

Compressed Once, Read Many Times: Why Prompt-Compression Results Do Not Transfer to Agent Memory, What the Query-Agnostic Line Already Showed, and the Two Properties of a Memory Write No Protocol Yet Scores

Mahendrakar, Pranay

SONYTECH

ORCID: 0009-0003-7224-029X

Life of Research

2026-09-26

Pranay Mahendrakar

Abstract

Language-model agents that remember across sessions compress what they store: they summarise dialogue, extract facts, or evict cache entries, and then answer later questions from what is left. The compression ratios used to justify these designs come mostly from prompt-compression work in which the question is visible when compression runs. This paper examines whether those results transfer to the memory write, where the compressor must decide what to keep before any question exists. The premise that no one measures this does not survive the literature. The key-value cache line has built query-agnostic compressors and audited query visibility directly: in one matched-budget audit, the method the auditors call the most widely deployed beats a keep-the-start-and-recent-window baseline when it sees the question and loses to it when it does not. The paper argues that what remains open is narrower and harder. Every located "query-agnostic" result is still distribution-aware: the compressor is tuned for, or scored against, questions drawn from the same generator. A memory write has two properties no located protocol scores: its future query distribution is set by events that have not happened, and its compressions compose, because summaries are summarised again and records are overwritten. Published agent-memory evidence is consistent with that reading. Summary units are retrieved with 90.7 percent recall yet answer at 31.5 F1. In a 2024 pilot, two commercial assistants answering from fact stores written during the conversation scored 24.7 to 71.1 percent, where reading the raw history scored 91.8. The paper separates three query-visibility regimes, states what a write-time compression claim would have to report, and names the experiment that would settle the open part.

1. Introduction

A deployed assistant that remembers a user across months cannot keep everything in its context window, and it rarely keeps everything anywhere. MemGPT pages old messages out of the window and keeps a recursive summary of what was evicted (Packer et al., 2023). MemoryBank, Mem0, Zep and A-Mem extract, consolidate and rewrite what the user said into shorter records (Zhong et al., 2023; Chhikara et al., 2025; Rasmussen et al., 2025; Xu et al., 2025). Recursive summarisation replaces a dialogue history with a running summary that is itself rewritten each session (Wang et al., 2023). Compressed-context-memory methods fold accumulated key-value pairs into a fixed memory as the interaction grows (Kim et al., 2023). The benchmark that defines the "memory agent" setting states the design choice as a definition: memory "serves as a compressed and distilled representation of past information" (Hu et al., 2025a).

The evidence that such compression can be cheap comes largely from a neighbouring literature. Prompt compression reports "up to 20x compression with little performance loss" (Jiang et al., 2023a), performance gains of up to 21.4 percent "with around 4x fewer tokens" (Jiang et al., 2023b), compression of retrieved documents to 6 percent of their length "with minimal loss in performance" (Xu et al., 2023), 26x compression of prompts into gist tokens (Mu et al., 2023), 16x soft compression with an accuracy loss of 0 to 3 percent (Louis et al., 2025), and 30x fewer inference tokens for long-document question answering (Tan et al., 2024). A 2024 survey organises this work into hard-prompt and soft-prompt methods (Li et al., 2024a).

This paper's starting question is whether those numbers describe the operation an agent memory performs. The queue entry that motivated it put the worry this way: headline ratios come from settings where the query is available at compression time, while a memory write is compressed once for an unbounded set of future queries, none of which the compressor has seen. If that is right, the field's benchmark and the memory use case are measuring different quantities, and the loss stays invisible until a later question needs the dropped detail.

The worry is well supported, but it is not new, and the evidence complicates the premise in a specific direction. The key-value (KV) cache literature has already named the problem and built the instruments the queue entry said were missing. SCBench evaluates long-context methods over the full lifecycle of a shared, reused cache and reports that "sub- $O(n)$ memory methods suffer in multi-turn scenarios" (Li et al., 2024c). KVzip names itself a query-agnostic method and reports that query-aware eviction "suffer[s] from performance degradation even at a 90% cache budget ratio under multi-query scenarios" (Kim et al., 2025a). A 2026 matched-budget audit holds model, ratio, instances and decoding fixed and varies only whether the question is visible to the compressor. Under the query-agnostic protocol, SnapKV falls below "keep the start and the recent window" by 0.066 on average, after beating it by 0.132 when the question was visible (Luo et al., 2026). On the rate-distortion side, the optimal query-agnostic distortion-rate curve for prompt compression has been computed, and a practical query-aware method is "the only method to outperform the optimal query-agnostic strategy" (Nagle et al., 2024).

