CAMEL: COLLABORATIVE AGENTIC MEMORY VIA EVIDENCE LINKING ACROSS SPEAKERS AND TIME

Anonymous authors
Paper under double-blind review

ABSTRACT

Agents in collaborative settings must answer questions from histories spanning multiple speakers, conversation threads, and extended periods. Terminology differences, decision revisions, and scattered evidence make it difficult to retrieve complementary information while preserving speaker attribution and temporal context. We introduce CAMEL, a collaborative memory framework combining an offline typed graph with a query-time evidence assembly pipeline. The graph preserves messages, speaker identities, timestamps, and reply, decision, and term relations without generative LLM calls at ingestion. At query time, CAMEL seeds candidates through query decomposition and hybrid retrieval, links evidence through graph expansion and bidirectional contextual term alignment, grounds messages in speaker and temporal context, and selects a compact window with chronological ordering where appropriate. Across EverMemBench, GroupMemBench, and SocialMemBench, CAMEL improves answer accuracy over the strongest of ten baselines by 12.5, 11.4, and 11.7 percentage points. Component ablations support the contributions of candidate generation, reranking, and contextual processing across benchmarks, while sensitivity analyses show that gains vary with answer-model capability and the available evidence budget. Code and data are at https://anonymous.4open.science/r/CAMEL-0822.

1 INTRODUCTION

Large language models (LLMs) have evolved beyond single-turn response generation to support agents capable of planning, using tools, and executing tasks over extended horizons (Yao et al., 2023; Schick et al., 2023; Park et al., 2023; Chen et al., 2026). Such effective operations require access to past interactions, yet processing complete histories is costly and does not ensure reliable use of information throughout long contexts (Liu et al., 2024). This motivates *agent memory systems* that retain past interactions and retrieve relevant evidence at inference time (Wu et al., 2026). In collaborative settings such as project teams and social groups, these agents must draw on information contributed by multiple participants. Memory in these settings must preserve clear associations between information and its source, allowing evidence to be interpreted with respect to both the speaker and the person asking the question. Maintaining this attribution within shared interaction histories is therefore a central requirement for *collaborative memory* construction.

Supporting queries over these shared histories entails three closely related requirements (Figure 1). ❶ **Speaker attribution and terminology:** Participants in different roles may describe the same issue using different expressions, such as an engineer's "latency regression" and a manager's "slowness issue". Evidence must therefore be interpreted across terminology differences while retaining its association with the appropriate participant. ❷ **Temporal validity:** Decisions are revised without removing earlier statements from the record. A budget established in March and updated in June requires the system to identify which statement applies to the query, while retaining earlier evidence when the question concerns the change itself. ❸ **Evidence completeness:** A compositional question may depend on several messages distributed across speakers, conversation threads, and months of interaction. These messages must be retrieved together, even when some have little lexical overlap with the query. Existing memory systems, however, have largely focused on dyadic conversational histories (Chhikara et al., 2025; Packer et al., 2023; Huang et al., 2026), leaving these requirements insufficiently addressed in combination. Ranking messages individually by relevance can favor redundant evidence, while compressing histories before the query is known may discard attribution

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

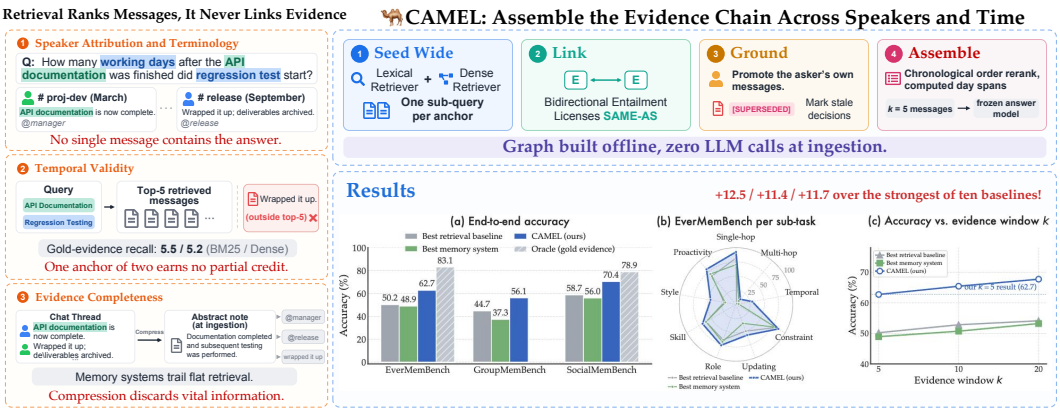


Figure 1: Evidence assembly for collaborative memory. **(a)** An illustrative question requires evidence from different speakers and dates. Retrieving one anchor is insufficient, while a compressed summary may omit the attribution or timing needed to relate the events. **(b)** CAMEL seeds, links, grounds, and assembles evidence. **(c)** Results across benchmarks, sub-tasks, and evidence budgets.

and contextual details needed for later questions. Collaborative memory therefore requires *assembling* complementary messages into a compact context that preserves the speaker attribution and temporal relationships needed to interpret them.

Building on this perspective, we propose CAMEL, a collaborative memory system organized around four functions (Figure 1, top right). An offline typed graph retains original messages with speaker attribution and timestamps and records reply, decision, and term relations, without generative LLM calls at ingestion. At query time, *seeding* decomposes compositional questions into sub-queries targeting distinct anchors and fuses lexical and dense retrieval to form a broad candidate pool. *Linking* expands candidates through graph relations and cross-role term aliases, admitting an alias only when contextual scores pass the threshold in both directions. *Grounding* supplies asker-specific evidence, participant profiles, and temporal annotations to contextualize retrieved statements. *Assembly* reranks and selects attributed messages within the evidence budget, preserves chronological order where needed, and supplements the context with deterministically computed date spans. With selected budget, CAMEL improves accuracy over the strongest of ten baselines by 12.5, 11.4, and 11.7 percentage points on EverMemBench, GroupMemBench, and SocialMemBench, respectively. Beyond answer accuracy, gold-evidence recall increases from 5.5% to 11.0% on EverMemBench and from 46.3% to 63.4% on SocialMemBench relative to the strongest retrieval baseline. These results show improvements in both answer correctness and annotated evidence retrieval under the evaluated settings. Our contributions are threefold:

- **Collaborative Memory Representation:** We introduce CAMEL, a collaborative memory framework built on a typed graph that preserves speaker attribution and timestamps and links equivalent terms across roles, without LLM calls at ingestion (Section 2.2).
- **Evidence Assembly:** We develop a query-time pipeline that combines hybrid retrieval, query decomposition, and graph expansion with speaker and temporal grounding to select and organize complementary evidence within a limited context budget (Sections 2.3–2.6).
- **Extensive Benchmarking:** We evaluate CAMEL across diverse collaborative memory benchmarks with different focus and demonstrate strong overall performance relative to a broad set of competitive retrieval and memory baselines (Sections 3).

2 METHOD

CAMEL combines an offline memory representation with query-time evidence assembly (Figure 2). We first formulate evidence selection under a limited context budget (Section 2.1), then describe the offline construction of attributed messages, typed relations, and speaker profiles (Section 2.2). Query-time processing comprises four components: *seeding* combines hybrid retrieval with query

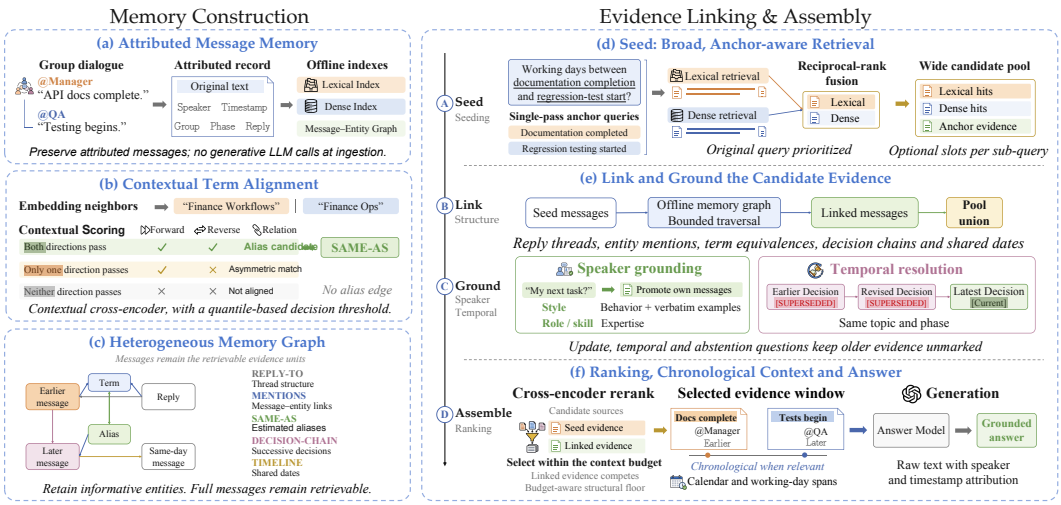


Figure 2: Overview of CAMEL. **Left:** offline construction preserves attributed messages and builds retrieval indexes (a), estimates term equivalence through bidirectional contextual scoring (b), and organizes message relations in a typed graph (c). **Right:** query-time processing retrieves seed candidates (d), expands related evidence and incorporates speaker and temporal context (e), and selects and organizes the evidence window (f). These operations implement four functions: seeding, linking, grounding, and assembly.

decomposition to obtain candidate messages (Section 2.3); *linking* expands these candidates through graph relations and aligned terms (Section 2.4); *grounding* incorporates speaker and temporal context into candidate generation and evidence interpretation (Section 2.5); and *assembly* reranks, selects, and organizes the evidence within the context budget (Section 2.6).

