

Evidence in Context Is Not Enough: Why Small Language Models Fail at Long-Term Conversational Memory

[OPERATOR NAME]

Archilas · archilas.com

Preprint, October 2026. Not peer reviewed.

Keywords: long-term conversational memory, agent memory, small language models, LoCoMo, error analysis, retrieval-augmented generation, negative results, pre-registration

Abstract

Memory layers for conversational agents are usually evaluated end to end, with large readers, and with a single accuracy number. That hides *where* a pipeline fails. We study a retrieval-based memory agent whose reader is a 4-billion-parameter instruction-tuned model (Qwen3-4B-Instruct-2507) answering from stored conversation notes under an interactive latency constraint. On a 233-question non-adversarial split of LoCoMo (conversations conv-26 and conv-30), our frozen baseline answers 160/233 (68.7%) correctly, judged by an LLM; on a disjoint 233-question sample from eight other conversations it answers 168/233 (72.1%). We classify all 73 errors on the first split by pipeline stage. 22 are retrieval-stage failures: the evidence never reached the prompt, either because it ranked too low (14) or because it was ranked but evicted by the context budget (8). 49 are answer-stage failures, and 2 were labelled gold or judge errors. In **all 49 answer-stage failures the gold evidence session was in the prompt**. No answer was truncated, and neither context length nor evidence position separates failures from successes at our roughly 9k-token context length. The dominant answer-stage modes are abstaining or hedging while the fact is present, and committing to a near-duplicate "sibling" event from another session. We then report a set of interventions evaluated against gates written and hashed before measurement, with gold labels used only for scoring and no post-hoc rescue. None beat the baseline under its gate: parameter-efficient fine-tuning (−18 questions), quote-before-answer prompting (−7), a gated self-correction retry (net 0), an anti-abstention instruction (8 wins, 12 losses), LLM-rewritten notes, retrieval re-scoring and diversity methods, deterministic note annotation, a fact-log representation, and turn-level packing. Second-pass retrieval gained +11 on the development split but did not hold on held-out conversations. Two larger readers were ruled out on speed alone before any accuracy test. The only large gain we observed came from changing the *representation* of the notes, not the reader or the prompt: deterministic resolution of relative dates moved held-out accuracy from 136/233 to 150/233. We argue that small-model memory is limited by two things. One is retrieval depth under a fixed raw-context budget. The other is evidence *conversion*: whether the reader uses evidence that is already in its prompt. Offline coverage proxies can measure the first but not the second, and in our data they over-predicted gains. We report noise explicitly: a single prompt line changed 20 of 233 answers, so we treat differences of about ± 5 questions on our development split as noise.

1. Introduction

Long-term memory for conversational agents is now an active research and engineering area. Systems such as MemGPT/Letta [Packer et al., 2023], Mem0 [Chhikara et al., 2025], Zep [Rasmussen et al., 2025], A-MEM [Xu et al., 2025] and others extract, store and retrieve information from long interaction histories, and are commonly compared on benchmarks such as LoCoMo [Maharana et al., 2024] and LongMemEval [Wu et al., 2024]. Most published results use large, often proprietary readers, and report one aggregate score per system.

Small open models (≤ 4 B parameters) are attractive as memory readers. They are cheap to self-host, can run inside a user's own infrastructure, and are fast enough to sit inside an agent's tool-call loop. However, end-to-end scores say

little about *why* such a reader fails. A wrong answer can come from evidence that was never retrieved, from evidence that was retrieved but cut by a context budget, or from evidence that was in front of the model and not used. These failures call for different remedies, and an aggregate score cannot tell them apart.

This paper is an error analysis and a structured report of negative results for a small-model memory pipeline on LoCoMo. We ask three research questions:

- **RQ1 (Where).** When a small-reader memory pipeline answers a long-term conversational memory question wrongly, at which stage does the failure occur: retrieval (the evidence is absent from the prompt) or answering (the evidence is present and the answer is still wrong)?
- **RQ2 (Why).** For answer-stage failures, what distinguishes them from successes? Are they explained by context length, evidence position, truncation, or by properties of the surrounding evidence?
- **RQ3 (What does not help, and why).** Which common interventions (fine-tuning, prompting, self-correction, retrieval re-scoring, note rewriting and annotation, alternative memory units, larger readers) fail to improve such a pipeline under pre-registered gates, and what general lessons follow?

Contributions.

1. **A stage-level error taxonomy** covering every error of a working small-reader pipeline on a LoCoMo development split. It separates retrieval-stage failures (ranked out vs. evicted by the budget) from answer-stage failures, and it shows that in all 49 answer-stage failures the evidence session was in the prompt (Section 5.2–5.4).
2. **A methodology for low-cost, low-bias iteration:** offline gates written and cryptographically hashed before any measurement, CPU-only replays of stored pipeline state, gold labels used only for scoring, and no post-hoc rescue of failed gates. We also report where this method's offline proxies were miscalibrated (Section 4).
3. **Generalizable findings from negative results.** Interventions that did not help are stated with outcome numbers and with the mechanism we believe explains each failure, then distilled into general lessons for small-model memory (Sections 5.5 and 6).
4. **An explicit noise accounting** for a small development split, including a measured instance of one prompt line changing 20 answers (Section 5.1).

We make **no state-of-the-art claim**. Our numbers come from one LLM judge, small evaluation splits and one serving configuration. They are not comparable to published LoCoMo numbers obtained with different readers, judges and protocols (Section 7).

