WHENLOSS: DIAGNOSING WRITE AND RETRIEVAL BOTTLENECKS IN LONG-CONTEXT MEMORY SYSTEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-context memory systems are constrained both when writing memory and when retrieving from it, making failures difficult to diagnose under fixed budgets. To address this limitation, we introduce a four-condition diagnostic protocol: truncated full context (TFC), oracle evidence (OE), complete stored memory (CSM), and retrieved memory (RM), which work together to decompose end-to-end failures into write-side and retrieval-side gaps. Our protocol first evaluates a fixed reader on TFC and OE to set budget-limited and oracle-evidence references. Next, it evaluates CSM to measure how much answer performance survives the write stage. Finally, it evaluates RM to measure the additional drop introduced by question-time retrieval. Applying this protocol to six baselines on LongMemEval shows that write-side gaps exceed retrieval-side gaps for most systems, with four baselines robustly write-dominant under our default diagnosis margin. Motivated by this diagnosis, we introduce Expected Predictive Compression (EPC), a write-time method that first generates likely future questions and then preserves compact supporting evidence under the token budget while leaving retrieval unchanged. On 500 LongMemEval questions with three readers, EPC achieves the highest CSM score among all systems, improving over Summary (LLM) from 0.44 to 0.49 and reducing Δ_{write} to 0.04. On LoCoMo, EPC again has the lowest write-side gap (0.06). These results suggest that, for the tested fixed-budget systems, improving write-time evidence preservation can be more useful than optimizing retrieval alone.

1 INTRODUCTION

A user chats with an AI assistant for three months—sharing restaurant preferences, travel plans, project deadlines. One day they ask: "What Thai place did I mention liking in April?" The system fails. But why it fails matters: did the memory system discard that detail during the write stage, or is the detail stored in memory but missed during retrieval? The first case calls for a better write strategy; the second calls for better retrieval. End-to-end accuracy—the only metric most benchmarks report—cannot distinguish the two.

This is not a hypothetical problem. Memory-augmented LLMs now use diverse write strategies (Section 2)—but evaluations usually report only end-to-end performance, leaving practitioners unable to tell whether to invest in the write stage or the retrieval stage.

We expose these stage-level effects by evaluating the same reader under four controlled inputs that replace or bypass different parts of the memory pipeline. The resulting four-condition diagnostic protocol (Section 3) compares oracle evidence (OE) against complete stored memory (CSM) to define the write-side gap, and CSM against retrieved memory (RM) to define the retrieval-side gap. The protocol is operational—it localizes where performance degrades, without claiming to identify root causes.

Applying this protocol to six baseline systems on LongMemEval (Wu et al., 2025), we find a consistent asymmetry: write-side gaps exceed retrieval-side gaps for most baselines, with four baselines robustly write-dominant under our default diagnosis margin. For these baseline operating points, more performance is lost during the write stage than during the retrieval stage.

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.

This finding motivates using anticipated future questions to guide compression at write time. The challenge is that compression happens before the question arrives. A standard summarizer does not know which facts will matter later. Expected Predictive Compression (EPC) (Section 4) addresses this by prompting the LLM to generate likely future questions about the conversation. Those self-generated probe questions then guide evidence selection under a fixed token budget—preserving the specific facts, dates, and preferences most likely to be needed, rather than producing a generically readable summary.

Across 500 LongMemEval questions with three readers (GPT-5.2, Claude Sonnet 4, Gemini 2.5 Pro), EPC achieves the highest CSM score among all tested systems, with the lowest write-side gap ($\Delta_{\text{write}} = 0.04$). On a second benchmark (LoCoMo), EPC again has the lowest write-side gap ($\Delta_{\text{write}} = 0.06$). A cost-matched comparison separates the gain from an additional LLM call, and a budget sweep shows the advantage is largest under tight budgets—precisely where choosing the right evidence matters most.

2 Related Work

Long-context memory systems. Prior work explores a wide range of memory systems for long interactions. MemGPT (Packer et al., 2024) introduces a virtual memory hierarchy; MemWalker (Chen et al., 2023) organizes long documents as a summary tree; ReadAgent (Lee et al., 2024) stores gists and expands them at read time; MemoryBank (Zhong et al., 2024) scores memory by importance and temporal decay; and Wang et al. (2023) train a long-term memory module jointly with the model. Zhang et al. (2024) survey these memory mechanisms more broadly. Our focus is complementary: rather than proposing another end-to-end system, we ask how to localize where performance degrades in a memory pipeline.

Memory evaluation and error analysis. LongMemEval (Wu et al., 2025) provides multi-session conversations with gold turn-level evidence annotations, making it possible to evaluate memory systems against explicit supporting evidence. In retrieval-augmented generation, Xu et al. (2024b) compare retrieval-based and long-context approaches, but they evaluate end-to-end performance without separating write-side from retrieval-side degradation. Our protocol fills that gap.

Compression for downstream tasks. Prompt and context compression methods (Jiang et al., 2023; Pan et al., 2024) reduce token counts while preserving general utility. Query-aware variants—RECOMP (Xu et al., 2024a) and the broader query-focused summarization tradition (Daumé III & Marcu, 2006; Baumel et al., 2018)—go further by conditioning compression on a known downstream query. EPC instead operates before the downstream query is available. It approximates the future-question distribution via LLM self-questioning and minimizes expected answer loss over likely future questions, rather than optimizing for generic readability or responding to a known query.

3 Diagnostic Protocol

This section defines the four controlled input conditions, the resulting write-side and retrieval-side gaps, and the rule used to label the dominant bottleneck.

3.1 Problem Formulation

Consider a memory-augmented QA system operating over a conversation history $H = \{s_1, s_2, \ldots, s_n\}$ consisting of n sessions. Given a question q with gold answer y and gold evidence turns E_q drawn from H, the system must: (1) write the history into a budget-limited memory store M with capacity B tokens, and (2) retrieve content $R \subseteq M$ from memory to provide context for answering q. Here, the write stage denotes the pre-question processing that stores, indexes, evicts, or compresses history into M; the retrieval stage is the question-time selection of content from that store.