2.1 Problem Formulation

Let $\mathcal{M} = \{m_i\}_{i=1}^{n}$ denote a corpus of attributed messages. Each record $m_i = (t_i, a_i, \tau_i, \eta_i)$ contains text t_i, author a_i, timestamp τ_i, and available metadata η_i for group membership, conversation phase, and reply relations. Given a query q, optional asker identity u_q, and query-type signals z_q, the system returns an evidence sequence $E_q = (m_{i_1}, \ldots, m_{i_\ell})$, where $\ell \leq k$ and $i_r \neq i_s$ for $r \neq s$. A message limit k and a character limit L jointly constrain the retrieved-message context, with duplicate records excluded. The objective is to assemble complementary evidence that supports answering q while preserving speaker attribution and temporal relationships. Selection therefore considers both individual message relevance and relationships among the retrieved messages.

A lightweight local identifier infers question types from lexical cues in q, producing z_q to guide query decomposition, profile selection, and temporal processing. The frozen answer model receives the selected messages together with the query-dependent participant and temporal context described in Sections 2.5 and 2.6. Budget settings are reported in Appendix D.2.

2.2 Collaborative Memory Construction

Attributed messages and relations. We index the original messages for lexical and dense retrieval and construct a typed graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \mathcal{M} \cup \mathcal{X}$ and $\mathcal{X}$ contains terms extracted from capitalized phrases, quoted strings, and identifiers. The relation types are

$$\mathcal{E} = \mathcal{E}_{\text{Reply}} \cup \mathcal{E}_{\text{Mentions}} \cup \mathcal{E}_{\text{SameAs}} \cup \mathcal{E}_{\text{Decision}} \cup \mathcal{E}_{\text{Timeline}}. \quad (1)$$

Reply preserves parent–child links; Mentions associates messages with extracted terms; and SameAs records estimated term equivalence. Where decision annotations are supplied, Decision organizes statements by phase and topic in timestamp order. Timeline groups messages by date. These relations are materialized as adjacency lists and lookup indexes, alongside author and group indexes. The representation connects messages directly through conversational and decision relations, and indirectly through the terms associated with them. To control expansion through generic

terms, we remove singletons and the most frequent terms, then cap the retained vocabulary by idf-weighted frequency across the corpus of original attributed messages.

Contextual term alignment. Embedding similarity proposes candidate aliases, while a directional cross-encoder evaluates their contextual compatibility. For each term x, we retrieve its K nearest neighbors above a permissive cosine threshold and associate each term with a representative utterance c_x. Let $s(x \to y)$ denote the score obtained by placing (x, c_x) in the query position and (y, c_y) in the document position. We admit a SAMEAS edge only when both directions pass the threshold:

$$\textsc{Alias}(x, y) = \mathbf{1}[s(x \to y) \geq \tau_{\mathrm{a}} \ \wedge \ s(y \to x) \geq \tau_{\mathrm{a}}]\,. \tag{2}$$

Bidirectional relevance provides a heuristic for contextual term alignment. We set τ_{a} by a quantile of the observed directional scores. For a proposed pair, the acceptance criterion is equivalently $\min\{s(x \to y), s(y \to x)\} \geq \tau_{\mathrm{a}}$. Requiring both directions is intended to filter pairs with asymmetric coverage, such as a broad category and a specific operational term, without assuming that high embedding similarity alone establishes term equivalence in context.

Speaker profiles. Participant context is organized by identity, expertise, behavior, and activity for query-dependent grounding (Appendix D.2). Memory construction uses deterministic processing, embeddings, and cross-encoder inference, without generative LLM calls at ingestion.

2.3 SEEDING: HYBRID CANDIDATE RETRIEVAL

Seeding provides candidate evidence for both direct answers and subsequent expansion. BM25 ranks lexical matches, while cosine similarity over BGE-M3 embeddings retrieves semantic matches. Their rankings are combined through reciprocal-rank fusion (RRF) (Cormack et al., 2009):

$$\mathrm{RRF}(m \mid q) = \frac{1}{\kappa + r_{\mathrm{lex}}(m \mid q)} + \frac{1}{\kappa + r_{\mathrm{den}}(m \mid q)}, \tag{3}$$

where r_{lex} and r_{den} are the lexical and dense ranks, and κ is the rank offset. A message absent from a list contributes zero to that list's term. Combining these rank positions retains exact names and identifiers while accommodating differences in wording across participants.

For multi-hop and temporal query categories, we additionally decompose q into at most S retrieval sub-queries, each targeting a concrete event or fact. For example, a duration question requires separate evidence for its start and end events. Decomposition uses one LLM call when available, with a rule-based fallback, and does not condition retrieval on intermediate answers. Let $q_0 = q$ and $q_1, \ldots, q_s$, $s \leq S$, denote the resulting queries and $\mathcal{L}$ their hybrid ranking lists. Their fusion is

$$F(m \mid q) = \sum_{j=0}^{s} \frac{w_j}{\kappa + r_j(m)}, \qquad w_0 > w_{j>0}, \tag{4}$$

where $r_j(m)$ is the rank produced by Eq. 3 for q_j. The higher original-query weight preserves the full question as the primary retrieval signal. We retain the top N messages as the seed pool $C(q) = \mathrm{Top}_N(F(\cdot \mid q))$; decomposition broadens access to distinct anchors but does not guarantee their inclusion in the final window. All retrieval depths and weights are reported in Appendix D.2.

2.4 LINKING: GRAPH-BASED EVIDENCE EXPANSION

Linking retrieves messages related to the seed evidence, including messages with weak direct query overlap. From a bounded seed prefix $A(q) \subseteq C(q)$, we expand message-level neighborhoods for h rounds and independently retrieve messages through aliases of query-matched terms:

$$C_{\mathrm{link}}(q) = \mathrm{Expand}_h(G, A(q)) \cup \mathrm{AliasLookup}(G, q). \tag{5}$$

Each expansion round follows replies, neighboring decision statements, and messages sharing an informative term. Traversing from one message through a shared term to another counts as one round, although the path passes through an intermediate term node. Thus, h bounds message-level expansion rather than the number of individual edges traversed in the typed graph. Candidate priority depends on seed rank, relation type, term specificity, and hop decay, with bounded fan-out.

Explicit reply and decision relations receive greater weight than shared-term connections, and the strongest incoming relation determines priority with a small tie-breaking contribution from accumulated support. Alias lookup instead matches query terms against the offline alias index and retrieves messages mentioning their linked terms. It is separate from the bounded neighborhood traversal. Both operations retain their candidate lists for subsequent budgeted selection.

2.5 Grounding: Speaker and Temporal Context

Grounding incorporates speaker and temporal context in two places: it introduces speaker- and date-specific candidates before selection and contextualizes the evidence selected for the answer model.

Speaker context. For a first-person or `user_implicit` query with known asker u_q, auxiliary lexical retrieval is restricted to messages authored by that participant:

$$C_{\text{spk}}(q) = \text{BM25}_{b_s}(q, \{m_i \in \mathcal{M} : a_i = u_q\}), \tag{6}$$

where b_s is the speaker retrieval budget; otherwise, $C_{\text{spk}}(q) = \varnothing$. Selected messages by the asker or explicitly named participants receive ordering priority, accompanied by query-dependent participant context to help the answer model interpret the selected evidence (Appendix D.2).

Temporal context. Explicit query dates provide an additional source of candidates through the timeline index. Writing $\text{Dates}(q)$ for these dates, we obtain the date-based candidate set:

$$C_{\text{date}}(q) = \bigcup_{d \in \text{Dates}(q)} \text{Nearby}(G, d), \tag{7}$$

where $\text{Nearby}(G, d)$ returns messages near d. For a retrieved decision m, let $\mathcal{D}_q(m)$ contain selected decisions in the same phase and topic. The supersession indicator is

$$\text{stale}_q(m) = \mathbf{1}\left[\tau_m < \max_{m' \in \mathcal{D}_q(m)} \tau_{m'}\right]. \tag{8}$$

This annotates retrieved evidence without deleting messages. A capped restatement heuristic supplements explicit decision links (Appendix D.2). Annotation is disabled for knowledge-update, temporal, and abstention question types. Temporal grounding also resolves supported relative dates and computes date spans between candidate events for use in the answer context.

2.6 Assembly: Budgeted Evidence Selection

Assembly merges candidates by message identifier. Let Prefix_d retain the first d messages in a ranking. The combined pool submitted to the cross-encoder for relevance scoring is

$$C_{\text{score}}(q) = \text{Prefix}_{\max(D,k)}(C(q)) \cup C_{\text{link}}(q) \cup C_{\text{date}}(q) \cup C_{\text{spk}}(q). \tag{9}$$

Each candidate is scored against the original query using text with speaker and timestamp attribution. Expanded candidates remain eligible through scoring before the pool is truncated to N. We select up to k distinct messages in reranked order and fill remaining slots from the seed ranking.

An optional expansion floor targets

$$b_{\text{exp}} = \text{round}(\beta k) \tag{10}$$

messages supported by auxiliary retrieval or tracked sub-query provenance outside the top-k seed list. When this target is unmet, the procedure draws from the ordered expansion lists and replaces the lowest-ranked selected messages without such support, subject to candidate availability. The resulting heuristic incorporates candidate provenance into selection under the message budget.