2. Related Work

Memory systems for LLM agents. MemGPT [Packer et al., 2023] frames context management as an operating-system-style hierarchy in which the model pages information between a bounded main context and external storage. Mem0 [Chhikara et al., 2025] extracts salient facts from each message pair with an LLM and applies ADD/UPDATE/DELETE/NOOP operations, with an optional graph variant. Zep [Rasmussen et al., 2025] builds a temporally aware knowledge graph (Graphiti) over conversational and business data. A-MEM [Xu et al., 2025] organizes memories as interlinked, LLM-annotated notes. MemoryOS [Kang et al., 2025] uses a hierarchical short/mid/long-term store. Memory-R1 [Yan et al., 2025] trains memory-management and answer agents with reinforcement learning. Hindsight [Latimer et al., 2025] organizes memory into separate logical networks so that world facts are kept apart from the agent's own inferences. PREMem [Kim et al., 2025] moves reasoning to storage time ("pre-storage reasoning"). Of these, PREMem is the most relevant to our setting because it reports small readers. With Qwen2.5-3B as the response model, PREMem reports an LLM-judge score of 53.8 on LoCoMo, against 44.90 for turn-level retrieval and 42.56 for full context with the same reader (authors' numbers, their judge). TRemMu [Ge et al., 2025] targets temporal reasoning in multi-session dialogue: it summarizes sessions with inferred dates and has the

LLM generate code for temporal calculations. Our only large positive result, deterministic date resolution in notes, fits this line of work in which structure is added at ingest and arithmetic is moved out of the reader.

Benchmarks and evaluation validity. LoCoMo [Maharana et al., 2024] contains very long-term, multi-session dialogues between two personas, with question answering over single-hop, multi-hop, temporal, open-domain and adversarial (unanswerable) categories. Its authors also note that retrieval over "observations" (assertions about each speaker) works well. LongMemEval [Wu et al., 2024] evaluates information extraction, multi-session reasoning, temporal reasoning, knowledge updates and abstention. LoCoMo scores are contested. The Mem0 paper reports, with GPT-4o-mini and its own LLM-judge protocol (adversarial category excluded), 66.88% for Mem0, 68.44% for its graph variant, 65.99% for Zep and 72.90% for a full-context baseline [Chhikara et al., 2025]. Zep's developers disputed the Zep configuration used there and published a corrected self-evaluation of $75.14\% \pm 0.17$ [Zep, 2025]. A Mem0 author, in a public reply, argued for $58.44\% \pm 0.20$ under a corrected evaluation [Mem0 / Zep public discussion, 2025]. Letta reports 74.0% on LoCoMo with GPT-4o-mini by placing conversation histories in files and giving the agent file-search tools [Letta, 2025]. A third-party audit of LoCoMo (not peer reviewed) reports that 99 of 1,540 scored questions (6.4%) have wrong gold answers, implying a ceiling of about 93.6%. The same audit reports that a GPT-4o-mini judge with a lenient instruction accepted 62.81% of deliberately vague but on-topic wrong answers [locomo-audit, n.d.]. We use a different judge, so these rates do not transfer directly. We do take from this literature that LoCoMo numbers from different protocols should not be ranked against each other.

Long context, position and distraction. Language models can fail to use information in the middle of long inputs [Liu et al., 2023]. Effective context is often far shorter than advertised [Hsieh et al., 2024], and performance degrades when the question and the evidence share little lexical overlap [Modarressi et al., 2025]. Irrelevant context distracts models on reasoning problems [Shi et al., 2023]. In retrieval-augmented generation, highly ranked but non-answering passages hurt accuracy [Cuconasu et al., 2024]. An industry report describes degradation as input length and distractors grow [Hong et al., 2025]. Training on noisy retrieval [Yoran et al., 2023] and note-based reading [Yu et al., 2023] improve robustness, but both need training or extra decoding. Our answer-stage findings agree with the distraction results. However, we observe no position or length effect at our much shorter context length.

Self-correction. Without external feedback, LLMs often fail to improve their own answers and can make them worse [Huang et al., 2024]. Our gated retry (Section 5.5) is consistent with this.

Caching and serving. Paged KV-cache management and prefix caching [Kwon et al., 2023] make a repeated prompt prefix cheap. Cache-augmented generation preloads a whole knowledge source into a cached context [Chan et al., 2025]. CacheBlend [Yao et al., 2025] addresses reusing caches of non-prefix chunks. These works explain one architectural tension we observed: a memory prompt assembled per query defeats prefix reuse, and any change to model weights (for example an adapter) invalidates cached prefixes.

3. Setting and Problem Setup

3.1 Setting

The research system is the answering core of a memory/helper agent. The agent stores notes about what other agents and people were told and did, and answers questions about that history from those notes using a small language model. For this study the system is a conventional retrieve-then-read pipeline. Conversations are stored as notes, with one note per conversation session, kept verbatim with speaker labels and deterministic relative-date anchors. A non-LLM hybrid lexical-plus-dense retriever ranks sessions for each question. Whole sessions are packed into a fixed token budget in rank order (sessions are skipped, never truncated), which gives a mean context of about 8.9k tokens. The memory text precedes the question in the prompt, so that the memory part can in principle be served from a prefix cache. The reader is Qwen3-4B-Instruct-2507 [Yang et al., 2025] in bf16 with greedy decoding, no adapter, and a fixed answer-length cap. All experiments run under an interactive latency constraint. We call this frozen configuration the **baseline**.