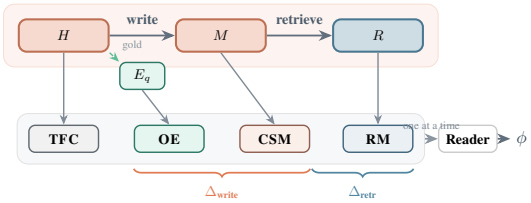

Figure 1: The four-condition diagnostic protocol. Each condition is evaluated by the same reader; $\Delta_{\text{write}} = \phi(\text{OE}) - \phi(\text{CSM})$ and $\Delta_{\text{retr}} = \phi(\text{CSM}) - \phi(\text{RM})$.

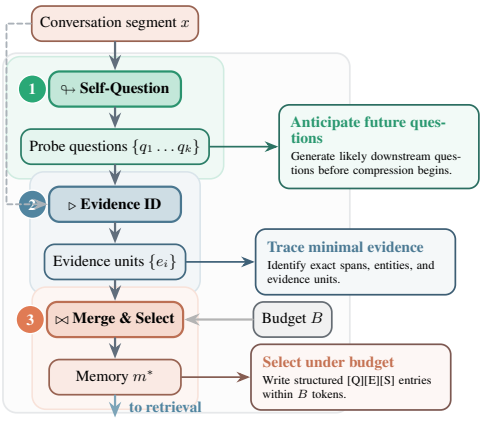

Figure 2: The EPC write pipeline. (1) Generate probe questions. (2) Identify supporting evidence. (3) Merge and select under budget B.

3.2 Evaluation Conditions

We define four evaluation conditions, each providing a different input to a fixed reader, to isolate reference performance and memory-pipeline effects (Figure 1):

TFC (Truncated Full Context): The reader receives H truncated to a fixed 32K-token context budget. No memory system is involved.

OE (Oracle Evidence): The reader receives only the gold evidence turns E_q—distractors removed, no truncation.

CSM (Complete Stored Memory): The reader receives all content in memory M after the write stage.

RM (Retrieved Memory): The reader receives the retrieved subset $R \subseteq M$.

Two diagnostic gaps localize where performance drops within the memory pipeline: OE→CSM defines the write-side gap, and CSM→RM defines the retrieval-side gap. Two caveats apply. First, the OE–TFC gap mixes distractor removal with the imposed context budget, so OE > TFC does not isolate either effect. Second, the OE→CSM gap should be read as an upper bound on write-side degradation: it can also reflect format mismatch between stored memory and reader expectations, or loss of contextual cues. These gaps are therefore operational measurements that localize where performance degrades, not causal attributions to a single mechanism.

3.3 Diagnostic Gaps

For a scoring metric ϕ, we define the two diagnostic gaps:

$$\Delta_{\text{write}} = \phi(\text{OE}) - \phi(\text{CSM}), \tag{1}$$

$$\Delta_{\text{retr}} = \phi(\text{CSM}) - \phi(\text{RM}). \tag{2}$$

These are additive: $\phi(\text{OE}) - \phi(\text{RM}) = \Delta_{\text{write}} + \Delta_{\text{retr}}$. For compact notation, we write $\Delta_{\text{w}} = \Delta_{\text{write}}$ and $\Delta_{\text{r}} = \Delta_{\text{retr}}$ below.

3.4 Diagnosis Rule

With margin $\epsilon = 0.02$:

$$\text{diagnosis} = \begin{cases} \textsc{Write} & \text{if } \Delta_{\text{w}} > \Delta_{\text{r}} + \epsilon, \\ \textsc{Retrieval} & \text{if } \Delta_{\text{r}} > \Delta_{\text{w}} + \epsilon, \\ \textsc{Mixed} & \text{otherwise.} \end{cases} \tag{3}$$

The margin prevents near-ties from being labeled as stage-specific bottlenecks.

4 EXPECTED PREDICTIVE COMPRESSION

Motivated by the write-side gap identified above, this section introduces a compression method that anticipates future questions at write time. Consider LLM summarization, the strongest query-agnostic baseline among those we test (Section 6.2): it compresses 121K-token conversations into 5K-token memories, yet its CSM score remains below .5 in our main setup. The summarizer preserves broadly salient content but omits specific dates, entity names, and preference details that downstream questions target. To address this, we propose Expected Predictive Compression (EPC), which shifts the compression objective from generic readability to preserving likely-needed evidence.

4.1 FORMULATION

Given a conversation segment x (a session in our experiments) and a token budget B, let $Q(x)$ denote a set of possible future questions that may require information from x, and let $w(q)$ denote each question's estimated likelihood. Let $A(c, q)$ denote the reader's answer to question q given context c, and L a loss function measuring answer degradation. EPC seeks the compressed memory m^* that minimizes expected answer loss:

$$m^* = \arg\min_{|m| \leq B} \sum_{q \in Q(x)} w(q) \cdot L\big(A(x, q), A(m, q)\big). \tag{4}$$

The weighted sum approximates the expectation over the future-question distribution. This is loosely motivated by rate–distortion theory (Cover & Thomas, 2006), where the budget B plays the role of a rate constraint and the distortion is future-question answer loss rather than surface-level reconstruction error. The actual implementation uses a greedy heuristic (Equation 5) rather than formal rate–distortion optimization; the connection is conceptual rather than algorithmic.

4.2 SELF-QUESTIONING PROCEDURE

Since $Q(x)$ is unknown at write time, we approximate it via LLM self-questioning: the LLM first generates probe questions that might later be asked about the segment, then uses them to guide evidence selection. The implementation treats those generated probe questions as an unweighted approximation to $Q(x)$. Figure 2 illustrates the procedure:

Step 1: Generate probe questions. Given segment x, prompt the LLM to generate $k = 5$ likely future questions targeting factual details, preferences, plans, temporal information, and state changes.

Step 2: Identify supporting evidence. For each probe question q_i, the LLM identifies compact supporting evidence: specific turns, spans, and entities.

Step 3: Merge, score, and select. Overlapping evidence spans are merged, and each evidence unit e receives a utility score:

$$u(e) = \alpha \cdot \text{coverage}(e) + \beta \cdot \text{specificity}(e) - \lambda \cdot \text{redundancy}(e). \tag{5}$$

Here coverage counts how many probe questions e supports; specificity rewards named entities, dates, and numbers; and redundancy penalizes overlap with already-selected evidence. At each greedy selection step, units are ranked by the current $u(e)$ until the budget B is exhausted, and each selected unit is written as a structured memory entry:

```
[Q] What food does the user prefer?
[E] User prefers Thai over Italian.
[S] session_12_turn_3
```

Design rationale. Even an approximate $Q(x)$ can direct compression toward useful evidence. Unlike conventional summarization, EPC prioritizes supporting evidence—producing structured, information-dense output that preserves exact entities, dates, and numbers.