Selected messages receive the speaker ordering and temporal annotations described above. Temporal, multi-hop, and knowledge-update queries use chronological order; other queries retain speaker-adjusted relevance order. We render complete attributed messages up to the character cap L and combine them with the participant and temporal context described in Section 2.5. This keeps the choice of evidence and its contextual interpretation distinct from the order in which the answer model receives the selected messages.

Method	EverMemBench				GroupMemBench		SocialMemBench			
	Acc ↑	Rec ↑	F1 ↑	ms ↓	Acc ↑	ms ↓	Acc ↑	Rec ↑	F1 ↑	ms ↓
Reference condition: gold evidence handed to the answer model										
Oracle	83.1	100.0	90.8	0	—	—	78.9	100.0	88.2	0
Retrieval baselines: no memory structure										
BM25	50.2	5.5	9.9	698	44.7	941	32.1	17.1	22.3	273
Dense (BGE-M3)	49.2	5.2	9.4	194	39.3	164	58.7	46.3	51.8	451
GraphRAG	36.8	0.2	0.4	1088	25.4	685	28.6	10.6	15.5	1242
Memory systems: structure built at ingestion										
Mem0	37.8	0.6	1.2	2052	32.6	460	27.4	3.2	5.7	1746
Zep	37.2	0.6	1.2	1330	19.6	2294	17.5	2.7	4.7	1743
HippoRAG	35.3	0.1	0.2	3105	30.9	9863	29.8	4.7	8.1	933
RAPTOR	38.7	0.7	1.4	2749	—	—	34.6	5.2	9.0	1050
ReadAgent	41.6	2.6	4.9	174	25.5	88	34.9	17.2	23.0	57
MemGPT	48.9	4.5	8.2	334	37.3	340	56.0	41.7	47.8	241
Flat Memory	39.2	0.9	1.8	49	27.2	45	25.8	3.2	5.7	45
CAMEL (ours)	**62.7**	**11.0**	**18.7**	2532	**56.1**	2867	**70.4**	**63.4**	**66.7**	298

Table 1: Main results. **Acc**: answer accuracy; **Rec**: gold-evidence recall; **F1**: their harmonic mean; **ms**: retrieval latency excluding answer generation. **Bold**/underlined values mark the best/second-best non-oracle results. GroupMemBench lacks gold evidence for Rec and F1.

3 EXPERIMENTS

We evaluate the comprehensive performance of CAMEL based on four research questions:

- **RQ1:** How do memory systems built for dyadic histories perform on collaborative corpora, and does CAMEL close the gap? *(**Main Results**, Section 3.2)*
- **RQ2:** How sensitive are the gains of CAMEL to the answer model? *(**Robustness**, Section 4)*
- **RQ3:** How does the advantage vary with evidence budget? *(**Parameter Sensitivity**, Section 5)*
- **RQ4:** How do component groups contribute to the gains of CAMEL? *(**Ablations**, Section 6)*

3.1 EXPERIMENTAL SETUP

Benchmarks. We evaluate CAMEL on three benchmarks that differ in collaboration setting and memory requirements. **EverMemBench** (Hu et al., 2026) simulates a year of enterprise collaboration (2,400 questions, nine sub-tasks). **GroupMemBench** (Yang et al., 2026) comprises four enterprise domains of 30k threaded messages each, with documented decision revisions and adversarial queries (745 questions, six types). **SocialMemBench** (Owolabi, 2026) covers 43 informal social networks with turn-level gold evidence (1,031 questions, nine types). Details in Appendix C.1.

Models and baselines. We adopt **GPT-5.6-Luna** as the primary backbone model for all methods, with **Qwen2.5-3B-Instruct** and **DeepSeek-V4.1-Flash** evaluated as alternative backbones in Section 4. Dense retrieval and reranking use `BGE-M3` and `BGE-reranker-v2-m3` (Chen et al., 2024), respectively. We compare against ten baselines in two categories (Table 1): *retrieval baselines*, comprising **BM25** (Robertson & Zaragoza, 2009), **dense retrieval**, and **GraphRAG** (Edge et al., 2024); and *memory systems*, comprising **Mem0** (Chhikara et al., 2025), **Zep** (Rasmussen et al., 2025), **HippoRAG** (Gutiérrez et al., 2024), **RAPTOR** (Sarthi et al., 2024), **ReadAgent** (Lee et al., 2024), **MemGPT** (Packer et al., 2023), and **flat memory**. An **oracle** uses gold evidence where annotations are available. RAPTOR is omitted on GroupMemBench because evaluation coverage is incomplete. Detailed configurations in Appendix D.1.

Metrics. We adopt answer **accuracy (Acc)** as the primary metric, with exact option matching for multiple-choice questions and LLM-based grading for open-ended responses. Judge reliability is assessed in Appendix C.3. We also report **retrieval latency (ms)** and **retrieval recall (Rec)** and the acc–rec harmonic mean (**F1**) where gold evidence is available. Formal definitions in Appendix C.2.

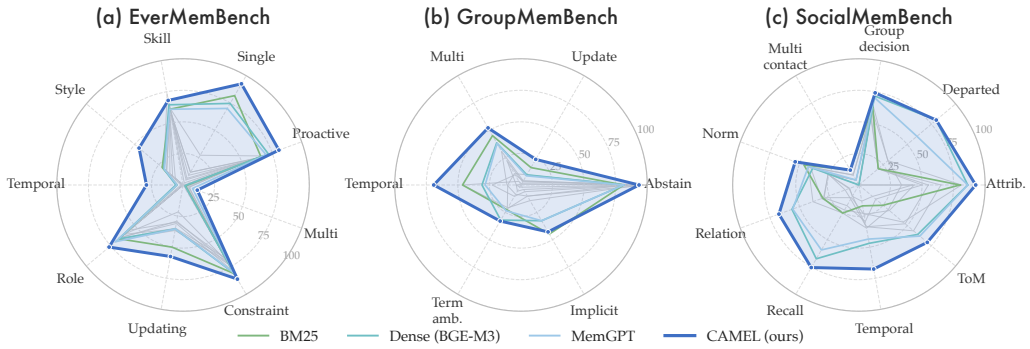


Figure 3: Sub-task accuracy. Each axis denotes a sub-task and each polygon a method; grey lines represent the remaining baselines in Table 1. Complete scores appear in Appendix E.

3.2 Main results (RQ1)

To address **RQ1**, we compare CAMEL with ten baselines. Table 1 and Figure 3 report overall and sub-task performance across three benchmarks. Detaied satistics are shown in Appendix E.

❶ *Overall performance and evidence coverage.* Table 1 shows that CAMEL achieves the highest non-oracle accuracy on all three benchmarks: 62.7% on EverMemBench, 56.1% on GroupMemBench, and 70.4% on SocialMemBench, exceeding the strongest baselines by 12.5, 11.4, and 11.7 percentage points, respectively. These gains are accompanied by higher evidence recall: 11.0% versus 5.5% for BM25 on EverMemBench, and 63.4% versus 46.3% for dense retrieval on SocialMemBench. With the answer model held fixed, the joint improvements in accuracy and recall are consistent with more effective evidence selection. All evaluated memory-system baselines remain below BM25 in accuracy on EverMemBench and GroupMemBench, indicating limited benefits from existing memory mechanisms when retrieving evidence from large collaborative histories.

❷ *Sub-task performance and limitations.* Figure 3 shows that CAMEL achieves the best or tied-best non-oracle accuracy on 21 of the 24 sub-tasks. Gains are pronounced for style-related questions on EverMemBench and temporal questions across all three benchmarks, consistent with the framework's emphasis on speaker and temporal context. The exceptions are Skill on EverMemBench and Abstain and Implicit on GroupMemBench, where the strongest baselines retain an advantage. Multi-hop accuracy on EverMemBench also remains low at 12.0% versus 99.2% with gold evidence, indicating substantial room to improve retrieval for compositional questions.

> ***Takeaway :*** CAMEL improves accuracy across all three benchmarks and leads or ties on 21 of 24 sub-tasks, with higher evidence recall but persistent gaps in compositional retrieval.

4 Sensitivity to the Answer Model (RQ2)

To address **RQ2**, we repeat the comparison at $k = 5$ with Qwen2.5-3B-Instruct and DeepSeek-V4.1-Flash on benchmark subsets, holding the judge fixed (Table 3; Appendix E.4). With DeepSeek-V4.1-Flash, CAMEL exceeds the strongest baseline by 15.8, 6.7, and 14.7 percentage points on EverMemBench, GroupMemBench, and SocialMemBench, respectively. With Qwen2.5-3B-Instruct, the corresponding margins are 2.2, 2.7, and 4.0 points. Across these settings, the strongest baseline varies between lexical retrieval, dense retrieval, and MemGPT, whereas CAMEL consistently ranks first within each answer-model block. The smaller margins with Qwen accompany lower answer accuracy across methods, consistent with answer-model capability limiting how effectively retrieved evidence is interpreted and combined to answer questions across the three benchmarks.

> ***Takeaway :*** CAMEL leads across the tested backbones, with smaller gains on the 3B model. Its benefit should therefore be assessed jointly with answer-model capability.