3.2 Data

We use LoCoMo [Maharana et al., 2024].

- **Development ("smoke") split:** conversations conv-26 and conv-30, 304 questions, of which 233 are non-adversarial and 71 are adversarial (no answer exists; the correct behaviour is to abstain). These two conversations have 19 sessions each. All error analysis in Section 5 is on this split.
- **Fresh sample:** 300 questions from eight other conversations (conv-41, 42, 43, 44, 47, 48, 49, 50), 233 non-adversarial and 67 adversarial, with no overlap with the development split.
- **Original held-out sample:** an earlier 300-question sample (233 non-adversarial, 67 adversarial), used for the date-representation result.

Most interventions were run only on the development split (Section 7).

3.3 Metrics

- **Accuracy:** the fraction of the 233 non-adversarial questions that the LLM judge marks correct. The judge is Claude Sonnet 4.6, run blind at temperature 0 with one fixed rubric.
- **Adversarial abstention:** the number of the adversarial questions on which the system correctly abstains. Its complement, the **confident-wrong count**, is the number of unanswerable questions that receive a substantive answer.
- **Wins / losses / net:** paired against the baseline per question. A win is baseline-wrong and variant-correct; a loss is baseline-correct and variant-wrong.
- **Coverage (offline):** whether the gold evidence (session-level, or turn-level where stated) is present in the packed context. Coverage is not accuracy.

3.4 Error taxonomy

Each non-adversarial error is assigned to exactly one stage, using the stored per-question retrieval rankings, the set of loaded sessions and LoCoMo's gold evidence identifiers.

- **Retrieval stage, ranked out:** at least one gold evidence session was ranked outside the candidate pool considered for packing.
- **Retrieval stage, packing:** all missing gold sessions were ranked inside the candidate pool but were not loaded because the token budget was exhausted by higher-ranked sessions.
- **Answer stage:** all gold evidence sessions were loaded and the answer was judged wrong.
- **Gold/judge:** the error is attributable to the gold label or to judge strictness (labelled conservatively; see Section 5.4).

Answer-stage errors are further assigned a primary cause by precedence rules frozen before classification: truncation, then loose gold or judge strictness, then false abstention or hedging, then ambiguous question, then distraction (the answer value comes from another session), then "unclear" (the evidence is present but no rule fires). Independent co-occurrence tags are recorded alongside the primary cause.

4. Method: Pre-Registered Offline Gating

Small development splits and noisy LLM judges make it easy to fool oneself. We used the following protocol for every analysis and intervention reported here.

1. **Write the gate first.** Before any result is computed, the metrics, the pass criteria and what counts as a regression are written to a rules file. The file is hashed (SHA-256) and time-stamped, and the hash is re-

checked after the run. Gates typically combine a minimum gain on the target errors, a maximum number of regressions on previously correct questions, zero loss of adversarial abstentions, and a cost or latency condition.

2. **Replay on stored state.** The baseline run stored, for every question, the retrieval ranking, the loaded sessions, the context size, the answer and the finish reason. Together with the note text, this lets retrieval and packing changes be replayed exactly on CPU. As a fidelity check, re-packing the stored rankings with recovered session sizes reproduced the stored loaded set in 301 of 304 rows.
3. **Use gold only for scoring.** Design choices are made from question text and structure, never from gold answers or evidence identifiers. Code is checked for corpus-specific names and cue words.
4. **No post-hoc rescue.** A gate that fails by one item fails. A revised design needs a new frozen gate, and it is labelled post-hoc whenever the revision was informed by the failed run.
5. **Disclose contamination.** Where an analyst had seen earlier per-question labels before freezing rules, we say so, and we do not present the result as a blind second rating.
6. **A pass licenses a model run; it predicts nothing.** Offline proxies measure coverage, while the dominant failure is conversion. In our data an offline replay predicted +2 to +3 for a retry mechanism and the real run gave 0 (Section 5.5). We therefore discount any offline estimate heavily before deciding to run a model.

The protocol's value lay mostly in what it stopped. Five offline studies, covering more than a dozen variants, were parked on CPU without any model inference. Its limit is equally clear. Offline replays measure *coverage* (is the gold text in the prompt?), but our main failure mode is *conversion* (does the reader use it?), and conversion can only be measured by running the reader.

5. Experiments and Results

5.1 Baselines and noise

Table 1. Baseline accuracy. Accuracy is on the 233 non-adversarial questions of each split. The 95% Wilson intervals are computed by us from the counts.

Split	Accuracy	95% CI	Adversarial abstention
Development (conv-26/30)	160/233 (68.7%)	62.5–74.3%	36/71
Fresh sample (8 other conversations)	168/233 (72.1%)	66.0–77.5%	49/67
Original held-out sample	158/233 (67.8%)	61.6–73.5%	45/67

The two headline figures come from **different question samples**: 160/233 = 68.7% on the development split and 168/233 = 72.1% on the fresh sample. They should not be merged or compared as a before/after. Development accuracy by LoCoMo category is single-hop 93/114 (82%), temporal 51/63 (81%), multi-hop 14/43 (33%) and open-domain 2/13 (15%). Multi-hop questions account for 29 of the 73 development errors (15 retrieval-stage, 14 answer-stage).

