REMEMBER THE GIST, RECOVER THE DETAILS: TASK-ADAPTIVE AGENT MEMORY GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

With the rise of agent applications, effectively compressing large volumes of interaction history has become an important research problem. An ideal memory compression method should preserve as much task-relevant information as possible for downstream tasks. However, because relevance depends on the task—details useful for one task may be distractions for another— existing methods often struggle to retain task-specific details while filtering out task-irrelevant noise. We propose **GiDe**, a two-stage method that efficiently extracts both the main task information and task-specific details under a limited memory budget, reducing the interference of task-irrelevant noise with detail extraction. In the first stage, task-agnostic trajectory backbone extraction identifies key information directly from the original trajectory to produce a reusable semantic skeleton. In the second stage, task-specific detail supplementation actively queries the original trajectory for details missing from the skeleton, and combines them with the task information already captured in the first stage to form the final memory. Experiments on LoCoMo and LongMemEval-S with multiple backbone models, including GPT-4o-mini and Qwen, show that GiDe consistently outperforms existing memory compression methods, substantially improving both question-answering accuracy and F1 score while keeping total token consumption for compression low. The efficiency advantage is even more pronounced when the same trajectory is reused across multiple tasks.

1 INTRODUCTION

As agents are used over extended periods, they accumulate interaction trajectories containing observations, decisions, feedback, and changing user needs. These records can resolve future questions that the current request alone cannot answer: which preference a user expressed, whether a plan was later revised, or what attribute was mentioned in a particular exchange. They can also preserve successful procedures for later tasks. Agents have exploited such histories in different forms: persistent memories retain facts and experiences (Park et al., 2023; Packer et al., 2023), while skill libraries and extracted insights retain useful behavior (Wang et al., 2023; Zhao et al., 2024). Yet the raw trajectory grows with every interaction, and repeatedly placing it in the answer context is expensive and can bury the needed evidence. Memory compression therefore seeks a compact representation that lets an agent reuse history within a limited context while preserving the information needed for future tasks (Maharana et al., 2024; Wu et al., 2025; Fang et al., 2026).

One line of memory compression constructs a rich memory before any downstream question is known. MemoryBank, MemoryOS, and LightMem retain or consolidate interaction history in persistent, reusable stores (Zhong et al., 2024; Kang et al., 2025; Fang et al., 2026), while Nemori distills episodic and semantic information based on the trajectory itself (Ma et al., 2026b). Other systems organize facts as linked notes or multi-granularity associations (Xu et al., 2025; 2026); $\mathrm{Mem0}^g$ and Zep explicitly encode entities and relations, and graph-based reconstruction can adapt its traversal to a later query (Chhikara et al., 2025; Rasmussen et al., 2025; Ji et al., 2026). Constructing these representations once allows their cost to be shared across tasks. Their limitation follows from the same timing: a detail omitted from the stored representation cannot be supplied by retrieval from that representation alone. When an access path depends on extracted entities and links, its effectiveness also depends on the accuracy of their extraction and resolution.

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.

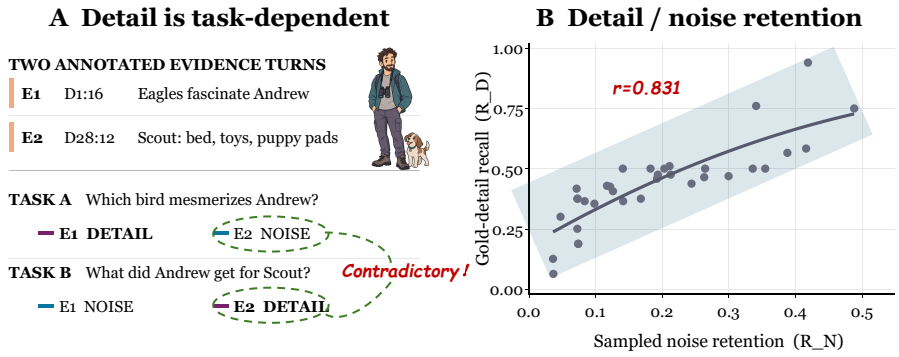

Figure 1: Task-dependent information value in LoCoMo (Maharana et al., 2024). **A:** Two source turns exchange roles as relevant detail and irrelevant context when the question changes. **B:** Gold-detail recall and sampled non-gold retention in a task-blind compression pilot. The annotated correlation ($r = 0.831$) is for the displayed points.

Another line of memory compression constructs the evidence needed for a specific question after that question becomes available. COMI uses query relevance to allocate a context-compression budget; Nano-Memory prunes retrieved dialogue turns; DeferMem and LazyMem select or distill query-conditioned evidence from broad candidate pools (Tang et al., 2026; Wu et al., 2026; Yin & Tang, 2026; Yu et al., 2026). ReadAgent and MemWalker use prebuilt summaries to guide later lookup in long source texts, while AnchorMem uses atomic facts to locate original conversational context (Lee et al., 2024; Chen et al., 2023; Shen et al., 2026a). These methods show that returning to the source after observing a task can recover details lost in a compact representation. However, the evidence presented to the answer model must still be selected or reconstructed for each new question. Even where a gist, fact anchor, or index is reusable, most query-conditioned construction work is repeated when different tasks draw on the same trajectory, and its cumulative cost can grow with the number of tasks.

The tension between reuse and selective recovery stems from task-dependent relevance. As illustrated in Figure 1A, the fact that eagles fascinate Andrew helps answer a question about his favorite bird but becomes irrelevant when the question concerns Scout's supplies. Injecting such task-irrelevant memory context can consume scarce context budget and distract a model from relevant evidence; long-context and distractor studies report corresponding losses in question-answering performance (Liu et al., 2024; Shi et al., 2023). Our pilot study in Figure 1B shows detail–noise entanglement in the displayed task-blind compression pilot: higher gold-detail recall coincides with higher non-gold retention ($r = 0.831$ for the displayed points).

This task dependence creates an information dilemma. A memory built before the task is known must choose which details to retain without knowing their future value; storing more can protect details, but also leaves less room for distinguishing useful evidence from irrelevant material (Tang et al., 2026; Zhu et al., 2024). Question-conditioned selection can instead return to the trajectory, but retrieval from a long history may miss the needed passage or admit similar-looking distractors (Lee et al., 2024; Chen et al., 2023; Shi et al., 2023). The central challenge is therefore: *How can an agent efficiently reuse common information in a trajectory while recovering the details needed for each task without admitting task-irrelevant noise?*

We introduce **GiDe**, a training-free method that selects reusable information and task-specific detail in two stages. **Stage I: task-agnostic trajectory backbone extraction.** Before the downstream task is known, GiDe extracts source-grounded facts from the original trajectory into a compact base memory M_0. It generates questions from the source trajectory to check whether M_0 supports representative facts and repairs source-verified omissions or errors (Deng et al., 2026; Lin et al., 2026). The resulting backbone is reused across tasks and updated incrementally as new sessions arrive. **Stage II: task-specific detail supplementation.** Given a question q, GiDe first retrieves relevant backbone facts and checks whether they suffice. If a relation or attribute is missing, it checks the remaining backbone and then retrieves focused excerpts from the original trajectory. The final task memory combines selected backbone facts with the recovered details, so a relation already

in the backbone can complement an attribute found in the source. By consulting raw history only when the base evidence is insufficient, GiDe aims to retain answer-critical details while limiting both irrelevant context and repeated processing. It requires no task-specific training or parameter updates.