5 SENSITIVITY TO THE EVIDENCE BUDGET (RQ3)

To address **RQ3**, we vary the evidence window $k \in \{5, 10, 20\}$ under the primary answer model and compare CAMEL with BM25, dense retrieval, and MemGPT (Table 2). On SocialMemBench, the margin over the strongest baseline decreases from 11.7 to 6.3 and 1.1 percentage points as k increases. This reduction primarily reflects the baseline improving from 58.7% to 72.2%, whereas CAMEL increases from 70.4% to 73.3% between the endpoint settings. Its intermediate result of 69.3% at $k = 10$ also shows that a larger window does not uniformly improve answer accuracy. The corresponding margin declines from 11.4 to 8.7 points on GroupMemBench but increases from 12.5 to 13.6 on EverMemBench. On GroupMemBench, CAMEL reaches 63.9% at $k = 10$ and remains unchanged at $k = 20$. Over the full range, the strongest baseline gains 10.5 points on this benchmark, compared with 7.8 points for CAMEL. On EverMemBench, it improves from 62.7% at $k = 5$ to 67.7% at $k = 20$; its $k = 5$ result exceeds the strongest baseline at $k = 20$ by 8.6 points. This pattern is consistent with corpus scale: 20 messages cover more of a 171-turn SocialMemBench network than of an EverMemBench topic with approximately 10k messages distributed across three concurrent group chats throughout a full year of enterprise collaboration.

	EverMem			GroupMem			SocialMem		
Method	5	10	20	5	10	20	5	10	20
BM25	50.2	52.8	54.1	44.7	54.9	55.2	32.1	41.8	50.8
Dense	49.2	51.0	53.7	39.3	50.6	48.6	58.7	63.0	72.2
MemGPT	48.9	50.7	53.2	37.3	48.3	48.1	56.0	60.2	69.4
CAMEL	**62.7**	**65.4**	**67.7**	**56.1**	**63.9**	**63.9**	**70.4**	**69.3**	**73.3**

Table 2: Accuracy (%) by k. **Bold** / underline: best / second. Full results are shown in Appendix E.

Takeaway : CAMEL leads across all tested budgets. Larger windows narrow the gap most on SocialMemBench, while gains remain substantial on the two enterprise benchmarks.

6 ABLATION STUDIES (RQ4)

To address **RQ4**, we compare the full system with variants that remove one of three component groups across all benchmarks at $k = 5$, using the shared answer model and evaluation protocol. Each single-group ablation retains the other two groups, measuring the effect of that group's removal within the full pipeline. Removing **Seeding** replaces hybrid retrieval with BM25 and disables query decomposition. Removing **Ranking** bypasses the cross-encoder and retains the initial candidate ordering. Removing **Linking and Grounding** disables graph expansion, term equivalence, speaker profiles, asker-specific processing, and temporal operations. We additionally disable all three groups to assess their combined contribution to evidence selection and contextual processing, using the same answer model and evaluation protocol throughout.

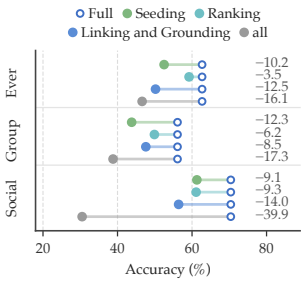


Figure 4: Component ablation. Filled / hollow markers denote ablated / full accuracy. Full results appear in Appendix E.5.

Figure 4 shows the largest accuracy drops from removing **Linking and Grounding** on EverMemBench (12.5 points) and SocialMemBench (14.0), and **Seeding** on GroupMemBench (12.3). Removing **Ranking** also reduces accuracy by 3.5–9.3 points, indicating gains from reranking beyond candidate retrieval. On SocialMemBench, the 9.3-point loss from removing Ranking is comparable to the 9.1-point loss from removing Seeding, suggesting that both candidate generation and ordering remain useful in this smaller-corpus setting. Disabling all three lowers accuracy to 46.6%, 38.8%, and 30.5%, corresponding to drops of 16.1, 17.3, and 39.9 points, respectively. These losses exceed those of each individual ablation on the corresponding benchmark, with the most pronounced decrease occurring on SocialMemBench.

Takeaway : All three groups improve accuracy, with Seeding or Linking and Grounding contributing most. Joint removal causes the largest accuracy losses on every benchmark.

Method	GPT-5.6-Luna			Qwen2.5-3B			DeepSeek V4.1 Flash		
	EverM	Group	Socia	EverM	Group	Socia	EverM	Group	Socia
Retrieval baselines									
BM25	<u>50.2</u>	<u>44.7</u>	32.1	<u>31.7</u>	26.7	20.2	47.0	38.2	31.6
Dense (BGE-M3)	49.2	39.3	<u>58.7</u>	30.8	35.0	<u>32.8</u>	<u>47.5</u>	39.0	47.7
GraphRAG	36.8	25.4	28.6	19.2	16.6	12.9	34.7	20.6	23.7
Memory systems									
Mem0	37.8	32.6	27.4	19.2	17.1	7.9	36.0	21.2	34.6
Zep	37.2	19.6	17.5	18.3	16.7	7.7	35.8	20.8	14.2
HippoRAG	35.3	30.9	29.8	21.7	15.8	8.4	41.2	39.6	15.4
RAPTOR	38.7	—	34.6	22.5	—	9.5	37.8	—	27.5
ReadAgent	41.6	25.5	34.9	27.5	20.9	18.7	40.5	25.9	31.4
MemGPT	48.9	37.3	56.0	26.7	<u>39.0</u>	32.1	43.3	<u>44.1</u>	<u>53.2</u>
Flat Memory	39.2	27.2	25.8	20.0	16.8	8.1	37.3	20.9	34.8
CAMEL (ours)	**62.7**	**56.1**	**70.4**	**33.9**	**41.7**	**36.8**	**63.3**	**50.8**	**67.9**

Table 3: Backbone transfer.**Bold** and <u>underlined</u> mark the best and second best results in each block.

7 Related Work

Agent Memory Systems. Existing approaches organize past information through fact extraction, structured representations, and context management. Mem0 (Chhikara et al., 2025) extracts and consolidates conversational information, while Zep (Rasmussen et al., 2025) maintains a temporal knowledge graph that tracks evolving facts and their historical relationships. MemGPT (Packer et al., 2023) manages information across memory tiers to support interactions beyond the active context window. Related approaches to long-term knowledge access include HippoRAG (Gutiérrez et al., 2024), which combines knowledge graphs with personalised PageRank, and RAPTOR (Sarthi et al., 2024), which retrieves from a hierarchy of recursively generated summaries. ReadAgent (Lee et al., 2024) uses gist memories while retaining access to the original text for detailed lookup. *For collaborative histories, CAMEL combines cross-role linking with speaker and temporal grounding to assemble complementary evidence while preserving message attribution.*

Multi-Party Dialogue and Collaborative Memory. Research on multi-party dialogue examines how utterances relate to participants and their conversational context. Ouchi & Tsuboi (2016) jointly model participant identities and dialogue content for addressee and response selection. Query-based meeting summarization, exemplified by QMSum (Zhong et al., 2021), studies how to locate and summarize relevant spans of a multi-party discussion in response to a query. Recent benchmarks extend evaluation to persistent memory across group interactions. EverMemBench (Hu et al., 2026) evaluates long-horizon recall, memory awareness, and user profile understanding in collaborative dialogues; GroupMemBench (Yang et al., 2026) emphasizes group dynamics, speaker-grounded beliefs, and role-dependent language; and SocialMemBench (Owolabi, 2026) examines memory for shared histories, relationships, and norms in social groups. These benchmarks make speaker attribution and evolving conversational context explicit dimensions of memory evaluation.

8 Conclusion

We presented CAMEL, a collaborative memory framework for retrieving and organizing evidence across speakers and time. Its typed memory graph preserves speaker attribution and timestamps and links equivalent terms across roles, without LLM calls at ingestion. At query time, the framework combines candidate retrieval with speaker and temporal grounding to assemble complementary messages within a limited context budget. Extensive experiments across three collaborative memory benchmarks demonstrate consistent improvements in overall accuracy over competitive retrieval and memory baselines. Further analyses show that these gains vary with the evidence budget and answer-model capability. These findings support the importance of jointly considering evidence coverage, attribution, and temporal context in collaborative memory. An important direction for future work is retrieval conditioned on previously recovered evidence to improve coverage of information that is difficult to locate from the query alone.

AI USE STATEMENT

Generative AI tools were used to assist with drafting and polishing the manuscript, including improving clarity, grammar, and organization. The authors are responsible for the final text, claims, and research artifacts.

ETHICS STATEMENT

This study evaluates memory systems on existing benchmarks of simulated multi-party interactions. We credit the source benchmarks, and their data remain subject to the original usage conditions.

REPRODUCIBILITY STATEMENT

Sections 2 and 3.1 describe the framework and experimental setup. Appendices C, D, and F provide evaluation protocols, implementation settings, and prompts, respectively. Detailed benchmark and ablation results are reported in Appendix E.

REFERENCES

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. M3-Embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 2318–2335. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.137.

Junle Chen, Wei Chen, Yehong Xu, Zhengjun Huang, Yuqian Wu, Zhoujin Tian, Kai Wang, Lei Wang, and Xiaofang Zhou. Trip+: Benchmarking agents in personalized interactive travel planning. *arXiv preprint arXiv:2606.21169*, 2026.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready AI agents with scalable long-term memory. In *European Conference on Artificial Intelligence (ECAI)*, pp. 2993–3000. IOS Press, 2025. doi: 10.3233/FAIA251160.