So the claim that no one measures query visibility is false. The claim this paper defends is narrower. It concerns what the existing query-agnostic instruments hold fixed, and what an agent memory does not hold fixed.

The argument has four steps. Section 3 locates where the headline ratios come from and shows that they conflate compression with relevance filtering: the query-aware method that scores 48.0 on LongBench at 5x compression is scoring above the uncompressed prompt's 44.0, so part of what it measures is retrieval (Pan et al., 2024). Section 4 reports the query-agnostic line and argues that every located result in it is distribution-aware. The rate-distortion limit is an expectation over a known joint distribution of prompts and queries (Nagle et al., 2024). KVzip optimises for all queries in a task set and scores tokens by how well the context can be reconstructed (Kim et al., 2025a). The newest one-to-many memory agents train against queries "sampled from $P(q|C)$ ", where that distribution is the benchmark's (Zhang et al., 2026b). Section 5 collects the agent-memory evidence, which is broadly consistent with a loss the downstream score does not show until the right question arrives. Sections 6 to 8 name the two properties of a memory write that no located protocol scores: the query distribution is set by future events, and the compressions compose. Both properties matter more because the one located test of loss detection found that it needed the question (Belikova et al., 2026). Section 9 gives the strongest counter-evidence, Section 10 states what a write-time compression claim would have to report, and Section 11 puts the numbers and the decision structure in one place.

The strongest objection to the whole enterprise is easy to state: agent memories keep the raw record anyway, so the compression is an index and not a loss. Some do. MemGPT writes every message to recall storage (Packer et al., 2023). Generative agents "store a complete record of the agent's

experiences" (Park et al., 2023). ReadAgent keeps page pointers and looks up the original text when its gists are insufficient (Lee et al., 2024). Where the raw record is retained and reachable, most of this paper's worry dissolves, and Section 9 says so. But the retention is a design variable, not a guarantee. The commercial systems measured on LongMemEval maintain "a set of memorized user facts". One of them "tended to overwrite crucial information as the chat continues", and the other "often failed to record indirectly provided user information" (Wu et al., 2024). Those are write-path losses. A compression ratio measured with the question visible cannot rule them out.

2. Scope, terms, and what this paper is not

Write-time compression. An operation that maps a stored item x (a dialogue turn, a session, an observation, a document, a block of KV pairs) to a shorter representation m before the questions that will be answered from m are known. A summary written at session end, a fact extracted from a turn, a gist written for a page and a KV cache evicted after prefill are all write-time compressions when the question arrives later.

Read-time compression. An operation that shortens an input after the question is known and for that question only. LongLLMLingua compressing retrieved documents given a question (Jiang et al., 2023b), RECOMP summarising retrieved documents for the query that retrieved them (Xu et al., 2023), and a KV press that scores tokens with the question inside its observation window (Li et al., 2024b) are read-time compressions.

Three query-visibility regimes. This is the paper's main distinction. It is a stipulation, offered to make the argument checkable.

Query-aware (QA) means that the compressor sees the specific question it is compressing for.

Distribution-aware (DA) means that the compressor does not see the question, but it was designed, trained, tuned or calibrated against a distribution of questions, and it is evaluated on questions from the same distribution. This regime covers learned compressors trained on question-answer pairs over documents (Louis et al., 2025), compressors whose instructions were optimised on a task's failures (Kang et al., 2025), task-guided eviction that uses a high-level task guide as a meta-query (Park and Hwang, 2026), and memory policies trained against sampled future questions (Zhang et al., 2026b).

Distribution-free (DF) means that the compressor has no knowledge of the question distribution beyond the stored item itself, and that the evaluation questions are drawn by a process the compressor could not have been fitted to.

The literature's own term "query-agnostic" covers both DA and DF. The paper's claim is that the located evidence for "query-agnostic" methods is evidence about DA. An agent memory that serves an open-ended user is closer to DF. No located evaluation places a compressor in DF.