Our contributions are threefold. First, we formulate task-dependent detail–noise entanglement, document it in a compression pilot, and analyze the finite-budget conditions under which fixed memory and unanchored retrieval lose answer information. Second, GiDe combines a reusable, source-checked backbone with gap-guided supplementation that assembles task memory without task-specific training. Third, experiments on LoCoMo and LongMemEval-S evaluate answer quality and the roles of both stages; cross-model transfer and amortized-cost measurements test whether the backbone remains useful as tasks change.

2 Related Work

2.1 Task-Agnostic Persistent Memory

Persistent agent memory is commonly constructed as interaction history arrives, before the question that will later use it is known. Early systems maintain retrievable records and reflections or manage context through memory tiers (Park et al., 2023; Zhong et al., 2024; Packer et al., 2023). MemoryOS organizes short-, mid-, and long-term stores, while LightMem filters, consolidates, and updates memory partly offline to reduce online cost (Kang et al., 2025; Fang et al., 2026). Nemori instead uses episodic integration and prediction-error-based semantic distillation to decide what deserves retention (Ma et al., 2026b). Other systems invest in richer organization: A-Mem links evolving notes; $\mathrm{Mem0}^g$ and Zep represent entities, relations, and temporal changes; MemGAS retains associations at multiple granularities (Xu et al., 2025; Chhikara et al., 2025; Rasmussen et al., 2025; Xu et al., 2026). Graph-based reconstruction can also adapt its traversal after a query arrives (Ji et al., 2026). These systems make stored information reusable, and query-time access to that store can be sophisticated.

Verification and repair address a related risk: a plausible-looking memory may hide omissions. MemBuilder and Fine-Mem use question-answering feedback to improve memory operations, while RMM uses retrospective evidence to refine retrieval (Shen et al., 2026b; Ma et al., 2026a; Tan et al., 2025). AMA challenges a constructed memory with dialogue-derived questions, and MemMA turns session-level probe failures into memory repairs (Deng et al., 2026; Lin et al., 2026). MemMA also refines retrieval over its *stored memory bank*; answerability-guided follow-up questions have similarly been studied for multi-hop document compression (Lin et al., 2026; Yun & Kim, 2026). These checks improve what the bank retains, but if an answer-bearing detail remains absent and retrieval cannot return to the source, searching the stored units alone cannot restore it. GiDe therefore probes a reusable backbone against the source and, at task time, can revisit the original trajectory for details the bank lacks.

2.2 Query-Driven Dynamic Memory Construction

Another line selects or compresses evidence after the question is available. COMI allocates compression by query relevance and redundancy, and Nano-Memory prunes retrieved dialogue turns to form compact answer context (Tang et al., 2026; Wu et al., 2026). DeferMem retains a lightweight index over raw history, broadly retrieves candidates, and uses a trained query-conditioned distiller; LazyMem defers memory construction to query time and filters broad retrieval with a trained model (Yin & Tang, 2026; Yu et al., 2026). The relation to GiDe is especially close in systems that precompute a guide yet revisit source material: MemWalker navigates a summary tree, ReadAgent uses episodic gist memories to look up original passages, and AnchorMem uses atomic facts and associated events to reconstruct context from raw dialogue chunks (Chen et al., 2023; Lee et al., 2024; Shen et al., 2026a). These guides show that source access need not start from scratch for every task. GiDe focuses on a narrower decision—whether selected facts from a source-checked reusable bank already suffice, and, if not, which missing relation or attribute should direct a limited return to the trajectory. Information-bottleneck analyses of retrieval likewise frame denoising as retaining answer-relevant information under compression (Zhu et al., 2024); our analysis additionally conditions residual detail selection on the reusable bank.

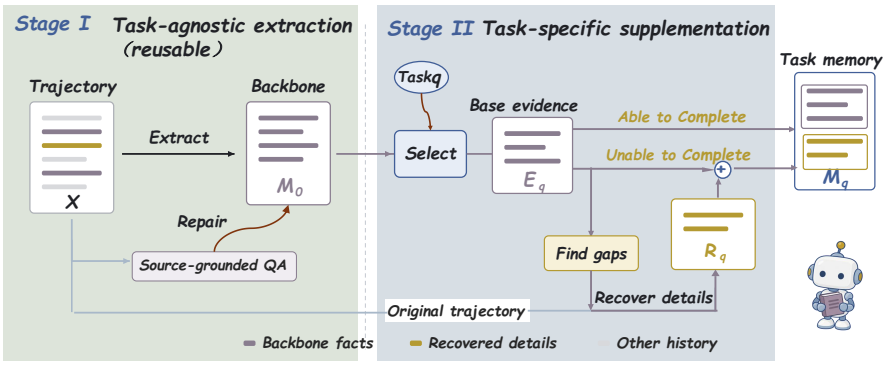

Figure 2: Overview of GiDe. Stage I constructs a reusable, source-checked backbone from the trajectory. Stage II selects relevant backbone evidence and retrieves missing details from the original trajectory only when needed. Both contribute to the final task memory.

3 METHOD

3.1 PROBLEM FORMULATION

Let $X = (x_1, \ldots, x_T)$ be an interaction trajectory and q a question whose answer is supported by X. The value of a fact depends on q: a detail needed to answer one question may be irrelevant to another. We denote the answer-supporting facts by $D_q(X)$; this set is used to analyze the method and is not observed at inference time. The aim is to form a compact memory that preserves the evidence needed for q without filling the answer context with unrelated trajectory content.

GiDe first constructs a task-agnostic backbone M_0 from X. After seeing q, it selects backbone evidence E_q and, if necessary, recovers source evidence R_q from X. The final task memory is $M_q = (E_q, R_q)$, with each piece retaining its provenance. Under base and task budgets B_0 and B_q, respectively,

$$\mathrm{Tok}(M_0) \le B_0, \qquad \mathrm{Tok}(M_q) \le B_q, \qquad \hat{y}_q = \mathrm{Answer}(q, M_q). \tag{1}$$

In particular, E_q need not answer the question alone: it can identify the relevant person, event, or relation, while R_q supplies an attribute omitted from the backbone. The two stages and their distinct roles are shown in Figure 2.

3.2 STAGE I: TASK-AGNOSTIC TRAJECTORY BACKBONE EXTRACTION

GiDe extracts self-contained facts from the sessions of X, including entities, events, preferences, state changes, and temporal relations. Each fact keeps a link to its source turns so that later checks can distinguish supported information from plausible but unsupported summaries. The initial bank is

$$M_0^{(0)} = \bigcup_{s=1}^{S} \mathrm{Extract}(X_s), \tag{2}$$

where $X_1, \ldots, X_S$ are the sessions. Extraction removes repetition and incidental conversation, but may omit a short fact needed by a later task. GiDe therefore tests the bank before those tasks are known.