Gordon V. Cormack, Charles L. A. Clarke, and Stefan Buettcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *ACM SIGIR*, 2009.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From local to global: A graph RAG approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*, 2024.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. HippoRAG: Neurobiologically inspired long-term memory for large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Chuanrui Hu, Tong Li, Xingze Gao, Hongda Chen, Yi Bai, Dannong Xu, Tianwei Lin, Xiaohong Li, Yunyun Han, Jian Pei, and Yafeng Deng. Evaluating long-horizon memory for multi-party collaborative dialogues. *arXiv preprint arXiv:2602.01313*, 2026.

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

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

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics (TACL)*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638.

Hiroki Ouchi and Yuta Tsuboi. Addressee and response selection for multi-party conversation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2133–2143. Association for Computational Linguistics, 2016. doi: 10.18653/v1/D16-1231. URL https://aclanthology.org/D16-1231/.

Olukunle Owolabi. SocialMemBench: Are AI memory systems ready for social group settings? *arXiv preprint arXiv:2605.17789*, 2026.

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

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *ACM Symposium on User Interface Software and Technology (UIST)*, 2023. doi: 10.1145/3586183.3606763.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

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

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

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. RAPTOR: Recursive abstractive processing for tree-organized retrieval. In *International Conference on Learning Representations (ICLR)*, 2024.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Yuqian Wu, Wei Chen, Zhengjun Huang, Junle Chen, Qingxiang Liu, Kai Wang, Xiaofang Zhou, and Yuxuan Liang. Back to basics: Let conversational agents remember with just retrieval and generation. *arXiv preprint arXiv:2604.11628*, 2026.

Jingbo Yang, Kwei-Herng Lai, Xiaowen Wang, Shiyu Chang, Yaar Harari, and Evgeniy Gabrilovich. GroupMemBench: Benchmarking LLM agent memory in multi-party conversations. *arXiv preprint arXiv:2605.14498*, 2026.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, and Dragomir Radev. QMSum: A new benchmark for query-based multi-domain meeting summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5905–5921. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.naacl-main.472. URL https://aclanthology.org/2021.naacl-main.472/.

APPENDIX

CAMEL: Collaborative Agentic Memory via Evidence Linking across Speakers and Time

TABLE OF CONTENTS

A System Overview . . . 13

B Results at a Glance . . . 13

C Benchmarks and Evaluation . . . 14

C.1 Dataset details . . . 14

C.2 Evaluation metrics . . . 15

C.3 Judge reliability . . . 16

D Implementation Details . . . 17

D.1 Models . . . 17

D.2 Hyper-parameters . . . 17

E Detailed Benchmark Results . . . 19

E.1 Evidence window $k = 5$ . . . 19

E.2 Evidence window $k = 10$ . . . 20

E.3 Evidence window $k = 20$ . . . 21

E.4 Answer-model transfer . . . 22

E.5 Complete ablation results . . . 22

F Prompts . . . 23

F.1 Our system . . . 23

F.2 Answer generation . . . 23

A System Overview

Figure 5 illustrates how CAMEL assembles evidence from different speakers and dates to answer a question requiring multiple anchors.

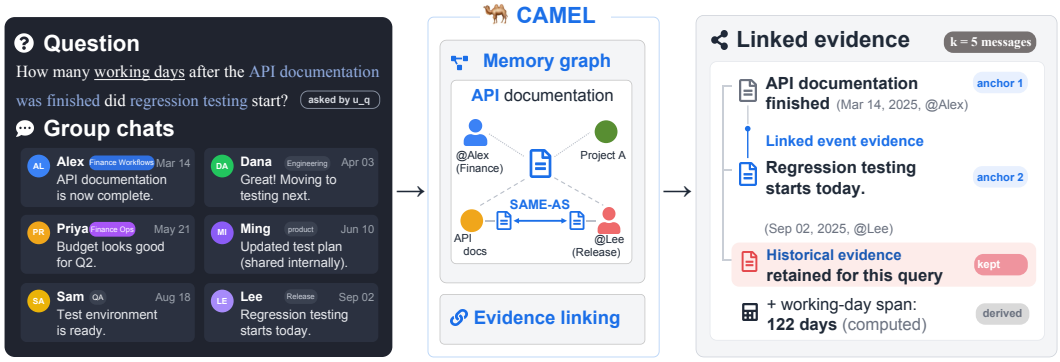


Figure 5: Illustrative overview of CAMEL. A temporal question requires documentation-completion and testing-start messages from different speakers. The example uses dates in 2025 and retains historical evidence; the span counts weekdays after March 14 through September 2.

B Results at a Glance

Figure 6 summarizes overall accuracy, sub-task performance, and sensitivity to the evidence budget. CAMEL leads the strongest baseline by 12.5, 11.4, and 11.7 percentage points across the three benchmarks; on EverMemBench, its $k = 5$ accuracy also exceeds the compared baselines at $k = 20$.

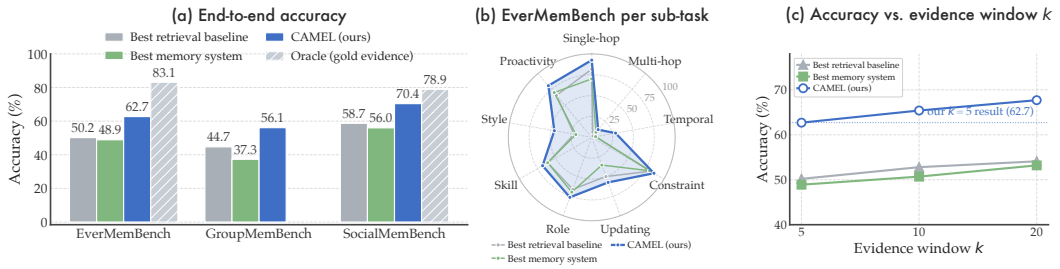


Figure 6: Results overview. **(a)** Accuracy at $k = 5$ against the best baseline in each family and the gold-evidence oracle, where available. **(b)** EverMemBench sub-task accuracy for CAMEL, BM25, and MemGPT at $k = 5$. **(c)** EverMemBench accuracy as the evidence window increases.

C Benchmarks and Evaluation

C.1 Dataset Details

Table 4 summarizes the source benchmarks. Corpus statistics and released question counts describe the complete benchmarks; the budgets and sample sizes of supplementary experiments are specified with their results.

		EverMemBench	**GroupMemBench**	**SocialMemBench**
Setting	Domain	Enterprise projects	Enterprise channels	Social groups
	Subset unit	topic	domain	network
	Subsets	5	4	43
	Time span	12 months	12 months	24 months
Corpus	Messages (total)	51,023	120,000	7,355
	Messages / subset	≈10,200	30,000	≈171
	Tokens (total)	4.2M	—	293K
	Speakers	170	45	430
	Speakers / session	—	6–18	9.9 (max 25)
	Reply threading	no	yes	yes
Annotation	Speaker roles	yes (job title)	yes (role tag)	yes (persona)
	Asking user	no	yes	no
	Decision flags	no	yes	no
	Gold evidence	date+group+index	none	turn id
	Evidence / question	6.7	—	1.9
Questions	Sub-tasks	9	6	9
	Full set	2,400	745	1,031
	Answer format	3 open / 6 MC	all open	6 open / 2 MC / 1 short
Provenance	Generator	Gemini-2.5-Pro	GPT-5	Claude Sonnet 4.5
	Licence	research use	research use	CC BY 4.0

Table 4: Detailed statistics for the three benchmarks. The three differ by two orders of magnitude in corpus size per subset (30,000 messages for a GroupMemBench domain against ≈171 turns for a SocialMemBench network), which is why a fixed evidence window behaves very differently across them. Dashes mark quantities the source paper does not report. Released question counts describe the source benchmarks; supplementary experiments specify their own evaluation subsets.

EverMemBench (Hu et al., 2026). A simulated enterprise of 170 employees across seven departments runs five projects over a calendar year (January–December 2025), each project split into three concurrent sub-project group chats. The corpus contains 51,023 turns and 4.2M tokens (≈845k tokens per topic, ≈28 turns per day, 82.8 tokens per turn); weekends carry no dialogue, which matters for the working-day arithmetic that temporal questions require. The 2,400 questions divide into fine-grained recall (762: single-hop 213, multi-hop 249, temporal 300), memory awareness (1,097: constraint 402, proactivity 427, updating 268), and profile understanding (541: style 176, skill 169, role 196). Each question carries on average 6.7 evidence references naming a date, group and message index, which we use for the oracle and for recall. Ingestion is streamed: sessions arrive in chronological order and all questions are issued after the final session, with memory persisting within a topic and reset between topics.

GroupMemBench (Yang et al., 2026). Four domains of exactly 30,000 messages each, generated by traversing a typed graph of projects, topics, phases and users, with a message mixture of 20% channel posts, 60% threaded replies and 20% cross-project bridges. Each user carries a persona (role, tone, style, expertise) sampled so that no two users share a voice, which is what produces the role-indexed vocabulary that term-ambiguity questions probe. Realism perturbations include ≈5% noise messages, ≈25% substantive disagreements, and—most relevant here—decision reversals in ≈5% of eligible phases, recorded as ground-truth metadata. Questions are produced by a solve–judge–refine loop that *accepts* a query only when a competent retrieval baseline fails it, so the benchmark is adversarial by construction. Every question is bound to an asking user, and first-person queries resolve against that user.