Raw retention (R). Whether the uncompressed item survives the write and can be reached at read time. R is a design variable, not a property of the compressor. MemGPT and ReadAgent have $R = \text{yes}$ (Packer et al., 2023; Lee et al., 2024). A fixed-size compressed context memory has $R = \text{no}$ by construction (Kim et al., 2023).

Composition depth (d). The number of compressions between the original item and what the reader sees. A turn summarised once has $d = 1$. A running summary rewritten every session has d that grows with the number of sessions (Wang et al., 2023). A fact that is extracted, merged and later overwritten has d of at least 2.

What this paper is not. It is not an empirical paper; no system was run. It does not claim that compression is harmful to agent memory: the evidence in Section 9 shows some compressions help, and

those results are reported at the weight of the ones that hurt. It does not re-litigate long-context modelling; context-window extension is a different axis and is out of scope. It does not cover importance scoring, retention and forgetting policies, erasure, or memory poisoning, which are separate decisions about whether an item is kept at all. The question here is what happens to an item that is kept but kept shorter. It covers textual and KV compression and mentions soft-token compression where the evidence is comparable. It does not cover parametric memory, in which the store is the weights.

The paper is written as an analysis. Its evidence is the published record, read with attention to the protocol each number was measured under. Where a claim depends on a single paper, the text says so.

3. Where the headline ratios come from

A compression ratio carries three facts that the ratio alone does not report: what was compressed, who could see the question, and what counts as success. The headline numbers differ on all three.

3.1 The largest ratios compress redundant demonstrations with the question present

LLMLingua's "up to 20x" is reported on prompts structured as instruction, demonstrations and question. Its budget controller is "designed to allocate different budgets, i.e., compression ratio, to different components in a prompt such as instructions, demonstrations, and questions". It gives "more budget (i.e., smaller compression ratios) for instructions and questions, and less budget for demonstrations", because when there are many demonstrations "the conveyed information may be redundant" (Jiang et al., 2023a). On GSM8K the full few-shot chain-of-thought prompt scores 78.85 exact match at 2,366 tokens. LLMLingua at a one-shot budget scores 79.08 at 446 tokens (5x). Simply keeping one demonstration scores 77.10 at 422 tokens (6x) (Jiang et al., 2023a). So most of the ratio on that benchmark is available to a compressor that drops whole redundant examples. The source reports its 20x figure on the same kind of input.

This is not a criticism of LLMLingua on its own terms. It shows what the input is. A few-shot prompt is several instances of the same skill, and the question that will use it is part of the prompt. An episodic memory is the opposite case. Each record is usually unique, and the question that will use it does not yet exist.

3.2 Question-aware compression partly measures retrieval

LongLLMLingua is explicitly question-aware; its authors' later paper describes it as applying "a question-aware coarse-to-fine compression approach" (Pan et al., 2024). Its headline result is a gain, not a loss: "up to 21.4%" on NaturalQuestions "with around 4x fewer tokens" (Jiang et al., 2023b). The LLMLingua-2 paper reports both families on the same LongBench table (Pan et al., 2024). The uncompressed prompt, averaging 10,295 tokens, scores 44.0. At a 2,000-token constraint (5x), the question-aware LongLLMLingua scores 48.0. The question-agnostic LLMLingua-2 scores 39.1, LLMLingua 34.6 and Selective-Context 24.8. At a 3,000-token constraint (3x) the corresponding scores are 48.8 for LongLLMLingua and 42.4, 37.4 and 32.0 for the three agnostic methods. The authors attribute the gap to "the additional information that they leverage from the question" (Pan et al., 2024).

The question-aware method scores four points above the uncompressed prompt. Compression adds no information, so a score above the uncompressed input cannot come from preservation. The more plausible reading is relevance filtering: removing material the reader would otherwise be distracted by. That reading is consistent with the finding that language models use information in the middle of long contexts less reliably than information at the ends (Liu et al., 2023). Relevance filtering is a real and useful operation. It is the operation a retriever performs at read time. Citing it as evidence that a memory write can shorten a record by 4x without loss credits the write with work that only a read can do.

