a generated answer
which you will score as CORRECT/WRONG.

The point of the question is to ask about
something one user should know about the
other user based on their prior
conversations.
The gold answer will usually be a concise
and short answer that includes the
referenced topic, for example:
Question: Do you remember what I got the
last time I went to Hawaii?
Gold answer: A shell necklace
The generated answer might be much longer,
but you should be generous with your
```

```
grading - as long as it touches on the
same topic as the gold answer, it should
be counted as CORRECT.

For time related questions, the gold
answer will be a specific date, month,
year, etc. The generated answer might be
much longer or use relative time
references (like "last Tuesday" or "next
month"), but you should be generous with
your grading - as long as it refers to the
same date or time period as the gold
answer, it should be counted as CORRECT.
Even if the format differs (e.g., "May
7th" vs "7 May"), consider it CORRECT if
it's the same date.

Now it's time for the real question:
Question: {question}
Gold answer: {gold_answer}
Generated answer: {generated_answer}

First, provide a short (one sentence)
explanation of your reasoning, then finish
with CORRECT or WRONG.
Do NOT include both CORRECT and WRONG in
your response, or it will break the
evaluation script.

Just return the label CORRECT or WRONG in
a json format with the key as "label".
```

T.13 LLM Judge (LongMemEval$_S$, temporal reasoning)

Official LongMemEval category-prompt text, as implemented in the Nemori evaluation framework (tagged question/response fields, structured yes/no output). Single user-message template.

Prompt 13: Temporal-Reasoning Judge

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

<QUESTION>
B: {question}
</QUESTION>
<CORRECT ANSWER>
{gold_answer}
</CORRECT ANSWER>
<RESPONSE>
A: {response}
</RESPONSE>
```

T.14 LLM Judge (LongMemEval$_S$, knowledge update)

Official LongMemEval category-prompt text, as implemented in the Nemori evaluation framework (tagged question/response fields, structured yes/no output). Single user-message template.

Prompt 14: Knowledge-Update Judge

```
I will give you a question, a correct
answer, and a response from a model.
Please answer yes if the response contains
the correct answer. Otherwise, answer no.
If the response contains some previous
information along with an updated answer,
the response should be considered as
correct as long as the updated answer is
the required answer.

<QUESTION>
B: {question}
</QUESTION>
<CORRECT ANSWER>
{gold_answer}
</CORRECT ANSWER>
<RESPONSE>
A: {response}
</RESPONSE>
```

T.15 LLM Judge (LongMemEval$_S$, single-session preference)

Official LongMemEval category-prompt text, as implemented in the Nemori evaluation framework (tagged question/response fields, structured yes/no output). Single user-message template.

Prompt 15: Preference Judge

```
I will give you a question, a rubric for
desired personalized response, and a
response from a model. Please answer yes
if the response satisfies the desired
response. Otherwise, answer no. The model
does not need to reflect all the points in
the rubric. The response is correct as
long as it recalls and utilizes the user's
personal information correctly.

<QUESTION>
B: {question}
```

```
</QUESTION>
<RUBRIC>
{gold_answer}
</RUBRIC>
<RESPONSE>
A: {response}
</RESPONSE>
```

T.16 LLM Judge (LongMemEval$_S$, default)

Official LongMemEval default-prompt text for the remaining categories, as implemented in the Nemori evaluation framework (tagged question/response fields, structured yes/no output). Single user-message template.

Prompt 16: Default Judge

```
I will give you a question, a correct
answer, and a response from a model.
Please answer yes if the response contains
the correct answer. Otherwise, answer no.
If the response is equivalent to the
correct answer or contains all the
intermediate steps to get the correct
answer, you should also answer yes. If the
response only contains a subset of the
information required by the answer, answer
no.

<QUESTION>
B: {question}
</QUESTION>
<CORRECT ANSWER>
{gold_answer}
</CORRECT ANSWER>
<RESPONSE>
A: {response}
</RESPONSE>
```