SocialMemBench (Owolabi, 2026). Forty-three social networks (nine close-friend, nine family, nine interest-community, eight acquaintance, eight recreational) with 430 personas, 348 sessions and 7,355 turns, spanning January 2025 to December 2026 with explicit inter-session gaps of up to 153 days. Sessions average 9.9 distinct speakers (maximum 25). Gold evidence is turn-level and speaker-attributed (mean 1.92 anchors per question). Note the scale contrast with the other two benchmarks: a network holds ≈171 turns, roughly two orders of magnitude smaller than an EverMemBench topic, yielding a larger fraction of retrieved messages at the same absolute evidence budget.

C.2 Evaluation metrics

We measure answer correctness, evidence coverage, and retrieval efficiency. Let $\mathcal{I}$ index the evaluated questions in a benchmark or sub-task. Acc, Rec, F1, and chain completeness are reported as percentages; retrieval latency is reported in milliseconds.

Answer accuracy (Acc). For question i, let $y_i \in \{0, 1\}$ denote the correctness of the generated answer under the benchmark's scoring protocol. Accuracy is

$$\mathrm{Acc} = \frac{100}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} y_i. \tag{11}$$

Multiple-choice responses are normalised to an option label and scored by exact match; an unparseable choice is incorrect. Open-ended responses receive a binary LLM judgement against the reference answer, with semantically equivalent paraphrases and the benchmark's date-format tolerances accepted. Records without a parseable judge verdict are excluded from the evaluated set. Overall accuracy weights questions equally; sub-task accuracy applies the same formula to the corresponding subset. Judge verification and scoring prompts are provided in Appendices C.3 and F.

Retrieval recall (Rec). Let G_i be the set of gold message identifiers resolved from the benchmark's evidence annotations, and let $R_i(k)$ be the set of unique message identifiers returned by retrieval at evidence budget k. For the subset $\mathcal{I}_G \subseteq \mathcal{I}$ with nonempty gold evidence and available retrieval outputs, we compute

$$r_i(k) = \frac{|R_i(k) \cap G_i|}{|G_i|}, \qquad \mathrm{Rec}(k) = \frac{100}{|\mathcal{I}_G|} \sum_{i \in \mathcal{I}_G} r_i(k). \tag{12}$$

Recall averages per-question coverage, so questions with more annotated messages do not receive greater weight. Evidence matches are determined by message identifiers, and duplicate retrievals count once. An empty retrieved set has zero recall. EverMemBench and SocialMemBench provide gold evidence; GroupMemBench does not, so its evidence-based metrics are omitted.

Accuracy–recall harmonic mean (F1). The F1 column in Table 1 combines answer correctness and evidence coverage using

$$\mathrm{F1} = \frac{2\ \mathrm{Acc}\ \mathrm{Rec}}{\mathrm{Acc} + \mathrm{Rec}}. \tag{13}$$

This score is computed from the aggregate Acc and Rec values for the reported evaluation set, with F1 defined as zero when both are zero. Its two inputs are answer accuracy and evidence recall; retrieval precision is not used in this definition. F1 is reported only when gold evidence is available.

Retrieval latency (ms). Let t_i^{start} and t_i^{end} denote wall-clock timestamps in seconds immediately before and after the retrieval call. Over questions with recorded timings, indexed by $\mathcal{I}_T$, mean latency is

$$\mathrm{Latency} = \frac{1000}{|\mathcal{I}_T|} \sum_{i \in \mathcal{I}_T} \left(t_i^{\mathrm{end}} - t_i^{\mathrm{start}} \right). \tag{14}$$

The timed interval includes candidate generation, graph processing, and reranking where applicable. Offline ingestion, answer generation, and judging are excluded, so this metric measures the cost of query-time retrieval.

Chain completeness. Let $\mathcal{I}_C \subseteq \mathcal{I}_G$ contain the evaluated multi-hop questions. We measure the proportion for which all annotated evidence is retrieved:

$$\mathrm{CC}(k) = \frac{100}{|\mathcal{I}_C|} \sum_{i \in \mathcal{I}_C} \mathbf{1}[G_i \subseteq R_i(k)]\,. \tag{15}$$

Here $\mathbf{1}[\cdot]$ is the indicator function. Each question contributes one only when every gold anchor is present; retrieving a proper subset yields zero. This diagnostic measures complete evidence coverage, whereas Rec captures partial coverage. Completeness alone does not guarantee that the answer model interprets the evidence correctly.

C.3 Judge Reliability

We assess judge reliability on the single-hop, multi-hop, and temporal open-ended tasks of EverMemBench. For each task, we manually sample 50 correct and 50 incorrect responses using human correctness labels, yielding 300 examples in total. GPT-5.6-Luna, DeepSeek-V4.1-Flash, and Qwen2.5-7B-Instruct judge the same examples using identical questions, reference answers, responses, and the grading rubric in Appendix C.2. We report each judge's correct/incorrect count and classification accuracy against the human labels. Item-level accuracy captures both false positives and false negatives, which can offset each other in aggregate verdict counts.

Judge	**Single-hop**		**Multi-hop**		**Temporal**	
	Count	**Acc. (%)**	**Count**	**Acc. (%)**	**Count**	**Acc. (%)**
Human verification	50 / 50	100	50 / 50	100	50 / 50	100
Qwen2.5-7B-Instruct	52 / 48	96	54 / 46	96	51 / 49	95
DeepSeek-V4.1-Flash	47 / 53	95	51 / 49	97	49 / 51	99
GPT-5.6-Luna (ours)	51 / 49	97	48 / 52	98	50 / 50	98

Table 5: Judge verification on 100 human-labeled responses per sub-task (50 correct and 50 incorrect). "Count" is each judge's correct/incorrect verdict split; "Acc." is classification accuracy against the human labels.

Table 5 shows high classification accuracy across all three judges. GPT-5.6-Luna achieves 97–98% accuracy across tasks (97.7% on average), compared with 97.0% for DeepSeek-V4.1-Flash and 95.7% for Qwen2.5-7B-Instruct. Every task–judge combination reaches at least 95%, and correct/incorrect counts remain within four examples of the balanced human labels. These results support the reliability of the grading protocol across the tested judge models on this verification set.

D Implementation Details

This section specifies the models, baseline implementations, and settings used by the framework in Figure 2. The complete retrieval pipeline is defined in Section 2; prompt templates are collected in Appendix F.

D.1 Models

Every model used anywhere in the paper is listed below with the role it plays and the source we took it from. None is fine-tuned: all are used in their released configurations, so CAMEL introduces no trained component of its own.

GPT-5.6-Luna. The answer model and judge throughout the main results, run at temperature 0 through the vendor's hosted API. It is the strongest model in our comparison and is held fixed across every system in Table 1, providing a shared backbone for comparing complete memory pipelines. It also serves as the default judge for the open-ended sub-tasks, a role whose reliability is verified in Appendix C.3.

DeepSeek-V4.1-Flash. A second answer model of capability comparable to the main backbone, used in the backbone-transfer experiment of Section 4 and as an out-of-family alternative judge in Appendix C.3. Being from a different vendor and training lineage than GPT-5.6-Luna, it tests whether our margin depends on one provider's model rather than on the retrieval pipeline.

Qwen2.5-3B-Instruct (Qwen Team, 2024). A compact answer model with 3B parameters. It is used only as an answer model in Section 4, where it shows that the size of CAMEL's contribution scales with the reader's ability to use a wider context.

Qwen2.5-7B-Instruct (Qwen Team, 2024). An alternative judge with 7B parameters, used in Appendix C.3 to test whether judge agreement depends on judge scale. It is never used as an answer model.

BGE-M3 (Chen et al., 2024). The dense retriever, a multilingual embedding model producing a single representation per message. It supplies the dense half of the fused candidate pool of Section 2.3 and is also the encoder for the **Dense** baseline and for every memory-system baseline that retrieves by embedding similarity, so the encoder is shared across the comparison.

BGE-reranker-v2-m3 (Chen et al., 2024). The cross-encoder, used for two distinct purposes in its default configuration: rescoring candidates during evidence assembly (Section 2.6) and evaluating contextual term pairs in both input orders to estimate SameAs relations (Section 2.2). The latter uses directional relevance as a proxy for entailment. A sigmoid maps the logits to scores, and a corpus-level quantile determines the acceptance threshold; neither operation calibrates them as entailment probabilities. Both it and BGE-M3 run on a single GPU.

D.2 Hyper-parameters

Table 6 lists the primary settings. Additional implementation constants are specified below.

Candidate and context budgets. The seed pool has size $N = \max(200, 3k)$, and graph traversal starts from at most $\max(40, k)$ leading seeds. Neighborhood expansion returns at most N messages, alias lookup at most $\lfloor N/2 \rfloor$, and each explicit query date contributes at most 20 messages within one day on either side. Asker-specific retrieval returns up to $b_s = \max(\lfloor k/2 \rfloor, 8)$ messages when its trigger is satisfied. After deduplication, the leading $\max(D, k)$ seeds and all auxiliary candidates enter the same cross-encoder scoring step; the N-candidate cap is applied afterward. Initial fusion scores determine the fallback order when reranking is disabled. The requested k remains an upper bound even when a corpus-relative cap is enabled.