Table 1: Seven memory systems. EPC shares the same budgets and embedding retriever with the Summary systems, and shares the writer model with Summary (LLM); CSM comparisons remove retrieval and isolate the write-side compression strategy.

System	Write strategy	Retrieval
Verbatim	Raw chunks, FIFO eviction	Emb. top-k
Summ. (Extractive)	Heuristic extractive	Emb. top-k
Summ. (LLM)	LLM compression (query-agnostic)	Emb. top-k
EPC (ours)	LLM self-question + evidence	Emb. top-k
MemWalker	Summary tree	Tree navigation
ReadAgent	LLM gist pages	Lookup expansion
MemoryBank	Importance + temporal decay	Importance + emb.

5 Experimental Setup

This section describes the dataset, memory systems, and evaluation setup for our main LongMemEval experiments.

5.1 Dataset

We evaluate on all 500 questions from LongMemEval (Wu et al., 2025), which pairs multi-session conversations averaging 121K tokens ($\sim$50 sessions per conversation) with gold turn-level evidence annotations. We keep the dataset's original session boundaries throughout.

5.2 Memory Systems

Table 1 lists the seven memory systems we compare. The baselines span raw storage, heuristic extraction, LLM-based compression, hierarchical memory, and importance-weighted memory to cover diverse write and retrieval designs. For controlled comparison, EPC shares the same budgets and embedding top-k retriever as the Summary systems and the same writer model as Summary (LLM). Because CSM removes retrieval, EPC's CSM gains over Summary (LLM) reflect the write-side compression strategy rather than retriever choice.

Baseline provenance. Verbatim and the two Summary variants are implemented directly in our codebase. MemWalker, ReadAgent, and MemoryBank were reimplemented following the original papers (Chen et al., 2023; Lee et al., 2024; Zhong et al., 2024) and adapted to the session-based LongMemEval format. All systems use the same dataset, budgets, and readers, so relative comparisons are controlled. Full prompts, hyperparameters, and baseline adaptation details are provided in Appendix A.

5.3 Implementation Details

Readers. We use three readers: GPT-5.2 (`gpt-5.2`), Claude Sonnet 4 (`claude-sonnet-4-20250514`), and Gemini 2.5 Pro (`gemini-2.5-pro`). All readers use temperature 0, max output tokens 200, and a simple prompt (see Appendix A.1). For TFC, truncation retains the most recent 32K tokens (tail of the conversation), discarding earlier sessions.

Memory budgets and chunking. Write budget $B = 5{,}000$ tokens; retrieval budget 5,000 tokens. Token counts use the `cl100k_base` tokenizer. For Verbatim, conversations are split into 200-token chunks with no overlap; FIFO eviction drops the oldest chunks when exceeding B. For Summary systems and EPC, each session is compressed independently; per-session budgets are allocated proportionally to the square root of session length.

Retrieval. Verbatim, Summary (Extractive), Summary (LLM), and EPC all use the same retriever: `all-MiniLM-L6-v2` sentence embeddings (Reimers & Gurevych, 2019; Wang et al., 2020) with cosine similarity, returning top-$k = 5$ chunks/entries. MemWalker uses its own tree navigation;

Table 2: Reader-level reference conditions on LongMemEval. OE $\gg$ TFC across all readers.

	CM		F1	
Reader	TFC	OE	TFC	OE
GPT-5.2	0.18	0.55	0.29	0.65
Claude S4	0.17	0.50	0.28	0.61
Gemini 2.5	0.19	0.53	0.30	0.63
Mean	.18	.53	.29	.63

Table 3: Diagnostic results (CM, 3-reader avg, $B = 5$K). Refs: TFC= 0.18, OE= 0.53. EPC CSM CI: [.46, .52], $p < .01$ vs. Summary (LLM). Token F1 yields identical system ordering.

System	CSM	RM	Δ_{w}	Δ_{r}
Verbatim	0.22	0.18	0.31	0.04
Summ. (E)	0.26	0.22	0.27	0.04
MemBank	0.30	0.25	0.23	0.05
MemWlk.	0.35	0.27	0.18	0.08
Summ. (L)	0.44	0.38	0.09	0.06
ReadAg.	0.43	0.35	0.10	0.08
EPC	**0.49**	**0.42**	**0.04**	0.07

ReadAgent uses lookup expansion; MemoryBank uses importance-weighted re-ranking over the same embedding scores.

Write-side LLM configuration. The directly compared LLM compression methods—Summary (LLM), EPC, and the two-pass baseline—use the same model (GPT-5.2, temperature 0) for compression, ensuring that CSM differences among them reflect the compression strategy, not the model. Summary (LLM) uses a single call per session; the two-pass baseline adds a second refinement call; EPC uses two calls for probe generation and evidence selection (Section 4). Utility weights in Equation 5: $\alpha = 1.0$, $\beta = 0.5$, $\lambda = 0.3$, selected in a 20-example pilot. Full prompt templates are provided in Appendix A.1.

5.4 Metrics

We report Contains Match (CM): whether the generated answer contains the gold answer substring. We also report Token F1: token-level precision/recall between generated and gold answers. Both are standard LongMemEval metrics (Wu et al., 2025). All scores are averaged over 500 questions. For the 3-reader tables, we report the mean across readers. TFC and OE are shared reference conditions, computed once per reader and reused across all systems.

CM is a strict substring test; we verify in Appendix A.3 that CM and F1 agree directionally in 95.5% of pairwise comparisons, and system rankings are identical under both metrics in all tables.

6 Results

We first verify that the reference conditions provide useful anchors, then report the main write–retrieval decomposition. The remaining subsections test whether the diagnosis is consistent with reader-independent evidence preservation, controlled perturbations, component ablations, cost matching, cross-benchmark generalization, and budget sensitivity.

6.1 Reference Conditions

Table 2 shows that all three readers exhibit OE $\gg$ TFC: CM rises from 0.18 to 0.53 and F1 from 0.29 to 0.63. This gives the diagnostic both an oracle-evidence reference and a budget-limited truncated-context reference. We now decompose the oracle-to-retrieved-memory drop into write-side and retrieval-side gaps across systems.