Noise. There is no repeated run of the baseline on the development split, so its run-to-run spread is unmeasured. The best evidence we have is that adding a single instruction line to the prompt changed 20 of 233 answers (8 wins, 12 losses; exact two-sided sign test $p \approx 0.50$). On a separate small private set, re-running one configuration on different hardware changed one answer out of 48. Sampling uncertainty of the split itself is wider still (Table 1). **We therefore treat differences of about ± 5 questions on the development split as noise** and make no claim based on smaller differences. Paired comparisons (wins/losses) are reported throughout because they are more informative than marginal accuracies.

The one large positive result: representation. Before the baseline was frozen, adding deterministic, rule-based relative-date anchors to notes (for example, resolving "yesterday" to the session-relative calendar date) moved

accuracy on the original held-out sample from **136/233 (58.4%) to 150/233 (64.4%)** ($p = 0.020$ as recorded in our log). On the relative-date subset, accuracy went from 7/40 to 26/40, while note length grew by only about 1%. The full baseline later scored 158/233 on the same sample. Note that "58% $\rightarrow$ 72%" would compare 136/233 on the original held-out sample with 168/233 on the fresh sample. These are different question sets, so we do not report that as a gain. The improvement came from changing *what the notes say*, not from the reader or the prompt.

5.2 RQ1: Errors by stage

Table 2. Development-split errors (73 of 233) by pipeline stage.

Stage	n	Evidence
Retrieval: ranked out	14	a gold session ranked outside the packing pool (11 with partial evidence loaded, 3 with none)
Retrieval: packing	8	gold session ranked in the pool but evicted by the budget (5 partial, 3 with none)
Answer stage	49	gold evidence session loaded in 49/49
Gold / judge (labelled)	2	plausible true share about 5–14 (Section 5.4)
Total	73	

So **two thirds of errors (49/73) occur with the evidence already in the prompt**. Measured another way, the reader is wrong on 49 of the 211 questions whose gold sessions were all loaded, about 23% (derived).

5.3 Retrieval-stage analysis (22 errors)

Across the 22 retrieval-stage questions, 26 gold sessions were not loaded. 16 were ranked outside the pool and 10 were inside the pool but evicted. **No gold session among the top 8 ranks was ever evicted**; every evicted gold session sat in the lowest-ranked part of the pool. In practice, the budget admits about 8–9 whole sessions of the ~ 19 per conversation, so roughly half of each conversation is never shown to the reader. When we recomputed exact ranks offline, **17 of the 26 missing gold sessions ranked 13th–19th of 19**. Seven of the 22 errors would not be fixed by retrieval at all: 5 are judge-strict or loose-gold cases and 2 are inference questions. Of the remaining 15 retrievable errors, **11 have their missing gold session in that bottom rank band**, and no reordering inside the pool can reach them.

The structural reason is that LoCoMo conversations are two-person dialogues. Every session shares the same entities and vocabulary, so the missing sessions are semantically similar to the loaded ones and are not obvious outliers that a better scorer would surface.

5.4 RQ2: Answer-stage analysis (49 errors)

All 49 answer-stage errors were classified by rules frozen and hashed before any classification output. One disclosure applies: the analyst had seen earlier hand labels for some of these items, so this is not a blind second rating.

The evidence is present. The gold evidence session was loaded in 49/49. Token-level gold recall in the evidence text was at least 0.5 for 45; the other 4 are inference questions whose gold is a judgment rather than a span. In a manual spot check of 10 items selected systematically (every fifth item), the gold fact appeared literally (3) or in equivalent form (7) in the packed note text in 10/10.

No truncation. All 49 answers ended with a normal stop. The longest was 94 tokens, well below the answer cap.

No length or position effect at about 9k tokens. The mean context was 8,937 tokens for errors and 8,921 for the 160 correct answers. In a reconstruction of prompt order (the exact prompt string was not stored), the gold session fell in the first third of the context for 36 of 49 errors, against 138 of 160 correct answers (73% vs. 86%). A separate analysis of gold-session midpoints found error shares of 26% / 25% / 32% for start, middle and end positions, which is no lost-in-the-middle pattern. Neither comparison was significance-tested, and both are descriptive.

Table 3. Primary causes of the 49 answer-stage errors (frozen precedence rules).

Cause	n
False abstention / hedge with the fact present	15 (19 counting disguised abstentions such as "did not specify")
On-topic but wrong, partial or over-general (rules could not separate further)	18
Loose gold / judge strictness (rule-assigned)	9 (about 3 defensible on manual audit)
Distraction: answer value taken from another session	5 (7 by a broader tag)
Ambiguous question	2
Truncation	0

The 15 false abstentions include 7 inference questions ("would X likely..."), all answered "I don't know". The rule for loose gold or judge strictness over-assigns: a manual reading of judge reasons supports only about 3 of its 9 assignments. The others are aggregation, date or contradiction failures. We report both figures and did not move items after freezing.

Co-occurring properties. These are tags on the 49 errors; we have no matching baseline rate on correct answers, so they are not causal claims. In 26/49, another line in the context fits the question at least as well as the gold turn. In 22/49, the conversation has a near-duplicate sibling question about the same session. In 18/49, the gold is split across turns or sessions. A second analysis with a broader definition of distraction marked 24/49. Recurring patterns include:

- sibling events of the same type in different sessions (several camping trips, several paintings);
- a referent that appears only in an image caption or as "this/it";
- a date attached to a neighbouring clause;
- a speaker or owner mismatch between the question and the note line;
- durations that must be computed rather than read.