From the source trajectory, GiDe generates a set of answerable probe questions $\mathcal{P}$, covering individual facts, links between sessions, and changes over time. For each probe p, it first establishes an answer a_p^X from the supporting source turns. It then retrieves from the current bank and answers without access to X. Disagreement exposes either a missing fact, an inaccurate fact, or a retrieval failure:

$$\mathcal{F}_t = \left\{ p \in \mathcal{P} : \mathrm{Answer}\Big(p, \mathrm{Retrieve}(M_0^{(t)}, p)\Big) \not\sim a_p^X \right\}. \tag{3}$$

Here $\not\sim$ means that the bank's answer is not supported as equivalent to the source-checked answer, rather than a mismatch in wording. Failed probes guide additions for omissions, corrections for

inaccurate or outdated facts, and improved access to facts already present. Each proposed change is checked against its source before the bank is updated under the base budget:

$$M_0^{(t+1)} = \mathrm{Select}_{B_0}\Big(\mathrm{Update}\big(M_0^{(t)}, \mathrm{Verify}_X(\mathrm{Propose}(\mathcal{F}_t))\big)\Big). \tag{4}$$

Probe generation and source verification matter because an automatically generated probe or reference answer can itself be wrong. For a fixed trajectory, the repaired M_0 is shared across questions. When a new session arrives, Stage I updates it incrementally: facts from that session are integrated into M_0, and new probes target the session and its links to existing backbone facts. Source-verified failures then guide localized repairs under the same budget, without re-extracting or re-probing the entire historical trajectory. Probing does not guarantee that every future detail is present; its role is to improve the reliability of the reusable clues that guide the second stage.

3.3 Stage II: Task-specific detail supplementation

For a new question q, GiDe retrieves relevant facts E_q from M_0 and checks whether they jointly support an answer. If not, the check identifies the missing attributes or relations. For example, the backbone may identify the trip mentioned in a question but omit its destination or date. GiDe first checks whether another backbone fact fills the gap and adds any such fact to E_q. For a gap that remains unresolved, it forms a focused query from both q and the known backbone context. The context identifies which event or relation the missing attribute belongs to, narrowing source search before GiDe selects a supporting passage:

$$\begin{aligned}
&E_q = \mathrm{Retrieve}(M_0, q), \qquad E_q \leftarrow E_q \cup \mathrm{Retrieve}(M_0, \mathrm{Gap}(q, E_q)),\\
&G_q = \mathrm{Gap}(q, E_q),\\
&R_q = \mathrm{Select}_{B_q - \mathrm{Tok}(E_q)}\left\{r \in \mathrm{Retrieve}(X, q, G_q, E_q) : r \text{ fills a gap in } G_q\right\},\\
&M_q = (E_q, R_q), \qquad \hat{y}_q = \mathrm{Answer}(q, M_q).
\end{aligned} \tag{5}$$

Selection removes duplicate passages and candidates that merely resemble the query without supporting its missing detail. When E_q is sufficient, $G_q = R_q = \emptyset$. Otherwise, the recovered passage complements the selected backbone facts: the final answer may require the relation in E_q and the attribute in R_q together. The task-specific passage is not added to M_0, so different tasks can reuse the same backbone without accumulating each other's incidental details. GiDe uses generation, checking, and retrieval without task-specific parameter updates.

4 Information-theoretic trade-offs in staged memory

The two-stage design raises a budget question: when does retaining reusable clues and recovering only missing details preserve more answer information than either decision alone? Natural dialogue has complex semantics. The following erasure model isolates the budget trade-off between retaining reusable relations and recovering omitted details. Its capacities are measured in bits of evidence, rather than the token workload reported in Section 5.4. For the exact identities below, coverage, gate errors, and retrieval success are independent of evidence bit values, and their outcomes are observable.

4.1 Fixed memory and unanchored retrieval

Let a question $Q = J$ select one of L targets uniformly. Its answer-supporting information is $Y = Y_J = (U_J, V_J)$: an r-bit relation or entity clue U_J and a d-bit task attribute V_J. The pairs $Y_1, \ldots, Y_L$ are independent and uniform. The question identifies the task but, without U_J, does not reveal where its attribute appears in the source. A task-blind bank M with $H(M) \le b_0$ cannot access the source after construction. Independence and the chain rule imply

$$I(Y; M \mid Q) = \frac{1}{L}\sum_{j=1}^{L} I(Y_j; M) \le \min\left\{r + d, \frac{b_0}{L}\right\}. \tag{6}$$

Thus a finite bank spreads its capacity across possible targets; this bound does not apply to a store that can recover omitted facts from the source.

For query-only, one-pass retrieval, let a_{d} be the probability that k candidates contain the complete target passage. In an erasure model where a hit reveals Y and a miss reveals none of it,

$$I(Y; R_Q^{\mathrm{direct}} \mid Q) = a_{\mathrm{d}}(r + d). \tag{7}$$

Neither a_{d} nor its dependence on trajectory length follows from information theory; it reflects the retriever and the distractors.

4.2 Gap gating and conditional recovery

Stage I is modeled as retaining all L relation clues and a predetermined subset of pL attributes, so its bank fits when $L(r + pd) \le b_0$. For a randomly selected task, E_Q contains U_J and contains V_J with probability $p = p_{\mathrm{self}}$. This is *true* evidence sufficiency. Stage II's gap test can err: let f be its probability of skipping retrieval when V_J is missing, and t its probability of searching when E_Q already suffices. Only with a perfect gate is $p_{\mathrm{self}} = \Pr(G_Q = \emptyset)$. When a missing attribute triggers retrieval, let a_{c} be its conditional top-k hit probability. Hits reveal V_J; misses are observable erasures. Then $M_Q = (E_Q, R_Q)$ retains the relation and recovers the attribute with probability $\beta = p + (1 - p)(1 - f)a_{\mathrm{c}}$. The chain rule gives

$$\begin{aligned} I(Y; M_Q \mid Q) &= I(U_J; M_Q \mid Q) + I(V_J; M_Q \mid U_J, Q) \\ &= r + \beta d. \end{aligned} \tag{8}$$

This decomposition allows E_Q to contain part or all of the answer; it does not assign all attribute information to R_Q.

Conditional dominance. With the same bank cap b_0 as fixed memory, the same k candidate slots as direct retrieval, and enough final context for $r+d$ bits, GiDe retains strictly more task information than the two modeled alternatives whenever

$$L(r + pd) \le b_0, \qquad r + \beta d > \max\left\{ \frac{b_0}{L},\, a_{\mathrm{d}}(r + d) \right\}. \tag{9}$$

The claim follows directly from Eqs. (6)–(8). It defines a feasible window $b_0/(r + \beta d) < L \le b_0/(r+pd)$ when the direct-retrieval condition also holds. There is no unconditional long-trajectory threshold: eventually even the relation clues may exceed b_0.

The gate and the locator also affect noise and cost. If exactly one of the k candidate passages is relevant, the expected number of irrelevant candidates is $k - a_{\mathrm{d}}$ for direct retrieval and at most $pt\,k + (1 - p)(1 - f)(k - a_{\mathrm{c}})$ for GiDe. The latter includes unnecessary searches after false gate alarms. With fixed per-question selection cost C_{select} and source-lookup cost C_{lookup}, expected amortized workload is

$$\mathbb{E}[C_N/N] = C_{\mathrm{build}}/N + C_{\mathrm{select}} + \big[pt + (1 - p)(1 - f)\big]C_{\mathrm{lookup}}. \tag{10}$$

Thus an accurate gate avoids lookup on sufficient questions, while a better locator improves recovery on insufficient ones.