6.2 Diagnostic Results

Table 3 and Figure 3 show a clear spectrum: write-side gaps range from .31 (Verbatim) to .04 (EPC), while retrieval-side gaps stay between .04–.08. EPC has $\Delta_{\text{w}} = .04$, leaving the retrieval-side gap larger than the write-side gap. Appendix A.6 illustrates with concrete examples which entities, dates, and negations are lost under query-agnostic summarization and preserved by EPC.

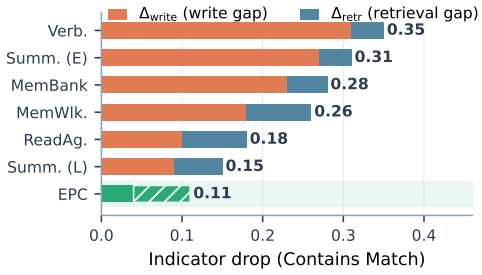

Figure 3: Write-side gap ($\Delta_w = \text{OE} - \text{CSM}$, coral) vs. retrieval-side gap ($\Delta_r = \text{CSM} - \text{RM}$, blue) for all seven systems (CM, 3-reader avg, $B = 5\text{K}$). Numbers at right: total OE→RM drop. EPC is highlighted in green, with hatching marking its retrieval-side gap, and has the smallest write-side gap.

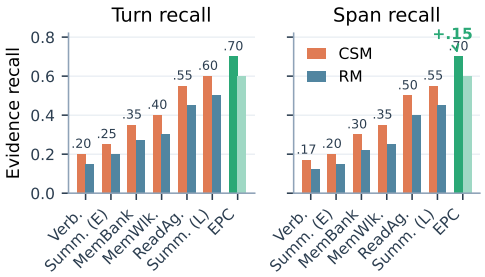

Figure 4: Evidence recall (CSM and RM). EPC preserves +.15 more gold answer entities than Summary (LLM) in CSM span recall.

Under the diagnosis rule (Section 3.4), four baselines are robustly write-dominant; Summary (LLM) is write-dominant in the aggregate but near the margin, and ReadAgent is mixed/near-boundary. EPC is the only system in Table 3 diagnosed as retrieval-dominant. EPC's CSM of .49 CM (.62 F1) has a 95% bootstrap CI of [.46, .52] and significantly exceeds Summary (LLM) under paired bootstrap ($p < 0.01$).

EPC remains the highest-CSM system for each reader individually (GPT-5.2, Claude Sonnet 4, and Gemini 2.5 Pro), so the aggregate result is not driven by a single reader.

6.3 Reader-Independent Evidence Preservation

The diagnostic gaps (Δ_{write}, Δ_{retr}) are based on *answer correctness*, which can mix evidence preservation, memory format, and reader ability. As a reader-independent check on the write-side diagnosis, we measure whether gold evidence survives in memory.

For each question, LongMemEval provides gold evidence turns. We compute two recall metrics over complete stored memory (CSM) and retrieved memory (RM): **Turn recall:** fraction of gold evidence turns preserved in memory (token-level Jaccard overlap > 0.5 with the closest memory segment). **Span recall:** fraction of gold answer entities (names, dates, numbers) found as exact substrings in memory. Both metrics are heuristic and intended as a reader-independent complement to the diagnostic gaps, not as a standalone evaluation.

Figure 4 provides the reader-independent check: CSM span recall follows the write-side ranking in Table 3, and EPC preserves 70% of gold answer entities versus 55% for Summary (LLM). Its CSM→RM span-recall drop (.70→.60) also matches the retrieval-side diagnosis, indicating that EPC's remaining gap is associated with losing preserved spans during retrieval.

6.4 Protocol Validation via Controlled Degradation

A controlled degradation experiment shows that the gaps respond selectively to their respective pipeline stages: artificially degrading write quality increases Δ_w while leaving Δ_r unchanged, and vice versa (Appendix A.8).

6.5 Probe Alignment and Component Ablation

EPC improves CSM on single-session, multi-session, and temporal questions; the largest gain is on single-session questions (+.06). Splitting questions at the median probe–question alignment gives gains of +.10 for aligned questions and +.04 for misaligned questions (Appendix B.2). Removing self-questioning drops CSM by .05; $k = 5$ probe questions suffice, and $k = 10$ provides no

Table 4: Cost-matched comparison (CM, GPT-5.2). The two-pass baseline recovers +.02 over one-pass Summary (LLM); EPC adds a further +.03 under the same two-call budget.

System	Calls	CSM	RM	Δ_{w}	Δ_{r}
Summ. (LLM) one-pass	1	0.46	0.40	0.09	0.06
Summ. (LLM) two-pass	2	0.48	0.42	0.07	0.06
EPC	2	0.51	0.44	0.04	0.07

Table 5: LoCoMo results (CM, $B = 2$K, 2-reader avg). Refs: TFC= 0.46, OE= 0.61. The CM ranking pattern matches LongMemEval.

System	CSM	RM	Δ_{w}	Δ_{r}
Verbatim	0.29	0.24	0.32	0.05
Summ. (E)	0.33	0.27	0.28	0.06
MemBank	0.37	0.31	0.24	0.06
MemWlk.	0.39	0.33	0.22	0.06
Summ. (L)	0.51	0.44	0.10	0.07
ReadAg.	0.49	0.41	0.12	0.08
EPC	0.55	0.46	0.06	0.09

additional benefit. Across six alternative utility-weight configurations, CSM stays within .45–.50 (Appendix B.3).

6.6 Cost-Matched Comparison

EPC uses two LLM calls per session versus one for Summary (LLM). To separate the effect of an additional LLM call from the effect of probe-question generation, we introduce a two-pass baseline that also uses two calls but without self-questioning: Pass 1 compresses normally, and Pass 2 performs query-agnostic refinement for information density.

Table 4 shows that a second query-agnostic refinement pass improves CSM by +.02, while EPC gains +.05 over one-pass Summary (LLM). EPC remains +.03 above the cost-matched two-pass baseline on this reader. The same ordering (EPC $>$ two-pass $>$ one-pass) holds for the other two readers. Relative to the one-pass Summary (LLM) baseline, EPC roughly doubles write latency ($\sim$30s$\rightarrow\sim$60s per session in our implementation) while roughly halving Δ_{write}.

6.7 Cross-Benchmark Generalization: LoCoMo

To test generalization, we evaluate on LoCoMo (Maharana et al., 2024), a multi-session benchmark with 348 QA pairs across 10 conversations, including two question categories absent from LongMemEval (elaboration and adversarial). For LoCoMo, we report a two-reader average using GPT-5.2 and Claude Sonnet 4.