Setting	Value	Role
Evidence budget		
Evidence window k	5	maximum number of retrieved messages
Retrieved-message character limit L	32,000	character limit
Ingestion		
Entity frequency quantile	0.98	threshold on the entity frequency distribution
Entity cap	3000	entities kept, by idf-weighted frequency
Seeding		
Candidate pool N	200	retained pool size, increased to $3k$ if larger
RRF constant κ	60	rank offset in Eq. 3
Sub-queries S	3	maximum decomposition width
Sub-query depth	12	base depth; each sub-query returns 3×12 hits
Sub-query weights w_s	$1.0/0.85$	original query / each sub-query, Eq. 4
Provenance fraction ρ	0.30	tracks leading sub-query hits (see text)
Fusion	RRF	rank fusion of the lexical and dense lists
Structure and ranking		
Rerank depth D	48	seed prefix scored, alongside auxiliary candidates
Graph budget β	0.10	expansion target via rounding in Eq. 10
Graph hops h	2	message-level expansion rounds in Eq. 5
Graph expansion anchors	40	seed prefix; increased to k if larger
Alias neighbours K	5	candidates per term for directional scoring
Alignment quantile	0.85	sets τ_{a} in Eq. 2
Supersession cap	$1/3$	restatement flags capped at $\max(1, \lfloor \lvert E_q \rvert / 3 \rfloor)$

Table 6: Hyper-parameter settings. All values are held fixed across the three benchmarks; none is tuned per dataset.

Provenance and selection. For s sub-queries, the implementation tracks the first $\max(1, \lfloor \rho k / s \rfloor)$ hits from each sub-query ranking. These identifiers contribute to expansion-support bookkeeping, together with graph, alias, date, and asker candidates; the top-k seed identifiers are excluded from this support set. Tracking a sub-query hit does not independently reserve it a final slot. Equation 10 rounds to the nearest integer, with ties to even. When the expansion floor is active and unmet, the algorithm considers the graph, alias, date, and asker lists in that order, using each list's existing order, and replaces selected messages without expansion support. Such insertions need not have cross-encoder scores. Remaining vacancies are filled from the seed ranking. The supersession cap in Table 6 applies only to heuristic restatement annotations; explicit decision-chain annotations are not subject to that cap.

Grounding rules. Grounding prioritizes the known asker or a participant explicitly named in the query and selects context relevant to the requested attribute. The restatement heuristic supplements Eq. 8 by requiring a shared term in the upper idf quartile, the same group and phase, and lexical update signals in both messages. It annotates earlier matching messages subject to the cap above.

Expansion and scoring details. Graph anchors receive weight $1/(1 + 0.05r)$ at zero-based rank r, and round j applies decay 0.6^j. Parent, child, and neighboring decision links have weights 1.0, 0.95, and 0.85, respectively. We use at most five children and three decision neighbors on either side. Shared-term links require idf at or above the vocabulary median and use weight 0.35 times normalized idf, divided by fan-out, with a 1.5 multiplier for query-matched terms; fan-out is capped at 12. The frontier is capped at 400 messages. A candidate's strongest incoming score receives a tie-breaking addition of $0.001 \min(v, 1)$, where v is its accumulated support in that round. Alias lookup considers up to eight messages per linked term. Term-pair proposals use a cosine floor of 0.55, and directional scoring retains at most 160 characters of context per term. Query-time reranking uses up to $2{,}000$ characters per attributed message and retains at most $2k$ scored candidates, topping up from the unscored pool tail when necessary before final selection.

E Detailed Benchmark Results

The following tables provide sub-task breakdowns of Tables 1 and 2. Inspection traced cases where methods exceed Oracle to issues in benchmark gold annotations rather than our evaluation procedure. Oracle is therefore reported as a gold-evidence reference condition.

E.1 Evidence window $k = 5$

Method	Constr	Multi	Proact	Single	Skill	Style	Temp	Role	Updat	Avg
Reference condition: gold evidence handed to the answer model										
Oracle	99.0	99.2	93.7	96.2	47.9	42.0	76.3	50.0	98.1	83.1
Retrieval baselines: no memory structure										
BM25	81.1	2.0	65.6	81.7	60.9	21.6	4.7	66.3	50.0	50.2
Dense (BGE-M3)	80.8	1.6	71.9	74.6	64.5	19.9	5.7	66.3	35.4	49.2
GraphRAG	72.9	5.2	55.5	4.7	64.5	8.0	0.0	65.3	29.1	36.8
Memory systems: structure built at ingestion										
Mem0	72.9	3.6	56.4	8.9	59.2	17.0	0.0	60.7	35.4	37.8
Zep	66.9	0.8	62.8	11.7	61.5	21.0	0.0	55.6	29.9	37.2
HippoRAG	70.4	4.8	57.4	1.9	64.5	10.8	0.0	60.2	21.6	35.3
RAPTOR	66.7	2.4	60.7	6.1	**72.8**	19.9	0.0	66.8	34.7	38.7
ReadAgent	69.7	0.8	67.2	27.2	62.7	17.0	1.7	67.3	36.6	41.6
MemGPT	80.1	8.0	70.0	70.0	60.9	19.3	4.7	69.9	35.4	48.9
Flat Memory	78.4	2.0	60.9	11.7	61.5	13.6	0.0	64.3	31.0	39.2
CAMEL (ours)	**86.1**	**12.0**	**80.8**	**92.5**	68.0	**45.5**	**29.0**	**76.5**	**57.5**	**62.7**

Table 7: Per-sub-task accuracy (%) on EverMemBench at k=5. **Bold**/underlined: best/second-best non-oracle scores, including ties.

Method	Abstain	Update	Multi	Temp	TermAmb	Implicit	Avg
Retrieval baselines: no memory structure							
BM25	81.3	15.9	45.1	46.3	22.6	**44.9**	44.7
Dense (BGE-M3)	81.3	9.3	38.5	30.9	32.1	32.7	39.3
GraphRAG	95.0	2.8	22.0	3.7	5.7	4.1	25.4
Memory systems: structure built at ingestion							
Mem0	95.0	0.9	3.3	46.3	20.8	14.3	32.6
Zep	97.1	0.9	0.5	3.7	1.9	2.0	19.6
HippoRAG	**99.3**	3.7	30.8	11.1	8.5	10.2	30.9
ReadAgent	83.5	9.3	10.4	10.5	21.7	10.2	25.5
MemGPT	81.3	8.4	37.9	28.4	23.6	32.7	37.3
Flat Memory	94.2	3.7	9.3	22.8	10.4	6.1	27.2
CAMEL (ours)	93.5	**23.4**	**52.2**	**69.1**	**33.0**	42.9	**56.1**

Table 8: Per-sub-task accuracy (%) on GroupMemBench at k=5. **Bold**/underlined: best/second-best non-oracle scores, including ties. RAPTOR is omitted, consistent with Table 1.

Method	Attrib	Depart	GrpDec	MultiC	Norm	Relat	Recall	TempSh	ToM	Avg
Reference condition: gold evidence handed to the answer model										
Oracle	97.5	100.0	97.5	18.2	90.7	71.3	84.3	70.2	76.1	78.9
Retrieval baselines: no memory structure										
BM25	80.0	20.0	63.0	**13.6**	46.3	30.4	25.7	16.8	25.0	32.1
Dense	86.2	**80.0**	71.6	0.0	38.9	56.4	67.5	46.9	60.9	58.7
GraphRAG	50.0	20.0	54.3	4.5	25.9	29.8	18.9	29.4	17.4	28.6
Memory systems: structure built at ingestion										
Mem0	55.0	20.0	60.5	4.5	27.8	8.3	10.0	33.6	47.8	27.4
Zep	42.5	0.0	54.3	0.0	27.8	7.7	10.4	13.0	14.1	17.5
HippoRAG	65.0	20.0	55.6	4.5	25.9	11.6	12.9	33.6	56.5	29.8
RAPTOR	33.8	30.0	60.5	9.1	42.6	38.7	25.7	34.7	30.4	34.6
ReadAgent	81.2	30.0	64.2	9.1	40.7	31.5	22.9	22.9	45.7	34.9
MemGPT	86.2	60.0	70.4	4.5	42.6	55.8	59.4	43.5	63.0	56.0
FlatMemory	43.8	0.0	53.1	4.5	25.9	29.3	11.6	23.7	31.5	25.8
CAMEL	**92.5**	**80.0**	**74.1**	**13.6**	**53.7**	**67.4**	**75.5**	**67.6**	**70.7**	**70.4**

Table 9: Per-sub-task accuracy (%) on SocialMemBench at k=5. **Bold**/underlined: best/second-best non-oracle scores, including ties.

E.2 EVIDENCE WINDOW $k = 10$

The larger-budget comparisons include BM25, dense retrieval, MemGPT, and CAMEL. The oracle receives all annotated evidence independently of k.

Method	Constr	Multi	Proact	Single	Skill	Style	Temp	Role	Updat	Avg
Oracle	99.0	99.2	93.7	96.2	47.9	42.0	76.3	50.0	98.1	83.1
BM25	85.8	3.6	60.7	**96.2**	66.3	26.7	11.7	77.0	38.4	52.8
Dense (BGE-M3)	82.1	6.4	57.1	88.7	65.7	26.7	10.7	73.0	42.2	51.0
MemGPT	85.8	4.8	67.2	73.7	61.5	25.6	11.7	61.7	41.8	50.7
CAMEL (ours)	**88.3**	**13.7**	**81.5**	**96.2**	**73.4**	**46.0**	**34.7**	**80.6**	**59.7**	**65.4**

Table 10: Per-sub-task accuracy (%) on EverMemBench at k=10. **Bold** and underlined mark the best and second-best distinct non-oracle scores, including ties.