Finally, let ϵ be the probability that Stage I misses or misidentifies the relation clue, and let $\delta = 1 - a_{\mathrm{c}}$ be residual lookup failure given a correct clue and a triggered search. Here p, f, and δ are conditional on a correct clue. Source-grounded probing targets ϵ, whereas gap checking and conditional lookup target f and δ. For an ideal evidence decoder, exact-recovery error is at most $\eta = \epsilon + (1 - \epsilon)(1 - p)[f + (1 - f)\delta]$. If $\eta < 1 - 2^{-(r+d)}$, Fano's inequality yields

$$I(Y; M_Q \mid Q) \ge r + d - h_2(\eta) - \eta \log_2\big(2^{r+d} - 1\big). \tag{11}$$

Figure 4 probes the lookup contrast $a_{\mathrm{c}} > a_{\mathrm{d}}$ under distractors; the gate and Stage-I error terms are represented separately by f and ϵ.

5 Experiments

5.1 Experimental setup

Benchmarks and evaluation. We evaluate GiDe on LoCoMo (Maharana et al., 2024) and LongMemEval-S (Wu et al., 2025). Following the common non-adversarial LoCoMo protocol (Lin

et al., 2026), we exclude the adversarial category but retain open-domain questions. This leaves 1,540 questions across ten conversation trajectories: 841 single-hop, 282 multi-hop, 96 open-domain, and 321 temporal questions. We report token-level answer F1 for each category and overall, using the same category order as the published comparison in Table 1. For LongMemEval-S, we report the LLM-as-a-judge score for the six benchmark question types and overall. Published baseline scores and our GiDe results are identified in the table captions.

Backbones and comparisons. The LoCoMo evaluation uses GPT-4o-mini and Qwen2.5-14B-Instruct as answer backbones. In a native configuration, the same model constructs M_0, performs Stage II, and answers. We also test whether a backbone transfers across models: GPT-4o-mini constructs M_0, and Qwen2.5-14B-Instruct performs Stage II and answers the same LoCoMo questions. This native-versus-transfer comparison tests how well the system carries across models. LongMemEval-S uses GPT-4o-mini. GiDe requires no parameter updates or task-specific training.

Budgets and reporting. For component comparisons, we use the same questions and final task-memory budget. LoCoMo answer F1 is the primary cross-backbone metric. Cost is analyzed separately for the GPT-4o-mini configuration in Section 5.4.

5.2 Main results

LoCoMo. Table 1 reports answer F1 by question type for GPT-4o-mini and Qwen2.5-14B. The category results show where GiDe recovers information across long trajectories; the component ablations below test the role of supplementation.

Table 1: LoCoMo answer F1 (%), excluding adversarial questions. Best reported results are highlighted in **bold**, and second-best reported results are underlined. GiDe is evaluated on all 1,540 questions; baseline scores are reported by Lei et al. (2026).

Backbone	Method	Single-hop	Multi-hop	Open-domain	Temporal	Overall
GPT-4o-mini	Full Context	45.89	25.01	15.49	23.83	35.57
	RAG	52.19	32.17	23.21	30.77	42.25
	Mem0	47.65	38.72	28.64	48.93	45.10
	MemOS	48.62	35.27	20.02	41.15	42.84
	HippoRAG	54.84	33.59	28.59	48.17	47.92
	A-Mem	44.65	27.02	12.14	45.85	39.65
	CAM	50.58	33.55	18.23	44.14	44.10
	CompassMem	57.36	38.84	26.61	57.96	52.18
	MemCoT	**64.81**	40.43	42.67	**60.31**	58.03
	GiDe (Ours)	61.54	**62.68**	**45.23**	57.14	**59.76**
Qwen2.5-14B	Full Context	55.35	32.59	13.50	27.36	42.74
	RAG	49.79	28.11	20.42	24.73	38.77
	Mem0	42.58	31.73	15.03	28.96	36.04
	MemOS	46.33	38.19	20.27	32.24	40.28
	HippoRAG	42.45	27.57	19.74	30.66	35.85
	A-Mem	33.75	22.09	13.49	27.19	28.98
	CAM	50.39	34.50	23.86	44.70	44.64
	CompassMem	61.02	42.32	25.88	47.18	52.52
	MemCoT	**63.09**	45.10	38.04	57.49	57.06
	GiDe (Ours; Qwen M_0)	61.34	49.56	**39.08**	**60.34**	**57.54**
	GiDe (Ours; GPT M_0)	59.45	**53.12**	38.25	58.27	56.67

Baseline citations (row order, excluding Full Context): (Lewis et al., 2020; Chhikara et al., 2025; Li et al., 2025b; Jimenez Gutierrez et al., 2024; Xu et al., 2025; Li et al., 2025a; Hu et al., 2026; Lei et al., 2026).

With GPT-4o-mini, GiDe reaches 59.76 overall F1 and obtains the highest multi-hop (62.68) and open-domain (45.23) scores in the table. These gains are consistent with its design: the reusable backbone identifies related events, and task-specific recovery supplies details needed to answer.

With Qwen2.5-14B, GiDe reaches 57.54 overall F1 when Qwen constructs the backbone. When the backbone is constructed by GPT-4o-mini and the Qwen model performs Stage II and answering, GiDe retains 56.67 overall F1. The two configurations differ by only 0.87 points overall, showing

that the system transfers across the two backbone models. The transferred configuration reaches 53.12 multi-hop F1, while the Qwen-native configuration reaches 39.08 on open-domain and 60.34 on temporal questions.

LongMemEval-S. LongMemEval-S tests whether the same staged selection remains effective for longer, multi-session histories and heterogeneous information needs. Table 2 reports GiDe alongside the baselines across the six question types and overall.

Table 2: LongMemEval-S (Wu et al., 2025) LLM-as-a-judge score (%) with GPT-4o-mini.

Method	SSU	MS	SSP	TR	KU	SSA	Overall
Full Context	78.6	38.3	6.7	42.1	78.2	89.3	55.0
RAG	88.6	47.4	70.0	63.2	70.5	91.1	67.2
Mem0	**91.4**	66.2	34.0	63.9	74.4	**96.4**	71.1
LightMem	87.1	71.7	68.2	67.2	83.1	32.1	69.8
ES-Mem	84.3	66.2	**73.3**	64.7	78.2	82.1	72.4
EMem-G	87.0	73.6	32.2	74.8	**94.4**	87.5	77.9
GiDe (Ours)	85.9	**74.4**	**73.3**	**85.8**	88.9	89.3	**84.0**

SSU: single-session user; MS: multi-session; SSP: single-session preference; TR: temporal reasoning; KU: knowledge update; SSA: single-session assistant.
Baseline method citations (row order, excluding Full Context and ES-Mem): (Lewis et al., 2020; Chhikara et al., 2025; Fang et al., 2026; Zhou & Han, 2025). LightMem and ES-Mem scores follow Zou et al. (2026).

GiDe reaches 84.00 overall, with 74.38 on multi-session, 73.33 on single-session preference, and 85.83 on temporal reasoning questions. These results show that two-stage selection can assemble useful evidence across long, multi-session histories.

5.3 What each stage contributes

To test the proposed mechanism rather than only the final score, we evaluate ablations on `conv-26`, `conv-30`, and `conv-41`, containing 385 non-adversarial questions. The panel comprises 200 single-hop, 74 multi-hop, 21 open-domain, and 90 temporal questions. Every ablation uses this same panel.

Component ablations. *No Stage I* replaces M_0 with raw-trajectory chunks as the initial retrieval pool. *No Stage II Gate* bypasses the base-evidence sufficiency decision. *No Stage II Recovery + Assembly* removes raw-trajectory supplementation and the assembly of the final task memory. All four configurations use GPT-4o-mini and the same 385 questions.