LoCoMo conversations average $\sim$9K tokens, fitting within the 32K context budget, so TFC effectively becomes a full-context baseline. We set $B = 2{,}000$ tokens to create meaningful compression pressure (22% retention).

Table 5 shows a similar ranking pattern: EPC has the lowest Δ_{w} and is the only system where $\Delta_{\mathrm{r}} > \Delta_{\mathrm{w}}$, with a +.04 CSM gap over Summary (LLM). By category, EPC gains most on factual questions (+.06), moderately on temporal/elaboration (+.04), and least on inferential/adversarial cases (+.02).

6.8 Budget Sensitivity on LongMemEval

One question raised by the rate–distortion framing (Section 4) is whether self-questioning matters more under tight budget constraints, where choosing the right evidence is critical, than under re-

laxed budgets. We test this by sweeping the write budget $B \in \{2\text{K}, 5\text{K}, 10\text{K}, 20\text{K}\}$ on all 500 LongMemEval questions with three readers, while keeping the retrieval budget fixed at 5K tokens.

Table 10 shows that EPC's CSM advantage over Summary (LLM) shrinks from +.11 at 2K to +.01 at 20K as both systems approach the OE upper bound (.53). The relative size of the two gaps also shifts with budget: EPC reaches $\Delta_{\text{r}} > \Delta_{\text{w}}$ at 5K, Summary (LLM) at 10K, while Verbatim and Summary (Extractive) have $\Delta_{\text{w}} > \Delta_{\text{r}}$ throughout (Table 9).

7 Analysis

Operating point matters. The budget sweep (Table 10) shows that the diagnosis is not an intrinsic property of an architecture: as B grows, write-dominant systems can become retrieval-dominant. The protocol should therefore be run under the target budget and question distribution rather than inferred from the system family.

Benefits and limits of self-questioning. EPC's gains are largest under aggressive compression (+.11 at $B = 2\text{K}$), on single-session LongMemEval questions (+.06), and on factual LoCoMo questions (+.06). They shrink when budgets are generous (+.01 at $B = 20\text{K}$) or questions require compositional inference (+.02 on LoCoMo's inferential category). This suggests that self-questioning is most useful when the write stage must choose a small set of answer-relevant details before the question is known.

Relation to full-context reading. Because our TFC condition imposes a fixed 32K-token budget, full-context reading could narrow the OE–TFC gap. The diagnostic protocol is complementary: it evaluates fixed-budget memory pipelines and helps determine whether the write stage or the retrieval stage is the limiting factor within that design. On LoCoMo, where conversations fit within the context window, the protocol remains informative despite the smaller OE–TFC gap (.15 vs. .35 on LongMemEval), suggesting that the diagnostic does not depend solely on truncation-induced loss.

Limitations. The diagnostic gaps depend on the chosen reader and metric. EPC adds write-time latency, and the experiments rely on gold evidence annotations. Silver evidence remains unvalidated, and LoCoMo's shorter contexts limit generalizability to longer and more diverse settings. Appendix B.4 retains the full limitations discussion.

8 Conclusion

This paper introduced a four-condition diagnostic protocol that decomposes end-to-end memory failures into write-side and retrieval-side gaps without requiring architectural changes. Across LongMemEval and LoCoMo, the diagnostic reveals that the write-side gap is often the larger bottleneck. Motivated by this finding, we proposed EPC, which uses likely future questions to guide write-time evidence selection; it reduces the write-side gap and achieves the strongest CSM performance among the tested systems.

The broader lesson is that memory-system bottlenecks are operating-point dependent: as budgets grow or write methods improve, the limiting stage can shift from writing to retrieval. A diagnostic evaluation should therefore precede optimization to avoid improving a non-limiting stage.

Future work should reduce reliance on human evidence annotations through LLM-generated silver evidence, develop adaptive compression policies that respond to per-session diagnostic signals, and test the protocol on longer conversations and more diverse domains.

AI USE STATEMENT

During preparation of this submission, OpenAI Codex was used to convert an existing manuscript from PDF into LaTeX, adapt its layout to the ICLR template, preserve figures, reconstruct editable equations and tables, and assist with reference verification and consistency checks. The LLMs used as components of the proposed method, experimental baselines, and readers are described in Section 5 and Appendix A.

ETHICS STATEMENT

Long-term memory benchmarks and systems necessarily involve storing user-specific facts, preferences, dates, and other potentially sensitive information. Because EPC explicitly tries to preserve supporting evidence, a deployed version could also preserve sensitive information more reliably than a generic summary unless retention, deletion, access control, and redaction policies are carefully designed. More broadly, the work studies diagnostic structure rather than downstream social impact, so it should not be interpreted as an argument for indiscriminate memory retention in real applications. Reproducibility is also limited by our use of proprietary LLM readers and writers, whose behavior may change over time.

REPRODUCIBILITY STATEMENT

Section 5 specifies the datasets, memory and retrieval budgets, tokenizer, model identifiers, and decoding settings. Appendix A provides prompt templates, hyperparameters, baseline adaptation details, metric agreement checks, per-reader results, qualitative examples, the full budget sweep, controlled degradation settings, and the paired bootstrap procedure. The additional analyses in Appendix B describe probe alignment and component sensitivity. The experiments use proprietary LLM services; changes to these services can affect exact reproducibility.

REFERENCES

Tal Baumel, Matan Eyal, and Michael Elhadad. Query focused abstractive summarization: Incorporating query relevance, multi-document coverage, and summary length constraints into seq2seq models. *arXiv preprint arXiv:1801.07704*, 2018. URL https://arxiv.org/abs/1801.07704.

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

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 2 edition, 2006. ISBN 9780471241959. doi: 10.1002/047174882X.

Hal Daumé III and Daniel Marcu. Bayesian query-focused summarization. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pp. 305–312. Association for Computational Linguistics, 2006. doi: 10.3115/1220175.1220214. URL https://aclanthology.org/P06-1039/.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. LLMLingua: Compressing prompts for accelerated inference of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 13358–13376. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.825. URL https://aclanthology.org/2023.emnlp-main.825/.

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

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of LLM agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13851–13870. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.747. URL https://aclanthology.org/2024.acl-long.747/.

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*, 2024. URL https://arxiv.org/abs/2310.08560v2.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. LLMLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 963–981. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.57. URL https://aclanthology.org/2024.findings-acl.57/.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3982–3992. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1410. URL https://aclanthology.org/D19-1410/.

Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. Augmenting language models with long-term memory. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://papers.nips.cc/paper/2023/hash/ebd82705f44793b6f9ade5a669d0f0bf-Abstract-Conference.html.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

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://openreview.net/forum?id=pZiyCaVuti.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. RECOMP: Improving retrieval-augmented LMs with compression and selective augmentation. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=mlJLVigNHp.

Peng Xu, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. Retrieval meets long context large language models. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview.net/forum?id=xw5nxFWMlo.

Zeyu Zhang, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Quanyu Dai, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. A survey on the memory mechanism of large language model based agents. *arXiv preprint arXiv:2404.13501*, 2024. URL https://arxiv.org/abs/2404.13501.

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.

A Reproducibility Details

This appendix provides the prompt templates, hyperparameters, metric agreement checks, baseline adaptation details, per-reader results, qualitative examples, and bootstrap procedure used in the main experiments.

A.1 Prompt Templates

All LLM calls use temperature 0. We use the following templates, with bracketed fields replaced by the corresponding instance-specific content.

Reader prompt.

```
Based on the following context,
answer the question.
If the answer cannot be determined,
say "I don't know".

Context:
{context}

Question:
{question}
```

Summary (LLM) write prompt.

```
You are compressing conversation memory
under a hard budget.
Summarize this session from {date}
in under {max_tokens} tokens.
Preserve names, dates, numbers,
preferences, negations, state changes,
decisions, and event relations.
Stay query-agnostic: do not guess
future questions.
Return only the compressed memory text.

{session_turns}
```

EPC Step 1: probe generation.

```
You are writing long-term memory before
the future question is known.
Given this conversation session,
generate 5 likely future questions
that may require information from it.
Prefer questions about factual details,
names, dates, numbers, preferences,
plans, temporal changes, decisions,
and negations.

Return only a numbered list.

{session_turns}
```

EPC Step 2: evidence selection and memory writing.

```
You are compressing a conversation
session into long-term memory under
a hard budget of {max_tokens} tokens.

Use the probe questions to identify
minimal supporting evidence.
```

```
Keep exact entities, dates, numbers,
preferences, decisions, negations,
and temporal relations.
Avoid generic summaries.

For each selected item, use this format:
[Q] likely future question
[E] minimal supporting evidence
[S] session_{session_id}_turn_{turn_id}

Probe questions:
{probe_questions}

Conversation:
{session_turns}
```

Two-pass baseline refinement prompt.

```
You are improving an existing compressed
memory under the same hard budget of
{max_tokens} tokens.
Increase density by replacing
vague references with specific names,
dates, numbers, preferences, decisions,
negations, and event relations.
Do not guess future questions.
Return only the revised memory.

Original session:
{session_turns}

Current compressed memory:
{summary}
```

A.2 HYPERPARAMETERS

Table 6 lists the main hyperparameters used in the LongMemEval experiments.

Table 6: Main hyperparameters used in the LongMemEval experiments unless otherwise specified.

Parameter	**Value**
Write budget	5,000 tokens
Retrieval budget	5,000 tokens
Tokenizer	`cl100k_base`
Retriever	`all-MiniLM-L6-v2`
Retrieval top-k	5
Verbatim chunk size	200 tokens, no overlap
Verbatim eviction	FIFO over oldest chunks
TFC truncation	most recent 32K tokens
Writer temperature	0
Reader temperature	0
Reader max output	200 tokens
EPC probe questions	$k = 5$
EPC utility weights	$\alpha = 1.0$, $\beta = 0.5$, $\lambda = 0.3$
Budget allocation	proportional to $\sqrt{\text{session length}}$

A.3 CM AND F1 AGREEMENT

Contains Match (CM) is the primary metric in the main tables, but we also compute Token F1 for every system and condition. Across pairwise comparisons among LongMemEval system–condition

scores, CM and F1 agree directionally in 95.5% of cases, and the system rankings reported in the main tables are unchanged under Token F1. For LoCoMo, Token F1 yields the same system ranking as CM in Table 5.

A.4 Baseline Adaptations

Table 7 summarizes how each baseline system was adapted to the session-based LongMemEval setting.

Table 7: Baseline adaptations to the session-based LongMemEval setting.

System	Write	Retrieval
Verbatim	200-token chunks; FIFO eviction.	Embedding top-$k = 5$.
Summ. (E)	Score and pack turns with dates, entities, negation, preference, and decision cues.	Embedding top-$k = 5$.
Summ. (L)	Query-agnostic session compression using the same writer model and budgets as EPC.	Embedding top-$k = 5$.
MemWalker	Summary tree over sessions; internal nodes summarize child summaries.	Relevance-guided tree navigation.
ReadAgent	One gist page per session using the same writer model and boundaries.	Select relevant gists and expand under budget.
MemoryBank	LLM-estimated importance with temporal decay and session metadata.	Importance-weighted embedding reranking.
EPC	Probe generation, evidence selection, merging, and greedy budgeted selection.	Embedding top-$k = 5$.

Note that MemWalker, ReadAgent, and MemoryBank were reimplemented from their paper descriptions, not from official code. To mitigate reimplementation risk, we verified two properties: (1) the qualitative behavior of each system matches the original (e.g., MemWalker's tree depth scales logarithmically with session count; MemoryBank's importance scores decay with time as described); (2) all systems are compared under identical data, budgets, and readers. We therefore interpret absolute comparisons to reimplemented systems cautiously, while emphasizing controlled comparisons within our evaluation setup.

A.5 Per-Reader Diagnostic Results

Table 8 reports the diagnostic results for each reader individually. The system ranking is consistent across all three readers: EPC achieves the highest CSM for each reader. The robust write-dominant pattern holds for Verbatim, Summary (Extractive), MemoryBank, and MemWalker under each reader, while Summary (LLM) and ReadAgent lie near the margin.

A.6 Qualitative Case Studies

To illustrate how the diagnostic gaps correspond to concrete information loss, we present three representative LongMemEval examples, lightly paraphrased for brevity and anonymization, where Δ_{write} is large for Summary (LLM) but small for EPC.