Interpretation. The dominant answer-stage failure is a small reader that **does not commit to evidence it has been given**. It either abstains or hedges, or it commits to a plausible competitor from a similar event. This agrees with distractor-sensitivity results [Shi et al., 2023; Cuconasu et al., 2024], which were obtained on other tasks and mostly larger models. It does not match position or length effects reported for much longer contexts [Liu et al., 2023; Hsieh et al., 2024; Modarressi et al., 2025], which we do not observe at about 9k tokens.

The benchmark ceiling does not bind. Combining the 2 labelled gold/judge errors with the judge-strict or loose-gold cases found among the retrieval-stage items and the defensible answer-stage cases, we estimate that only about 5–14 of the 73 errors trace to gold or judge problems. Even with the third-party estimate of about 6.4% wrong gold answers [locomot-audit, n.d.], the ceiling lies far above 160/233.

Adversarial questions. The baseline abstains correctly on 36 of 71 adversarial questions and gives an answer to the other 35. For a memory agent whose answers other agents act on, this confident-wrong count is arguably as important as accuracy. For that reason every intervention gate required zero loss of adversarial abstentions.

5.5 RQ3: Interventions that did not help

Each intervention is compared against the baseline (development split: 160/233, adversarial abstention 36/71). Interventions are described at the level of the general technique. "Online" means the reader was run and judged. "Offline" means a CPU-only replay against a frozen gate, with no reader inference.

(a) Parameter-efficient fine-tuning of the reader (online). Accuracy fell to 142/233 (−18), while adversarial abstention rose to 47/71. Answers became shorter (mean 27 vs. 39 tokens). *Why it failed:* the adapter made the model terser and more inclined to abstain. That helped on unanswerable questions but cost far more on answerable ones. It

also conflicts with prefix caching, because any weight change invalidates every cached memory prefix, and these had to be recomputed.

(b) Second-pass retrieval (online). Development accuracy rose to **171/233 (+11; $p = 0.027$ as recorded)**, with adversarial abstention unchanged. However, it loaded more context, which pushed tail latency close to the interactive limit. On a partial held-out run (176 questions) it scored 123 against the baseline's 126. *Why it failed:* more depth helps where the evidence is deep (Section 5.3), but at a fixed latency it can only be bought with more prefill, and the development gain did not generalize to other conversations. This is our clearest example of a significant-looking development win that was split-specific.

(c) Quote-before-answer prompting (online). The reader was asked to quote supporting evidence before answering. Accuracy fell to 153/233 (14 wins, 21 losses; $p \approx 0.31$, computed by us). Adversarial abstention fell to 32/71, and latency was at the limit on multi-session questions. *Why it failed:* the format fixed 12 of the 49 answer-stage errors but broke more questions that the plain prompt already answered. 17 of the 21 losses came from the always-on first pass, not from retries. Always-on format pressure on a small reader is costly.

(d) LLM-rewritten notes (online, small private set). Rebuilding notes as LLM-selected topic summaries scored 33/48 against 35/48 for verbatim notes on a separate private 48-question set. *Why it failed:* LLM selection is lossy. On the private corpus, only 141 of 343 quantity-bearing sentences (41%) survived verbatim, and nothing in the pipeline detects the loss. Across our work, LLM-rewritten notes never beat verbatim notes.

(e) Anti-abstention instruction (online). An always-on instruction to answer when related evidence exists gave 156/233 (8 wins, 12 losses), and adversarial abstention moved to 44/71. A conditional re-ask after a bare abstention fired on 18 answerable questions and converted 0 of them (160/233; adversarial 35/71). *Why it failed:* the small reader keeps abstaining even when re-asked, and a single instruction reshuffles about 20 answers in both directions. This result is the source of our noise estimate.

(f) Aggregation templates and map-then-compose (online). A list-enumeration template gained +2 on multi-hop questions only and increased multi-session latency. A map-then-compose variant (parallel extraction followed by a composition pass) exceeded the latency constraint. *Why it failed:* any extra decoding or extra model pass consumes the latency headroom on exactly the multi-event questions that need it.

(g) Gated self-correction retry (online). If the first answer was an abstention or hedge, a quote-first retry ran, and its answer replaced the original only if a deterministic verbatim-evidence check passed. Accuracy stayed at **160/233 (net 0)**, and adversarial abstention fell to 35/71 or 34/71 depending on the trigger variant, both below the floor. Only 2 retries passed the check, and both were adversarial false accepts. No non-adversarial answer changed. *Why it failed:* the check was too strict to admit true fixes (offline, it passes only 68 of the 160 *correct* baseline answers) and too loose to block two wrong ones. **The offline proxy had predicted +2 to +3; the real run gave 0.** This matches the literature on self-correction without external feedback [Huang et al., 2024].

(h) Retrieval re-scoring and diversity (offline, 10 variants). We tested three pool-level fixes: maximal-marginal-relevance diversification, best-turn scoring with reciprocal-rank fusion, and density-based budget spreading. They made 0, 1 and 0 of the 15 retrievable errors gold-complete, and two of them changed the loaded set for about 100 of the 160 correct answers. Seven ranker changes (rank fusion, turn-window scoring, entity/numeral expansion, two alternative embedding models, and combinations) moved between 1 and 4 of the 13 deepest missing gold sessions into the pool. The stronger variants reduced top-1 agreement with the baseline to as low as 0.36, and up to 8 correct answers lost a gold session. None passed. *Why it failed:* the problem is structural. In a two-person conversation, the missing sessions sit low among semantically similar sessions, and scoring changes cannot reliably promote them without displacing evidence for questions that are already correct. Different variants rescued different questions, so no combination did better.