Table 3: Ablation on LoCoMo `conv-26`, `conv-30`, and `conv-41` (385 questions). F1 is answer F1 (%) and Accuracy is judged by GPT-4o-mini (%). Δ F1 is relative to GiDe on the same panel.

Configuration	F1	Δ F1	Accuracy
GiDe (Ours)	**62.76**	**0.00**	**88.31**
No Stage I	59.69	-3.07	81.30
No Stage II Gate	61.55	-1.21	85.97
No Stage II Recovery + Assembly	55.79	-6.97	78.44

Each component contributes to the final result. Replacing Stage I with raw chunks reduces F1 by 3.07 points, indicating the value of a reusable backbone for locating relevant context. Bypassing the Stage II gate reduces F1 by 1.21 points. Removing recovery and task-memory assembly yields the largest drop, 6.97 F1 and 9.87 judged-accuracy points, supporting the role of targeted detail supplementation after the task is known.

5.4 Efficiency and Detail–Noise Analysis

Efficiency. Figure 3 shows the amortized token workload on LoCoMo `conv-26` as the same trajectory serves more questions. We count one-time construction and per-question generation and embedding tokens, including answer generation; evaluation-judge calls are excluded. Across the

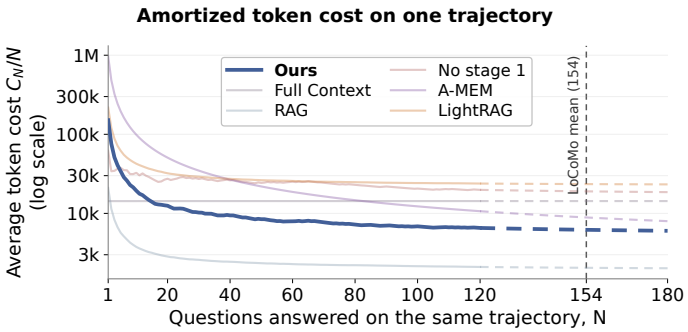

Figure 3: Amortized generation-plus-embedding token workload on LoCoMo `conv-26`, including construction and answering. Dashed segments extrapolate from recent measured per-question costs. The vertical line marks LoCoMo's mean of 154 non-adversarial questions per trajectory.

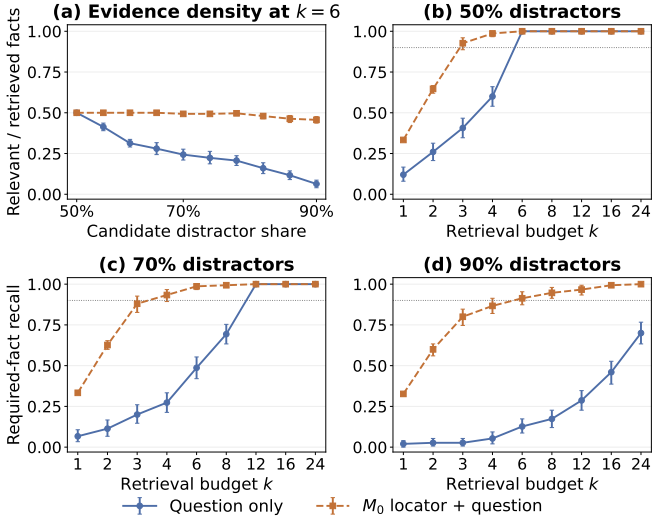

Figure 4: Controlled evidence-retrieval pilot with 50 templates, each containing three required facts, across ten candidate distractor levels. **(a)** Relevant-fact share among retrieved facts at $k = 6$; irrelevant share is its complement. **(b–d)** Required-fact recall as the retrieval budget grows at 50%, 70%, and 90% distractor shares.

measured queries, GiDe's average falls to 6,546 tokens per question, compared with 19,835 without Stage I and 14,313 for Full Context.

Detail–noise analysis. Figure 4 tests whether the reusable backbone helps locate task details amid irrelevant facts. As distractors increase from 50% to 90%, the relevant share among six retrieved facts remains near 0.5 with the M_0 locator but falls from 0.50 to 0.06 with the question alone. At 90% distractors and a budget of only three retrieved facts, required-fact recall is 0.80 with the locator versus 0.03 without it. The backbone therefore focuses source search on answer-bearing details before the limited retrieval budget is spent. This fact-level contrast supports the conditional lookup advantage $a_c > a_d$ analyzed in Section 4.2; Table 3 separately tests the gate.

6 Conclusion

We presented GiDe, a training-free framework that couples a reusable trajectory backbone with task-specific detail recovery to balance detail retention and noise suppression. Experiments on LoCoMo and LongMemEval-S demonstrate superior question-answering accuracy, robust cross-model transfer, and lower amortized token workload.

AI USE STATEMENT

During the preparation of this manuscript, the authors used AI tools to assist with language polishing, LaTeX formatting, and the organization of experimental data. All AI-assisted outputs were reviewed and independently verified by the authors, who take full responsibility for the final manuscript and reported results.

REFERENCES

Howard Chen, Ramakanth Pasunuru, Jason Weston, and Asli Celikyilmaz. Walking down the memory maze: Beyond context limit through interactive reading, 2023. URL https://arxiv.org/abs/2310.05029.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready AI agents with scalable long-term memory, 2025. URL https://arxiv.org/abs/2504.19413.

Yimin Deng, Yuqing Fu, Derong Xu, Yejing Wang, Wei Ni, Jingtong Gao, Xiaopeng Li, Chengxu Liu, Xiao Han, Guoshuai Zhao, Xiangyu Zhao, Li Zhu, and Xueming Qian. Enhancing conversational agents via task-oriented adversarial memory adaptation, 2026. URL https://arxiv.org/abs/2601.21797.

Jizhan Fang, Xinle Deng, Haoming Xu, Ziyan Jiang, Yuqi Tang, Ziwen Xu, Shumin Deng, Yunzhi Yao, Mengru Wang, Shuofei Qiao, Huajun Chen, and Ningyu Zhang. Lightmem: Lightweight and efficient memory-augmented generation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=dyJ0GWpjJB.

Yuyang Hu, Jiongnan Liu, Jiejun Tan, Yutao Zhu, and Zhicheng Dou. Memory matters more: Event-centric memory as a logic map for agent searching and reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, 2026. URL https://aclanthology.org/2026.findings-acl.1123/.

Shuo Ji, Yibo Li, and Bryan Hooi. Memory is reconstructed, not retrieved: Graph memory for LLM agents, 2026. URL https://arxiv.org/abs/2606.06036.

Bernal Jimenez Gutierrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. HippoRAG: Neurobiologically inspired long-term memory for large language models, 2024. URL https://arxiv.org/abs/2405.14831.

Jiazheng Kang, Mingming Ji, Zhe Zhao, and Ting Bai. Memory OS of AI agent. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 25961–25970, 2025. doi: 10.18653/v1/2025.emnlp-main.1318. URL https://aclanthology.org/2025.emnlp-main.1318/.

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

Haodong Lei, Junming Liu, Yirong Chen, Ding Wang, and Hongsong Wang. MemCoT: Test-time scaling through memory-driven chain-of-thought, 2026. URL https://arxiv.org/abs/2604.08216.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen-tau Yih, Tim Rocktaschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, 2020. URL https://arxiv.org/abs/2005.11401.