Case 1: Dropped date. **Question:** "When did the user say they were moving to Seattle?" **Gold evidence:** `session_7, turn_14`: "I'm planning to move to Seattle around mid-March." **Summary (LLM) CSM:** The summary mentions "the user discussed relocation plans" but drops "mid-March" and "Seattle" appears only as a general topic. Reader answers "I don't know." **EPC CSM:** A probe question "When is the user relocating?" preserves the entry `[E] Moving to Seattle around mid-March` → reader answers correctly.

Case 2: Merged preferences. **Question:** "Does the user prefer Thai or Italian food?" **Gold evidence:** `session_12, turn_3`: "I definitely prefer Thai over Italian." **Summary (LLM)**

Table 8: Per-reader diagnostic results (CM, $B = 5$K). System ranking is consistent across all three readers.

Reader	System	CSM	RM	Δ_{w}	Δ_{r}
GPT-5.2	Verbatim	0.23	0.19	0.32	0.04
	Summ. (E)	0.27	0.23	0.28	0.04
	MemBank	0.31	0.26	0.24	0.05
	MemWlk.	0.37	0.28	0.18	0.09
	Summ. (L)	0.46	0.40	0.09	0.06
	ReadAg.	0.45	0.37	0.10	0.08
	EPC	0.51	0.44	0.04	0.07
Claude S4	Verbatim	0.21	0.17	0.29	0.04
	Summ. (E)	0.25	0.21	0.25	0.04
	MemBank	0.28	0.24	0.22	0.04
	MemWlk.	0.33	0.26	0.17	0.07
	Summ. (L)	0.42	0.36	0.08	0.06
	ReadAg.	0.41	0.33	0.09	0.08
	EPC	0.47	0.40	0.03	0.07
Gemini 2.5	Verbatim	0.22	0.18	0.31	0.04
	Summ. (E)	0.26	0.22	0.27	0.04
	MemBank	0.30	0.25	0.23	0.05
	MemWlk.	0.35	0.27	0.18	0.08
	Summ. (L)	0.44	0.38	0.09	0.06
	ReadAg.	0.43	0.35	0.10	0.08
	EPC	0.49	0.42	0.04	0.07

CSM: The summary states "the user discussed food preferences" without specifying the preference direction. Reader answers incorrectly. **EPC CSM:** A probe question "What food does the user prefer?" preserves `[E] User prefers Thai over Italian` → reader answers correctly.

Case 3: Lost negation. **Question:** "Is the user still taking the advanced Python course?" **Gold evidence:** `session_22, turn_8`: "I dropped the advanced Python course last week." **Summary (LLM) CSM:** The summary mentions "the user is taking programming courses"—the negation (dropping the course) is lost. Reader answers "Yes." **EPC CSM:** A probe question "What courses is the user currently enrolled in?" preserves `[E] Dropped advanced Python course last week` → reader answers correctly.

These cases illustrate the pattern behind Δ_{write}: query-agnostic summarization tends to preserve topics but omit the specific facts (dates, preference directions, negations) that questions target. EPC's self-questioning improves coverage of these high-value details, explaining its lower Δ_{write}.

A.7 Full Budget Sweep Results

Table 9 reports the full budget sweep results for the four systems included in the sweep.

A.8 Protocol Validation via Controlled Degradation

As a validation check, we evaluate Summary (LLM) under five controlled settings: the unmodified system, mild and severe write-side degradation (randomly dropping 25% or 50% of memory entries after the write stage), and mild and severe retrieval-side degradation (replacing top-ranked retrieved entries with entries from the bottom-ranked 25% or 50% of the ranked candidate list). We run each setting with all three readers, yielding 15 setting–reader runs. Figure 5 shows that Δ_{w} responds selectively to write-side degradation and Δ_{r} to retrieval-side degradation; the degraded variants are diagnosed correctly.

Table 9: Full write-budget sweep (CM, 3-reader avg; retrieval budget fixed at 5K). Refs: TFC= .18, OE= .53. W/R denote write-/retrieval-dominant diagnoses. Gaps are computed from unrounded per-reader scores; rounding may cause ± 0.01 discrepancies with the rounded CSM column. Verbatim and Summary (Extractive) remain write-dominant at all budgets.

Budget	System	CSM	RM	Δ_w	Δ_r	Diag.
2K	Verbatim	0.14	0.11	0.40	0.03	W
	Summ. (E)	0.20	0.15	0.34	0.05	W
	Summ. (L)	0.33	0.28	0.21	0.05	W
	EPC	0.44	0.36	0.10	0.08	W
5K	Verbatim	0.22	0.18	0.31	0.04	W
	Summ. (E)	0.26	0.22	0.27	0.04	W
	Summ. (L)	0.44	0.38	0.09	0.06	W
	EPC	0.49	0.42	0.04	0.07	R
10K	Verbatim	0.32	0.26	0.22	0.06	W
	Summ. (E)	0.34	0.28	0.20	0.06	W
	Summ. (L)	0.50	0.42	0.04	0.08	R
	EPC	0.52	0.44	0.02	0.08	R
20K	Verbatim	0.37	0.31	0.17	0.06	W
	Summ. (E)	0.42	0.35	0.12	0.07	W
	Summ. (L)	0.52	0.44	0.02	0.08	R
	EPC	0.53	0.46	0.01	0.07	R

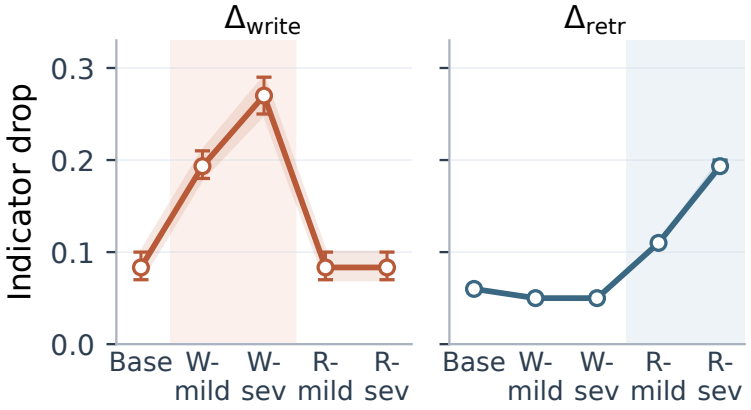

Figure 5: Controlled degradation (3 readers). Δ_w (write-side gap = OE − CSM) responds selectively to write-side degradation; Δ_r (retrieval-side gap = CSM − RM) responds selectively to retrieval-side degradation.