(i) Deterministic note annotation (offline). We added additions-only annotations (a date on every turn, speaker and date on continuation clauses, image-referent links for demonstratives, and second-person resolution). They were byte-

exact removable (114/114 renders). The annotation mechanism matched only about 7 of the 49 answer-stage errors, while every one of the 160 correct prompts changed and prompts grew by about 35–43% on average. The image-referent rule was wrong on 25 of 65 manual checks (38%). *Why it failed*: the token and latency cost is certain and the benefit is small and speculative. It was not run online.

(j) Fact log as memory (offline stand-in). At ingest, one writes an append-only log of dated, speaker-attributed atomic facts and serves it as a static, fully cacheable prefix, the common design in fact-extraction memory systems. As a stand-in we used LoCoMo's own per-session observation facts. All four frozen gates failed. The log was about 58% larger than the size cap. Turn-level gold coverage was 163/233 (70.0%). Only 6 of the 15 retrievable errors were covered. 39 of the 160 correct answers would lose gold coverage. A text-level spot check found that even "covered" facts contained the answer value in only 10 of 24 cases, and relative dates lost their anchors. *Why it failed*: session-level coverage was nearly total (231/233), but turn-level coverage was not. A fact log can silently drop exactly the details questions ask about. A log built by the small reader itself could behave differently, but this stand-in gives that design no support.

(k) Turn-level packing (offline). Packing individual turns (with neighbours) instead of whole sessions, under the same budget, was the first retrieval-side change that substantially raised coverage. Turn-level coverage was 219/233 against the baseline's 207/233 by the same measure, and 16 of the 22 retrieval-stage errors (10 of 15 retrievable) became covered, with negligible added retrieval time. However, 4 correct answers lost a gold turn, one more than the pre-registered regression cap allowed, and the date check found 1 wrong resolution in 40 where the gate required zero. **Verdict: fail.** A second, stricter gate frozen independently before the result was seen also returned no-go. *Why it failed (so far)*: coverage is not accuracy. Every one of the 160 correct prompts would become a fragmented list of turns, and the answer-stage data (49/49 loaded and still wrong) says the small reader often does not convert loaded evidence. We did not loosen the gate after seeing the 4 regressions; a revised design would need a new gate and would be labelled post-hoc.

(l) Larger readers (speed only). The obvious remedy for a reader limit is a larger reader. Before any accuracy test, we checked whether two larger open models could meet the latency constraint on our single-GPU setup. A 30B mixture-of-experts model with about 3B active parameters, quantized to 8 bits, decoded at a median of 85.4 tokens/s, about 1.7× slower than the 4B baseline (derived). On our hardware the 8-bit path ran through weight-only kernels. An 8B dense model quantized to 4 bits decoded at a median of 154.5 tokens/s, roughly the 4B's speed, but its median time-to-first-token was about 40% higher (derived). Both failed pre-registered speed gates. *Why*: at batch size 1, a large share of per-token time is fixed overhead rather than memory bandwidth, so quantization buys less than expected, and weight-only quantization does not speed up prefill. Neither model was accuracy-tested. These are speed results only.

Table 4. Summary of interventions (development split unless stated).

Intervention (general technique)	Outcome vs. baseline	Verdict
Deterministic relative-date anchors in notes	136 → 150 / 233 (held-out)	Adopted (pre-baseline)
Parameter-efficient fine-tuning	142 vs. 160; adversarial 47 vs. 36; invalidates prefix cache	Rejected
Second-pass retrieval	171 vs. 160; held-out partial 123 vs. 126; latency near limit	Rejected
Quote-before-answer	153 (14 W / 21 L); adversarial 32	Rejected
LLM-rewritten notes	33 vs. 35 / 48 (private set)	Rejected
Anti-abstention instruction	156 (8 W / 12 L); re-ask: 0 of 18 converted	Rejected
Aggregation template / map-compose	+2 multi-hop only; latency over limit	Rejected
Gated self-correction retry	160 (net 0); adversarial 35 / 34	Rejected

Intervention (general technique)	Outcome vs. baseline	Verdict
Retrieval re-scoring and diversity (10 variants, offline)	0–3 retrievable errors completed; heavy reshuffling	Parked
Deterministic note annotation (offline)	~7 errors touchable; +35–43% prompt	Parked
Fact log, stand-in (offline)	4/4 gates failed	Parked
Turn-level packing (offline)	16/22 covered; 4 regressions (over cap)	Failed gate
Larger readers (speed only)	failed speed gates; accuracy untested	Rejected

6. Analysis: Generalizable Lessons

We state the lessons in general form. Each rests on the evidence cited, and each is limited by the caveats in Section 7.

L1. For small readers, "the evidence was retrieved" is not the end of the problem. Two thirds of our errors occurred with the evidence session in the prompt, at a context length (about 9k tokens) where neither position nor length explained them. Memory pipelines with small readers should report answer-stage error rates separately from retrieval recall. Improving retrieval recall alone will be bounded by conversion.

L2. Coverage and conversion are different quantities, and offline proxies measure only coverage. Turn-level packing raised coverage substantially (16 of 22 retrieval-stage errors covered), yet we have no evidence that the reader would convert it into correct answers. A retry mechanism predicted offline at +2 to +3 gave 0. Offline gating is valuable for *rejecting* ideas cheaply, but its positive predictions should be heavily discounted.