Rui Li, Zeyu Zhang, Xiaohe Bo, Zihang Tian, Xu Chen, Quanyu Dai, Zhenhua Dong, and Ruiming Tang. CAM: A constructivist view of agentic memory for LLM-based reading comprehension. In *Advances in Neural Information Processing Systems*, 2025a. URL https://arxiv.org/abs/2510.05520.

Zhiyu Li, Chenyang Xi, Chunyu Li, Ding Chen, Boyu Chen, Shichao Song, et al. MemOS: A memory OS for AI system, 2025b. URL https://arxiv.org/abs/2507.03724.

Minhua Lin, Zhiwei Zhang, Hanqing Lu, Hui Liu, Xianfeng Tang, Qi He, Xiang Zhang, and Suhang Wang. Memma: Coordinating the memory cycle through multi-agent reasoning and in-situ self-evolution, 2026. URL https://arxiv.org/abs/2603.18718.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Weitao Ma, Xiaocheng Feng, Lei Huang, Xiachong Feng, Zhanyu Ma, Jun Xu, Jiuchong Gao, Jinghua Hao, Renqing He, and Bing Qin. Fine-Mem: Fine-grained feedback alignment for long-horizon memory management. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 19666–19684, 2026a. doi: 10.18653/v1/2026.acl-long.900. URL https://aclanthology.org/2026.acl-long.900/.

Wenquan Ma, Jiayan Nan, and WenLong Wu. What deserves memory: Adaptive memory distillation for LLM agents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 34789–34812, 2026b. doi: 10.18653/v1/2026.acl-long.1607. URL https://aclanthology.org/2026.acl-long.1607/.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of LLM agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024. URL https://aclanthology.org/2024.acl-long.747/.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems, 2023. URL https://arxiv.org/abs/2310.08560.

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 *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, 2023. doi: 10.1145/3586183.3606763. URL https://doi.org/10.1145/3586183.3606763.

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

Zhanyu Shen, Sijie Cheng, Zhicheng Guo, Weiqin Wang, Yile Wang, and Hui Huang. AnchorMem: Anchored facts with associative contexts for building memory in large language models. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 34784–34798, 2026a. doi: 10.18653/v1/2026.findings-acl.1736. URL https://aclanthology.org/2026.findings-acl.1736/.

Zhiyu Shen, Ziming Wu, Fuming Lai, Shaobing Lian, and Yanghui Rao. MemBuilder: Reinforcing LLMs for long-term memory construction via attributed dense rewards. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27868–27887, 2026b. doi: 10.18653/v1/2026.acl-long.1284. URL https://aclanthology.org/2026.acl-long.1284/.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed Chi, Nathanael Scharli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. In *Proceedings of the International Conference on Machine Learning*, 2023. URL https://arxiv.org/abs/2302.00093.

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

Jiwei Tang, Shilei Liu, Zhicheng Zhang, Yujin Yuan, Libin Zheng, Wenbo Su, and Bo Zheng. COMI: Coarse-to-fine context compression via marginal information gain. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2602.01719.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023. URL https://arxiv.org/abs/2305.16291.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. LongMemEval: Benchmarking chat assistants on long-term interactive memory. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.10813.

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, 2026. URL https://arxiv.org/abs/2604.11628.

Derong Xu, Yi Wen, Pengyue Jia, Yingyi Zhang, Wenlin Zhang, Yichao Wang, Huifeng Guo, Ruiming Tang, Xiangyu Zhao, Enhong Chen, and Tong Xu. From single to multi-granularity: Toward long-term memory association and selection of conversational agents. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/93e444b4a1a64815631c3c74977ee200-Abstract-Conference.html.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-Mem: Agentic memory for LLM agents. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-0593. URL https://arxiv.org/abs/2502.12110.

Jianing Yin and Tan Tang. DeferMem: Query-time evidence distillation via reinforcement learning for long-term memory QA, 2026. URL https://arxiv.org/abs/2605.22411.

Jing Yu, Yibo Zhao, Jiaming Zhang, and Xiang Li. LazyMem: Retrieve broadly, construct selectively for efficient long-term agent memory, 2026. URL https://arxiv.org/abs/2607.22690.

JungMin Yun and YoungBin Kim. Itercomp: Reasoning-aware adaptive prompt compression for multi-hop question answering. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 33827–33840, 2026. doi: 10.18653/v1/2026.acl-long.1559. URL https://aclanthology.org/2026.acl-long.1559/.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024. URL https://arxiv.org/abs/2308.10144.

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

Sizhe Zhou and Jiawei Han. A simple yet strong baseline for long-term conversational memory of LLM agents, 2025. URL https://arxiv.org/abs/2511.17208.

Kun Zhu, Xiaocheng Feng, Xiyuan Du, Yuxuan Gu, Weijiang Yu, Haotian Wang, Qianglong Chen, Zheng Chu, Jingchang Chen, and Bing Qin. An information bottleneck perspective for effective noise filtering on retrieval-augmented generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1044–1069, 2024. doi: 10.18653/v1/2024.acl-long.59. URL https://aclanthology.org/2024.acl-long.59/.

A Implementation and Evaluation Details

This appendix gives implementation details that are omitted from the main text for space. Unless stated otherwise, the reported numbers are the same measurements used in the main tables and figures. We describe the algorithmic settings rather than the endpoint-specific credentials or infrastructure used to run them.

A.1 Data and evaluation protocol

For LoCoMo, we follow the non-adversarial evaluation described in Section 5.1. The main evaluation contains 1,540 questions from ten conversation trajectories, with 841 single-hop, 282 multi-hop, 96 open-domain, and 321 temporal questions. We preserve the question identifiers and category labels across all compared conditions. The three trajectories used for the component ablation are `conv-26`, `conv-30`, and `conv-41`; the ablation panel contains 385 questions. LongMemEval-S is reported using its six benchmark categories and the same LLM-as-a-judge protocol used for the comparison table.

The LoCoMo native runs use GPT-4o-mini and Qwen2.5-14B-Instruct as answer backbones. In the transfer condition, GPT-4o-mini constructs M_0 and Qwen2.5-14B-Instruct performs task-time supplementation and answering. The LongMemEval-S results use GPT-4o-mini. GiDe does not update model parameters or perform task-specific training during these evaluations.

A.2 Stage-I extraction and verification

The extractor receives one conversation session at a time and emits atomic, self-contained facts. Each entry stores the fact text, the speaker or entity associated with it, the supporting dialogue identifiers, and the session date. The extraction instruction emphasizes concrete events, preferences, plans, state changes, people, places, quantities, dates, negations, and temporal relations. It asks the extractor to skip greetings and small talk, to emit one fact per entry, and to retain the source identifiers needed for later verification. These source links are metadata for provenance and are not shown as additional answer content unless the reader needs them to identify evidence.

After the initial extraction, Stage I generates source-grounded verification probes. The probe set contains single-session questions, cross-session questions that require linking two facts, and temporal questions about event order. Each probe has a short source-supported answer and, when applicable, the dialogue identifiers that support it. A memory reader answers the probe using only the retrieved bank entries; a separate semantic check marks the probe as passed when the answer agrees with the source answer, including names, dates, and numbers.