Method	Abstain	Update	Multi	Temp	TermAmb	Implicit	Avg
BM25	**87.1**	23.4	57.1	66.7	20.8	**59.2**	54.9
Dense (BGE-M3)	73.4	14.0	66.5	51.2	**34.0**	40.8	50.6
MemGPT	56.8	14.0	71.4	52.5	29.2	40.8	48.3
CAMEL (ours)	86.3	**25.2**	**81.9**	**74.1**	33.0	51.0	**63.9**

Table 11: Per-sub-task accuracy (%) on GroupMemBench at k=10. **Bold** and underlined mark the best and second-best distinct non-oracle scores, including ties.

Method	Attrib	Depart	GrpDec	MultiC	Norm	Relat	Recall	TempSh	ToM	Avg
Oracle	97.5	100.0	97.5	18.2	90.7	71.3	84.3	70.2	76.1	78.9
BM25	88.8	30.0	66.7	**13.6**	48.1	35.9	36.5	30.2	42.4	41.8
Dense	**92.5**	70.0	70.4	4.5	44.4	63.0	69.1	53.8	65.2	63.0
MemGPT	86.2	**80.0**	69.1	4.5	46.3	61.9	64.7	50.4	62.0	60.2
CAMEL	**92.5**	**80.0**	**76.5**	9.1	**55.6**	**64.6**	**73.5**	**66.4**	**69.6**	**69.3**

Table 12: Per-sub-task accuracy (%) on SocialMemBench at k=10. **Bold** and underlined mark the best and second-best distinct non-oracle scores, including ties.

E.3 EVIDENCE WINDOW $k = 20$

Method	Constr	Multi	Proact	Single	Skill	Style	Temp	Role	Updat	Avg
Oracle	99.0	99.2	93.7	96.2	47.9	42.0	76.3	50.0	98.1	83.1
BM25	82.1	2.8	71.4	92.5	53.8	15.3	27.0	65.3	49.6	54.1
Dense (BGE-M3)	89.3	3.6	64.4	96.2	58.0	23.3	11.7	**78.1**	42.2	53.7
MemGPT	85.8	5.2	60.7	92.5	65.7	26.7	15.3	69.4	46.3	53.2
CAMEL (ours)	**92.8**	**13.7**	**79.4**	**100.0**	**71.0**	**44.3**	**46.3**	**78.1**	**65.3**	**67.7**

Table 13: Per-sub-task accuracy (%) on EverMemBench at k=20. **Bold** and underlined mark the best and second-best distinct non-oracle scores, including ties.

Method	Abstain	Update	Multi	Temp	TermAmb	Implicit	Avg
BM25	**66.2**	**23.4**	62.1	71.0	33.0	**63.3**	55.2
Dense (BGE-M3)	43.2	14.0	66.5	52.5	51.9	53.1	48.6
MemGPT	38.1	9.3	66.5	57.4	**57.5**	40.8	48.1
CAMEL (ours)	62.6	**23.4**	**85.7**	**75.9**	52.8	59.2	**63.9**

Table 14: Per-sub-task accuracy (%) on GroupMemBench at k=20. **Bold** and underlined mark the best and second-best distinct non-oracle scores, including ties.

Method	Attrib	Depart	GrpDec	MultiC	Norm	Relat	Recall	TempSh	ToM	Avg
Oracle	97.5	100.0	97.5	18.2	90.7	71.3	84.3	70.2	76.1	78.9
BM25	83.8	50.0	60.5	**9.1**	53.7	51.4	47.0	40.8	59.8	50.8
Dense	93.8	**100.0**	75.3	**9.1**	**64.8**	66.9	**79.1**	66.4	75.0	72.2
MemGPT	92.5	90.0	71.6	4.5	57.4	62.4	77.1	64.9	73.9	69.4
CAMEL	**95.0**	70.0	**76.5**	**9.1**	63.0	**69.1**	77.9	**70.6**	**77.2**	**73.3**

Table 15: Per-sub-task accuracy (%) on SocialMemBench at k=20. **Bold** and underlined mark the best and second-best distinct non-oracle scores, including ties.

E.4 ANSWER-MODEL TRANSFER

Table 3 reports the comparison with Qwen2.5-3B-Instruct and DeepSeek-V4.1-Flash, alongside the primary-backbone results from Table 1. The judge remains GPT-5.6-Luna across answer models. The alternative-backbone experiments use benchmark subsets, and comparisons between memory systems are made within each answer-model block. Section 4 analyzes the resulting margins. RAPTOR is omitted on GroupMemBench, consistent with the primary comparison.

E.5 COMPLETE ABLATION RESULTS

Table 16 provides the complete results underlying Figure 4, including each component-group ablation and the variant with all three groups disabled.

Grouping rationale. The ablations follow three functional groups: seeding, ranking, and linking with grounding. The latter group spans candidate expansion and contextual interpretation: graph and term links provide related messages, speaker- and date-specific retrieval supplement the candidate pool, and participant and temporal context support interpretation of the selected evidence. These operations share attributed messages and feed the same evidence assembly pipeline, motivating their evaluation as an integrated component group. With seeding and reranking retained, joint removal tests the contribution of this group. The reported differences measure the combined effect of the group within the full system, without assigning that effect to any single mechanism.

Configuration	EverMemBench		GroupMemBench		SocialMemBench	
	Acc	Δ	Acc	Δ	Acc	Δ
Full system	62.7	—	56.1	—	70.4	—
w/o **Seeding**	52.5	−10.2	43.8	−12.3	61.3	−9.1
w/o **Ranking**	59.2	−3.5	49.9	−6.2	61.1	−9.3
w/o **Linking and Grounding**	50.2	−12.5	47.6	−8.5	56.4	−14.0
w/o all three	46.6	−16.1	38.8	−17.3	30.5	−39.9

Table 16: Component ablation. Acc denotes accuracy (%), and Δ is the difference in percentage points from the full system on the same benchmark. Component groups are defined in Section 6.

F PROMPTS

We reproduce the query-decomposition and answer-generation prompts below. Braces denote runtime substitution. SocialMemBench uses the same answer templates as EverMemBench. The grading protocol and judge verification are described in Appendices C.2 and C.3.

F.1 OUR SYSTEM

CAMEL: query decomposition

```
Decompose the question into the minimal set of independent
retrieval sub-questions needed to answer it.  Each sub-question
must name ONE concrete event/fact to look up.  Output one
sub-question per line, at most 3, no numbering, no preamble.
```

Issued once per compositional question, and the only LLM call CAMEL makes outside answer generation. Ingestion makes none.

F.2 ANSWER GENERATION

Answer generation: GroupMemBench

```
You are a careful QA agent answering questions about a multi-party
workplace group chat.  Use only the retrieved conversation
passages.
Guidelines:
- Pay attention to WHO said what, their role, and WHEN.
- The question is asked by a specific user; first-person words
('I', 'my') refer to that asker, and the answer is often in a
message the asker wrote.
- Speakers in different roles use different words for the same
thing (e.g.  an engineer's 'RLHF' is a manager's 'alignment').
Match meaning, not surface form.
- Prefer the most recent decision when passages conflict; an older
value marked [SUPERSEDED by a later update] is no longer current.
- Resolve relative dates ('by EOD Friday') against the message
timestamp and answer with an absolute YYYY-MM-DD date.
- If the passages genuinely do not contain the answer, say there is
no information available in the conversation.
First give a brief reasoning paragraph.  Then give the final answer
on a new line in exactly this format:
Final:  <answer>
Keep the Final line short and specific -- just the answer, no
explanation.
```

Shared by every method on GroupMemBench, so the comparison isolates the evidence each one supplies rather than the instruction.

Answer generation: EverMemBench, multiple choice

```
You are a rigorous question-answering assistant.  You will be
given retrieved memories from a multi-person group chat and one
multiple-choice question with four options.  Choose the single best
answer based ONLY on the memories.
- Pay attention to who said what and when.
- If memories contain contradictory information, prioritize the
most recent, and apply any later override on top of the base rule.
- Prefer the option consistent with rules the group actually
established, even when another option sounds more reasonable in
general.
- Do not prefer an option merely because it is longer or more
detailed.
- If the memories do not provide enough information to be certain,
you MUST still pick one option.  Choose the option that is least
inconsistent with the memories.  Never refuse and never say the
information is missing.
Output ONLY a single uppercase letter:  A or B or C or D.
```

Multiple-choice responses are normalized to an option letter and scored by exact match.

Answer generation: EverMemBench, open ended

```
You are an intelligent memory assistant.  Answer the question using
ONLY the retrieved memories from a multi-person group chat.
- Pay special attention to timestamps and to who said what.
- If memories are contradictory, prioritize the most recent one.
- Always convert relative time references into specific dates.
- Distinguish a genuine completion ('this task is now complete',
'deliverables archived') from an earlier intention to finish ('I
will finish this today'); only a genuine completion is the end
anchor.
- For 'how long' questions, identify the start date and the end
date, then state the interval.  If the question asks for WORKING
days, exclude Saturdays and Sundays from the count; otherwise count
calendar days.
- Be concise and specific:  answer in under about 10 words when
possible.
- Do NOT output any reasoning steps.  Output ONLY the answer text.
```

Used for the three open-ended EverMemBench categories.