L3. Under a fixed raw-context budget, retrieval failure is a depth problem, not a ranking problem. In homogeneous corpora (one relationship, recurring entities), the missing evidence sits deep among similar candidates. Re-scoring, fusion, diversity and better embedders reshuffle the same few slots and trade one question's evidence for another's. Gains come from adding depth, which costs latency (b), or from making memory units smaller and denser, which changes every prompt (k).

L4. Small readers fail by not committing, and near-duplicate siblings make it worse. The main answer-stage modes are abstention or hedging with the fact present, and commitment to a similar event from another session. Prompt-level pushes against abstention (e), forced quoting (c) and self-checking (g) moved accuracy by −7 to 0 and reshuffled answers in both directions. On small readers, always-on format or instruction changes carry high regression risk, because they touch every question in order to fix a few.

L5. Representation changes at ingest beat reader-side changes, when they are deterministic and lossless. The only large gain came from deterministic date anchors that add information without removing any. LLM rewriting of notes (d) and fact extraction (j) silently lost answer-bearing detail. This suggests two practical rules. Structure should be added at ingest without discarding the source text. Any extraction step should come with a recall check against the source.

L6. Fine-tuning and caching pull in opposite directions. In an architecture whose latency depends on reusing cached memory prefixes, any adapter or weight change invalidates those caches. In our case fine-tuning also shifted the model toward terseness and abstention, which helped on unanswerable questions and hurt more on answerable ones (a). Designers should decide early whether a memory system is cache-first or adapter-first.

L7. A significant result on a small development split can be split-specific. Second-pass retrieval reached $p = 0.027$ on the development split and did not hold on held-out conversations. With one prompt line moving 20 of 233 answers, development splits of this size need held-out confirmation before any claim.

L8. Report the confident-wrong count. About half of our unanswerable questions received a substantive answer. Several interventions traded answerable accuracy against abstention (a, e). A single accuracy number hides that trade,

and for agents that act on memory answers it may be the more important number.

7. Limitations

- **Small evaluation.** The development split is two two-person conversations from LoCoMo. There is no repeated baseline run, so the run-to-run spread is unmeasured, and we rely on the ± 5 -question noise estimate. Most interventions were not run on the fresh or held-out samples.
 - **One judge, one rubric.** All accuracy figures use one LLM judge with one rubric. LoCoMo gold answers are disputed (Section 2). Our numbers are **not comparable** to published LoCoMo scores that use different readers, judges, question subsets and protocols, and we make no ranking claim.
 - **Proxy labels.** Answer-stage causes come from frozen rules over stored outputs and judge reasons, some of which (such as loose gold) we know over-assign. Co-occurrence tags have no baseline rate on correct answers. The exact prompt string was not stored, so evidence position is reconstructed. The analyst was not blind to earlier hand labels.
 - **Offline replays are approximations.** The recomputed retriever reproduced the stored ranking closely but not exactly, so offline exposure counts are relative to a recomputed control. Coverage is measured at session or turn level, not as verbatim answer presence.
 - **Untested.** We did not test whether the prefix-cache path itself changes answers (no cached vs. fresh-prefill pair was stored), the accuracy of any larger reader, a fact log built by the small reader, or turn-level packing with the reader in the loop.
 - **Domain.** LoCoMo is synthetic social chit-chat. Agent histories (tool calls, coding sessions, decisions) may differ in vocabulary diversity and in how much near-duplicate structure they contain, so the depth and sibling-confusion findings may not transfer directly.
 - **Single serving configuration and hardware class.** Speed results for larger readers depend on the hardware and kernel support available to us.
 - **Conflict of interest.** The author works on a commercial memory agent. To limit this bias, the study reports negative results, makes no comparative claim against other systems, and uses only public data for all LoCoMo results.
-

8. Conclusion

We examined why a retrieval-based conversational memory pipeline with a 4B reader fails on LoCoMo. Two thirds of its errors happen with the evidence already in the prompt, at a context length where position and length effects do not explain them. The remaining errors are mostly evidence ranked deep among similar sessions under a fixed context budget. A broad set of interventions failed under pre-registered gates: fine-tuning, prompting, self-correction, re-ranking, note rewriting and annotation, fact logs, and turn-level packing. Second-pass retrieval improved the development split but not held-out conversations. The one large gain came from a deterministic, lossless change in note representation. We conclude that small-model memory is bounded by retrieval depth and by evidence conversion, and that offline proxies can measure only the first. We hope the stage-level taxonomy, the gating protocol and the explicit negative results help others avoid the same dead ends. Promising directions include memory prefixes that are static per scope and therefore fully cacheable, with ingest-time recall checks; typed answers in which deterministic code performs counting, deduplication and date arithmetic over cited facts; routine reporting of confident-wrong rates; and repeated baseline runs to measure control spread.