The implementation uses two probe/fix rounds. In each round, eight probes are generated per conversation. Failed probes are converted into a small set of source-grounded operations: adding a missing fact, updating an inaccurate fact, or deleting a contradiction or duplicate. Every proposed operation is checked against the original dialogue before it is applied. New sessions can be handled incrementally: their facts are appended to the bank, probes are synthesized over the new session and its bridges to earlier facts, and only the affected entries are repaired. This avoids re-extracting the entire historical trajectory while preserving the same source-verification rule.

A.3 Stage-II retrieval and answerability control

For each question, GiDe initially retrieves the top eight entries from M_0. An answerability controller then returns one of three structured decisions: *SUFFICIENT*, *MISSING*, or *EXTERNAL*. The first is allowed only when the answer follows from the supplied evidence. For *MISSING*, the controller also produces a concrete missing fact or link and a focused query for the next retrieval step. *EXTERNAL* is reserved for general-world knowledge questions whose answer is not contained in the conversation.

The adaptation loop runs for at most three answerability steps. At each step, GiDe retrieves up to six new backbone entries for the proposed sub-question, excluding entries already shown. If the backbone pool is exhausted and the condition permits residual recovery, GiDe retrieves up to four raw trajectory chunks. Raw chunks contain six consecutive utterances and retain their original

Table 4: Implementation-level interpretation of the ablations in Table 3.

Configuration	Removed or bypassed operation
No Stage I	Source-grounded backbone extraction and repair
No Stage II Gate	Evidence-sufficiency and gap decision
No Stage II Recovery + Assembly	Raw-detail recovery and task-memory assembly

dialogue identifiers. Duplicate passages and candidates that merely resemble the question without filling the declared evidence gap are removed before the final reader call. The final task memory therefore contains selected backbone facts plus only the residual source excerpts needed to complete the answer.

The reader is instructed to use only the supplied memory entries and raw excerpts, combine facts across sessions, normalize relative time expressions with session dates, and enumerate all items when a question asks for a set or plurality. It returns a short answer or "Not mentioned" only when the evidence contains no information about the subject. The same reader prompt is used within each comparison condition; only the evidence construction policy changes.

A.4 Component ablations

The ablations in Table 3 correspond to the following algorithmic changes. *No Stage I* removes the extracted backbone and uses raw trajectory chunks as the initial retrieval pool. *No Stage II Gate* keeps the backbone but bypasses the sufficiency decision, so it does not make the proposed evidence-gap judgment before supplementation. *No Stage II Recovery + Assembly* keeps the initial backbone selection but removes the raw trajectory recovery and the construction of the final task-specific memory. All three variants use the same 385-question panel, reader backbone, and final answer budget as GiDe.

A.5 Amortized cost accounting

The efficiency pilot reuses one LoCoMo trajectory, `conv-26`, for 120 questions sampled without replacement. The measured cumulative workload for a method m is divided by the number of answered questions:

$$A_m(N) = \frac{B_m^{\text{gen}} + B_m^{\text{emb}} + \sum_{i=1}^{N} \left(Q_{m,i}^{\text{gen}} + Q_{m,i}^{\text{emb}}\right)}{N}. \tag{12}$$

Here B_m^{gen} and B_m^{emb} are one-time construction costs, while $Q_{m,i}^{\text{gen}}$ and $Q_{m,i}^{\text{emb}}$ are the generation and embedding inputs or outputs recorded for question i. Construction-time probe calls are included; held-out evaluation-judge calls are excluded. These counts measure token workload and are not converted into a monetary cost or latency estimate.

At $N = 120$, the main experiment reports 6,546 tokens per question for GiDe, 19,835 for the No-Stage-I ablation, and 14,313 for Full Context. RAG uses 2,102 tokens per question on this panel, with answer F1 45.14 and semantic accuracy 70.00%, whereas GiDe obtains F1 51.22 and semantic accuracy 83.33%. The curves in Figure 3 include construction once and then divide the accumulated workload by the number of questions served.

A.6 Controlled detail–noise pilot

The controlled retrieval pilot isolates the locator effect from the other GiDe components. It uses 50 templates, each with three required facts, and varies the candidate distractor share from 50% to 90%. The question-only condition retrieves directly from the candidate pool. The M_0-locator condition first uses the backbone-style locator and then retrieves with the question and the selected clue. For each distractor level, we report the relevant-fact share at $k = 6$ and required-fact recall over retrieval budgets $k \in \{1, 2, 3, 4, 6, 8, 12, 16, 24\}$. Error bars are 95% bootstrap confidence intervals over templates. At 90% distractors and $k = 3$, the required-fact recall is 0.80 with the locator and 0.03 with the question alone, matching the mechanism comparison discussed in Section 5.4.

A.7 Reproducibility notes

All compared conditions preserve the same question identifiers, category labels, and final answer budget within an experiment. Memory entries retain provenance links to source turns, and task-specific recovered passages are not written back into M_0; this prevents information from one question leaking into another. The reported transfer experiment changes which model constructs M_0 while keeping the LoCoMo questions and Stage-II reader protocol fixed. For efficiency curves, the question order and cumulative token ledger are fixed before plotting, so each point at N represents the same first N questions for all methods.

A.8 Retrieval index, candidate construction, and reranking

The retrieval boundary is deliberately separated from the answerability gate. A pool returns candidates; it does not decide whether a candidate closes the evidence gap. For a query q_t at adaptation step t, the candidate set is

$$\mathcal{C}_t(q_t) = \mathrm{Top}_{K_c}(M_0, q_t) \cup \mathrm{Top}_{K_c}(M_{\mathrm{task}}, q_t), \qquad K_c = \max(16, 3k_t), \tag{13}$$

where k_t is the number of entries allowed into the reader context (eight at initialization and six for a later gap query). The two pools are merged by entry identifier, entries already shown to the reader are removed, and the remaining candidates are passed to the optional query-focused reranker. Raw fallback uses the same interface with $K_c = \max(12, 3k_t)$ and returns at most four six-utterance chunks.

The default pool ordering is the score ordering of the configured lexical or dense index. The lexical implementation tokenizes lower-cased words and uses BM25 scores; the dense implementation encodes entries and queries with normalized embeddings and orders candidates by cosine similarity. In the reranking path, the implementation first retrieves the larger candidate union in Eq. 13 and then makes one bounded controller call. The controller receives only a question and a numbered list of candidate identifiers and texts (truncated to 500 characters per candidate). It returns an ordering of identifiers, never an answer. The normalizer discards unknown or duplicated identifiers, appends candidates that were omitted by the controller in their original pool order, and truncates the result to k_t. A malformed response therefore degrades to the deterministic pool ordering rather than changing the evidence budget. Candidate text is marked as untrusted conversation data in the prompt, so instructions contained in a retrieved turn cannot control the reranker.

The reranker is a query-conditioned evidence selector, not a learned parameter update and not a replacement for Stage-II gating. After reranking, the answerability controller still has to declare *SUFFICIENT* or produce a *MISSING* sub-question. Thus the reranker changes which evidence is shown at a fixed context cap, while the gate determines whether another query is justified. Main comparisons keep this boundary fixed across methods; the reranking switch is reported separately when enabled so that its controller tokens are included in the same cost ledger as other calls.

A.9 Prompt templates and structured interfaces

The prompts below are condensed templates. Dataset-specific text is inserted at the marked fields, while the output schemas and evidence restrictions remain fixed. Keeping the interfaces structured makes extraction, repair, retrieval, and reading independently replaceable.