A.9 Bootstrap Significance Testing

For significance testing, we use paired bootstrap resampling over questions. Each bootstrap sample resamples the 500 LongMemEval questions with replacement and computes the CSM difference between EPC and Summary (LLM) on the resampled set. We use 10,000 bootstrap samples. The reported p-value is the fraction of samples in which the paired EPC–Summary (LLM) difference is less than or equal to zero. Confidence intervals are percentile intervals over bootstrap samples.

B Additional Analyses and Limitations

B.1 Write-Budget Sweep Summary

Table 10 summarizes the Summary (LLM) and EPC operating points discussed in Section 6.8. Table 9 includes all four systems in the sweep.

Table 10: Write-budget sweep for Summary (LLM) vs. EPC (CM, 3-reader avg; retrieval budget fixed at 5K). Refs: TFC= .18, OE= .53. W/R denote write-/retrieval-dominant diagnoses. EPC's advantage shrinks from +.11 (2K) to +.01 (20K). Gaps are computed from unrounded per-reader scores; rounding may cause ± 0.01 discrepancies with the rounded CSM column. Full results for the budget-sweep systems are in Table 9.

Budget	System	CSM	Δ_w	Δ_r	Diag.
2K	Summ. (L)	0.33	0.21	0.05	W
	EPC	0.44	0.10	0.08	W
5K	Summ. (L)	0.44	0.09	0.06	W
	EPC	0.49	0.04	0.07	R
10K	Summ. (L)	0.50	0.04	0.08	R
	EPC	0.52	0.02	0.08	R
20K	Summ. (L)	0.52	0.02	0.08	R
	EPC	0.53	0.01	0.07	R

B.2 EPC Breakdown: Question Type and Probe Alignment

We next ask where EPC helps most: across question types, and as a function of how closely its probe questions match the held-out test question.

Figure 6 shows that EPC improves CSM on single-session, multi-session, and temporal questions, with the largest gain on single-session questions (+.06). We measure probe–question alignment as the maximum cosine similarity between the test question and EPC's generated probe questions, using the same `all-MiniLM-L6-v2` embeddings as retrieval. Splitting questions at the median alignment score shows the same pattern: +.10 CSM when probe questions align and +.04 when they do not. One explanation is that self-questioning encourages broader coverage of high-value information types (entities, dates, preferences), which can benefit even unanticipated questions.

B.3 Component Ablation and Weight Sensitivity

We then isolate which EPC components account for the gain and check sensitivity to probe count and utility weights.

Figure 7 identifies self-questioning as the largest contributor among the tested components: removing it drops CSM by .05, while random probe questions recover only +.01 over Summary (LLM). $k = 5$ probe questions suffice; $k = 10$ provides no additional benefit.

Utility weight sensitivity. The utility function (Equation 5) uses weights $\alpha = 1.0$, $\beta = 0.5$, $\lambda = 0.3$ selected in a 20-example pilot. Across six alternative configurations, CSM stays within

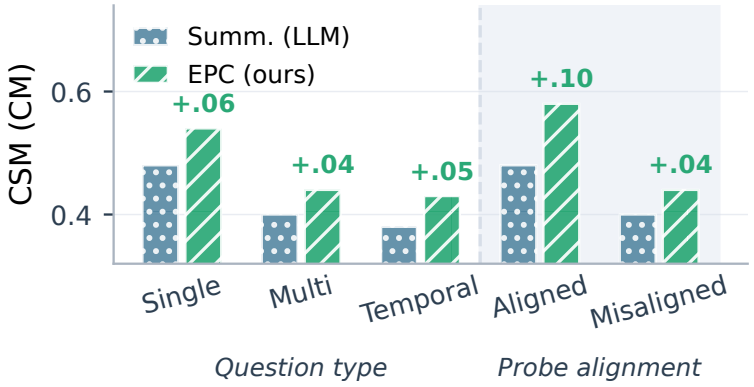

Figure 6: EPC vs. Summary (LLM) CSM (CM, 3-reader avg, $B = 5$K) by question type (left) and probe–question alignment (right). EPC gains are shown above bars; the largest gain (+.10) occurs when probe questions align with test questions.

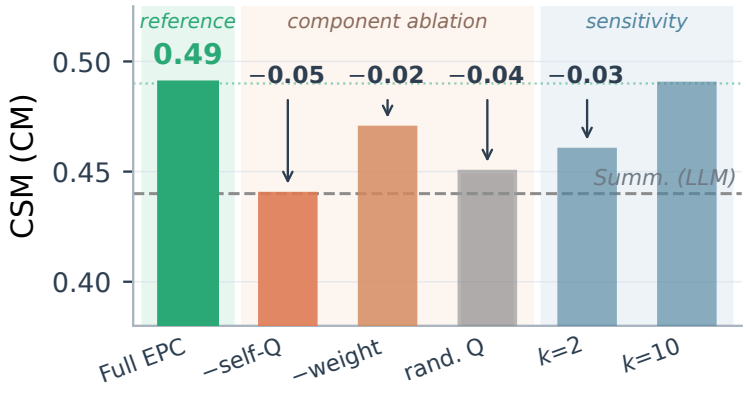

Figure 7: Component ablations and probe-count sensitivity (CSM CM, $B = 5$K). Removing self-questioning ($-.05$) is the largest drop; utility-weight sensitivity is reported below.

.45–.50; reducing specificity weight β hurts most ($-.04$), but all variants match or exceed Summary (LLM), suggesting that the self-questioning mechanism contributes more to EPC's gains than the particular weight settings. This low sensitivity to weight configurations suggests that the reported gains are not driven by this small pilot choice.

B.4 LIMITATIONS

The diagnostic gaps are conditioned on the chosen reader and metric. EPC's self-questioning step adds write-time latency. While we evaluate on two benchmarks (LongMemEval and LoCoMo), both rely on gold evidence annotations for the OE condition. This is an experimental choice rather than a protocol limitation: the four-condition structure is agnostic to how evidence is obtained. In practice, a practitioner could approximate OE with silver annotations—e.g., prompting an LLM to identify supporting turns for each question—which would introduce annotation noise but may preserve the directional diagnosis (write-dominant vs. retrieval-dominant), since the diagnosis rule already tolerates small perturbations via the margin ϵ. Empirically validating this silver-annotation path remains future work. LoCoMo's shorter context ($\sim$9K tokens) means the compression challenge is milder than in LongMemEval; evaluating on benchmarks with longer contexts and more diverse domains would further strengthen generalizability claims.