References

- Chan, B. J., Chen, C.-T., et al. (2025). *Don't Do RAG: When Cache-Augmented Generation is All You Need for Knowledge Tasks*. arXiv:2412.15605.
- Chhikara, P., Khant, D., Aryan, S., Singh, T., Yadav, D. (2025). *Mem0: Building Production-Ready AI Agents with Scalable Long-Term Memory*. arXiv:2504.19413.
- Cuconasu, F., Trappolini, G., et al. (2024). *The Power of Noise: Redefining Retrieval for RAG Systems*. arXiv:2401.14887.
- Ge, Y., Romeo, S., et al. (2025). *TReMu: Towards Neuro-Symbolic Temporal Reasoning for LLM-Agents with Memory in Multi-Session Dialogues*. arXiv:2502.01630.
- Hong, K., Troynikov, A., Huber, J. (2025). *Context Rot: How Increasing Input Tokens Impacts LLM Performance*. Chroma technical report. <https://www.trychroma.com/research/context-rot>
- Hsieh, C.-P., Sun, S., et al. (2024). *RULER: What's the Real Context Size of Your Long-Context Language Models?* arXiv:2404.06654.
- Hu, E. J., Shen, Y., et al. (2021). *LoRA: Low-Rank Adaptation of Large Language Models*. arXiv:2106.09685.
- Huang, J., Chen, X., et al. (2024). *Large Language Models Cannot Self-Correct Reasoning Yet*. ICLR 2024. arXiv:2310.01798.
- Kang, J., Ji, M., et al. (2025). *Memory OS of AI Agent*. arXiv:2506.06326.
- Kim, S., Lee, Y., et al. (2025). *Pre-Storage Reasoning for Episodic Memory: Shifting Inference Burden to Memory for Personalized Dialogue (PREMem)*. arXiv:2509.10852.
- Kwon, W., Li, Z., et al. (2023). *Efficient Memory Management for Large Language Model Serving with PagedAttention*. SOSP 2023. arXiv:2309.06180.
- Latimer, C., Boschi, N., et al. (2025). *Hindsight is 20/20: Building Agent Memory that Retains, Recalls, and Reflects*. arXiv:2512.12818.
- Letta (2025). *Benchmarking AI Agent Memory: Is a Filesystem All You Need?* Blog post, August 2025. <https://www.letta.com/blog/benchmarking-ai-agent-memory/>
- Liu, N. F., Lin, K., et al. (2023). *Lost in the Middle: How Language Models Use Long Contexts*. TACL 2024. arXiv:2307.03172.
- locomo-audit (n.d.; accessed October 2026). *Full audit of the LoCoMo benchmark*. GitHub repository dial481/locomo-audit (third-party, not peer reviewed). <https://github.com/dial481/locomo-audit>
- Maharana, A., Lee, D.-H., Tulyakov, S., Bansal, M., Barbieri, F., Fang, Y. (2024). *Evaluating Very Long-Term Conversational Memory of LLM Agents*. ACL 2024. arXiv:2402.17753.
- Mem0 / Zep public discussion (2025). *Revisiting Zep's 84% LoCoMo Claim: Corrected Evaluation & 58.44% Accuracy*. GitHub issue getzep/zep-papers#5. <https://github.com/getzep/zep-papers/issues/5>
- Modarressi, A., Deilamsalehy, H., et al. (2025). *NoLiMa: Long-Context Evaluation Beyond Literal Matching*. arXiv:2502.05167.
- Packer, C., Wooders, S., et al. (2023). *MemGPT: Towards LLMs as Operating Systems*. arXiv:2310.08560.
- Rasmussen, P., Paliychuk, P., Beauvais, T., Ryan, J., Chalef, D. (2025). *Zep: A Temporal Knowledge Graph Architecture for Agent Memory*. arXiv:2501.13956.
- Shi, F., Chen, X., et al. (2023). *Large Language Models Can Be Easily Distracted by Irrelevant Context*. ICML 2023. arXiv:2302.00093.
- Wu, D., Wang, H., et al. (2024). *LongMemEval: Benchmarking Chat Assistants on Long-Term Interactive Memory*. arXiv:2410.10813.
- Xu, W., Liang, Z., Mei, K., Gao, H., Tan, J., Zhang, Y. (2025). *A-MEM: Agentic Memory for LLM Agents*. arXiv:2502.12110.
- Yan, S., Yang, X., et al. (2025). *Memory-R1: Enhancing Large Language Model Agents to Manage and Utilize Memories via Reinforcement Learning*. arXiv:2508.19828.
- Yang, A., Li, A., et al. (2025). *Qwen3 Technical Report*. arXiv:2505.09388.
- Yao, J., Li, H., et al. (2025). *CacheBlend: Fast Large Language Model Serving for RAG with Cached Knowledge Fusion*. EuroSys 2025. arXiv:2405.16444.

- Yoran, O., Wolfson, T., et al. (2023). *Making Retrieval-Augmented Language Models Robust to Irrelevant Context*. arXiv:2310.01558.
- Yu, W., Zhang, H., et al. (2023). *Chain-of-Note: Enhancing Robustness in Retrieval-Augmented Language Models*. arXiv:2311.09210.
- Zep (2025). *Lies, Damn Lies, & Statistics: Is Mem0 Really SOTA in Agent Memory?* Blog post (corrected version). <https://www.getzep.com/blog/lies-damn-lies-statistics-is-mem0-really-sota-in-agent-memory/>

Third-party figures are reproduced as reported by their authors; several are self-reports by system developers and were obtained under different protocols.

Appendix A. Number provenance and conventions

All figures for our system come from stored per-question outputs and the experiment log of the study. We use three labels: *measured* (read from stored outputs), *derived* (arithmetic on measured values, marked "derived" or "computed by us"), and *estimated* (marked "about"). The Wilson confidence intervals and the sign-test p-values for the 8/12 and 14/21 win/loss splits were computed by us from the reported counts. The p-values for the date-anchor result (0.020) and for second-pass retrieval (0.027) are as recorded in the experiment log. All LoCoMo results use the public dataset release. No test-question or gold-label information was used to design any intervention.