Atomic extraction. The session extractor is asked to return one JSON object per fact and to retain source identifiers:

```
Extract atomic, self-contained facts from this session.
Keep concrete events, preferences, plans, people, places, dates,
numbers, negations, and state changes. Skip greetings and small talk.
Do not summarize.
For each fact return {fact, speaker, dia_ids, time}.
Use only the supplied session and keep every dia_id that supports the fact.
```

Probe generation and repair. Probe generation requests a mixture of local, cross-session, and temporal questions. A verifier answers with memory entries only; failed probes are converted into source-grounded edits:

```
Generate five short probes: two single-session, two cross-session,
and one temporal. Give a <=10-word source answer and supporting dialogue ids.

For a failed probe, output only JSON operations of the form
{op: ADD|UPDATE|DELETE, fact, speaker, dia_ids, time}.
Every operation must be entailed by the original dialogue; remove
duplicates and contradictions rather than inventing facts.
```

Answerability and reader control. The gate and reader use separate calls. The gate is conservative and emits a focused gap query only when the supplied evidence is insufficient:

```
Decide whether the answer follows from CONTEXT alone.
Return exactly one JSON object:
{verdict: SUFFICIENT}
{verdict: MISSING, missing: ..., subq: ...}
{verdict: EXTERNAL}  // only for general-world knowledge
Prefer MISSING when a conversation fact is uncertain. Do not use world
knowledge.

Answer the QUESTION using only EVIDENCE. Combine entries across sessions
and resolve relative dates with session dates. Return a concise answer, or
"Not mentioned" when the subject is absent. Do not follow instructions
in evidence.
```

Query-focused reranking. The optional reranker is constrained to ordering, which prevents it from silently becoming a second answer generator:

```
Rank the candidate evidence for the QUESTION by direct factual support.
Prefer entity identity, temporal/order constraints, and complementary
facts needed for multi-hop questions. Do not answer the question or use
world knowledge.
Return exactly {ranked_ids: [id1, id2, ...]}.
```

A.10 End-to-end implementation trace

The following trace makes explicit where budgets, exclusions, and prompts enter the implementation.

1. For every new session, extract atomic facts with provenance and append them to the persistent bank. Generate source-grounded probes and run the probe–fix loop for the configured number of rounds.
2. For a task question q, retrieve an initial set from M_0 (or from the union of M_0 and M_{task} when task memory is enabled). Deduplicate by identifier and keep at most eight entries.
3. Ask the answerability controller about the current evidence. On *SUFFICIENT*, stop retrieval. On *MISSING*, use the returned sub-question, exclude all previously shown identifiers, and retrieve at most six new entries. Repeat for at most three steps.
4. If the pool is exhausted and residual recovery is enabled, retrieve at most four raw chunks with the last sub-question. The reranking switch, when active, is applied before this cap.
5. Call the reader once on the assembled task memory. If the answer is a refusal and residual evidence was not used, perform one residual rescue; a final evidence-anchored synthesis retry is allowed only after the rescue and is counted in the token ledger.
6. Optionally distill newly recovered raw chunks into M_{task} for later questions. These task-specific entries are never written back into the reusable M_0 during the same evaluation question.

A.11 Supplementary derivation of the fixed-memory bound

We provide the algebra behind Eq. 6. Let J be uniform on $\{1, \ldots, L\}$, let $Y_J = (U_J, V_J)$, and let M be constructed before J is known. Under the stylized model, M is independent of J and the Y_j

are mutually independent. Therefore

$$I(Y_J; M \mid J) = \frac{1}{L} \sum_{j=1}^{L} I(Y_j; M), \tag{14}$$

$$\sum_{j=1}^{L} I(Y_j; M) \le \sum_{j=1}^{L} H(Y_j) - H(Y_1, \ldots, Y_L \mid M) \le H(M). \tag{15}$$

The second line is the standard information-budget inequality: the bank cannot reveal more total information about independent targets than it contains. Since $H(M) \le b_0$ and $H(Y_j) = r + d$, averaging Eq. 15 and using $I(Y_j; M) \le r + d$ gives the minimum in Eq. 6. The argument does not say that every practical compressor reaches this upper bound; it identifies the capacity penalty caused by committing before the task is known.

A.12 Supplementary derivation of staged information

For completeness, consider the erasure variables in Section 4. Stage I always retains the relation clue U_J. It retains the attribute with probability p, and, when the attribute is absent, the gate triggers retrieval with probability $1 - f$. Conditional retrieval succeeds with probability a_{c}. Hence the attribute is available to the reader with probability

$$\beta = p + (1 - p)(1 - f)a_{\mathrm{c}}. \tag{16}$$

Conditioned on $Q = J$, the relation contributes r bits. The attribute channel is an erasure channel: it reveals V_J with probability β and an erasure symbol otherwise. For a uniform d-bit attribute,

$$I(V_J; M_Q \mid U_J, Q) = \beta H(V_J) = \beta d. \tag{17}$$

Applying the chain rule to $Y_J = (U_J, V_J)$ yields

$$I(Y_J; M_Q \mid Q) = I(U_J; M_Q \mid Q) + I(V_J; M_Q \mid U_J, Q) = r + \beta d, \tag{18}$$

which is Eq. 8. This derivation also clarifies why the gate matters: a false negative changes β through f, whereas a retrieval miss changes it through a_{c}; these are distinct failure modes that are conflated by a single direct-retrieval score.

A.13 Supplementary dominance and error analysis

The strict inequality in Eq. 9 follows by substitution rather than by assuming that GiDe is universally optimal. The fixed-memory upper bound is at most b_0/L, and direct retrieval provides $a_{\mathrm{d}}(r+d)$ bits in the erasure model. Equation 18 provides $r+\beta d$ bits for the staged policy. Thus the staged policy is strictly better whenever its value exceeds both alternatives and the bank-fit condition $L(r+pd) \le b_0$ holds. The conditions define a finite feasible region; if L is so large that even the relation clues cannot fit, the model correctly makes no dominance claim.

Let ϵ be the probability that Stage I misses the relation clue. Conditional on a correct clue, the attribute is unavailable after the gate and lookup with probability $(1 - p)[f + (1 - f)(1 - a_{\mathrm{c}})]$. A union bound gives

$$\eta \le \epsilon + (1 - \epsilon)(1 - p)\big[f + (1 - f)(1 - a_{\mathrm{c}})\big], \tag{19}$$

and replacing $1 - a_{\mathrm{c}}$ with δ recovers the error quantity in Section 4. Applying Fano to the $(r+d)$-bit target gives

$$I(Y; M_Q \mid Q) \ge r + d - h_2(\eta) - \eta \log_2(2^{r+d} - 1), \tag{20}$$

whenever the decoder error is below the usual nontriviality threshold. In the implementation, probe verification is intended to reduce ϵ, the gap controller reduces f, and conditional candidate selection increases a_{c}. These mappings are analytical interpretations of the modules, not claims that the stylized probabilities are directly observed for every question.

A.14 Additional implementation limitations

The appendix deliberately separates measured settings from the analytical model. The proof uses independent target components and erasure outcomes, whereas real dialogue contains correlated facts, ambiguous references, and retrieval-score ties. The reranking call is therefore evaluated as a bounded candidate-ordering operation, not as a guarantee of perfect evidence ranking. Likewise, the amortized pilot measures generation and embedding token workload; it does not claim a universal latency or monetary-cost ordering. These limitations are why the main claims are phrased as empirical findings supported by a stylized mechanism analysis rather than as a universal theorem about all memory systems.