3.3 Learned compressors are trained on the question distribution they are scored on

Several of the strongest soft and learned compressors do not see the test question but were trained against questions like it. PISCO relies "solely on sequence-level knowledge distillation from document-based questions" and reports 16x compression with 0 to 3 percent accuracy loss on RAG question answering (Louis et al., 2025). RECOMP's compressors are "trained to improve LMs' performance on end tasks when the generated summaries are prepended" (Xu et al., 2023). LLoCO adds "in-domain parameter-efficient finetuning" to its offline context compression (Tan et al., 2024). The In-context Autoencoder is pretrained with autoencoding and language-modelling objectives and then "fine-tuned on instruction data" (Ge et al., 2023). The task-agnostic compressor TPC generates a task description with "a task descriptor trained on a curated dataset of context and query pairs" (Liskavets et al., 2025). In this paper's terms these are distribution-aware (Section 2). They are the right design when the question distribution at deployment matches the training distribution. They say nothing about a deployment where it does not.

3.4 Success is scored on one question per compression, or on similarity

The compression evaluations cited in this section score success in one of two ways. They pair each compressed input with one question from the benchmark, or they report similarity to the source. Selective Context, which is question-agnostic, reports its losses as "a minor drop of .023 in BERTscore and .038 in faithfulness" at a 50 percent reduction (Li et al., 2023). A similarity score of that kind cannot distinguish dropping a connective from dropping a date. Section 8 returns to this with direct measurements.

The engineering case has limits too. A large study of prompt compression in retrieval pipelines finds LLMingua achieving "up to 18% end-to-end speed-ups" when prompt length, ratio and hardware are well matched. Outside that operating window "the compression step dominates and cancels out the gains" (Kummer et al., 2026).

The survey that organises the field classifies methods as hard or soft prompts and compares them by mechanism (Li et al., 2024a). Query visibility is not one of its axes. The prompt-compression literature therefore offers no organising variable that would flag which of its results bear on a memory write.

The agent-memory surveys come closer without isolating the variable. One defines "Condensation" as one of six core memory operations and lists "lossy condensation" among the limitations of a fact-extraction memory layer (Du et al., 2025). Others organise the field by memory forms, functions and dynamics (Hu et al., 2025b), by how memory is designed and evaluated (Zhang et al., 2024), or as one component of context engineering (Mei et al., 2025). A 2026 framework places episodic memory, skills and rules on a single axis of increasing compression, from 5-20x to 1,000x and more. It observes that "evaluation methods are tightly coupled to compression levels" (Zhang et al., 2026c). These are differences of focus, not of quality. None of the located surveys treats the visibility of the future question at write time as the variable that decides whether a compression result transfers. That is the gap this paper occupies.

4. The premise, tested: the query-agnostic line already exists

The queue premise said the field's benchmark and the memory use case are not measuring the same quantity. For KV-cache compression this was said, measured and partly repaired between late 2024 and mid 2026. This section sets the record out, because the paper's contribution depends on what that line did and did not hold fixed.

4.1 The theory: query-agnostic compression has a limit, and it is worse

Nagle et al. (2024) formalise token-level prompt compression for black-box models as a rate-distortion

problem. They compute the optimal distortion-rate function by linear programming for both query-agnostic and query-aware compressors. On synthetic data, most existing methods are "far from the theoretical limit". Conditioning on the query "allows for a significant improvement". Their query-aware adaptation of LLMingua-2 is "the only method to outperform the optimal query-agnostic strategy". On NarrativeQA, approximated by beam search, "the gap between existing methods and the query-aware case is quite large, even for the methods that use the query to compress the prompt" (Nagle et al., 2024).

Two features of this result matter below. First, on the authors' synthetic data it exhibits a regime that a practical query-aware method reaches and no query-agnostic method can, since the optimum itself is beaten. A comparison that mixes the two protocols can therefore be comparing across a bound. Second, the query-agnostic distortion is defined as an expectation "taken with respect to the joint distribution" of prompt, query and answer (Nagle et al., 2024). The optimal query-agnostic compressor is optimal for a known query distribution. It is a distribution-aware object.

4.2 The benchmark: reuse breaks query-dependent compression

SCBench was built because "existing benchmarks often evaluate in single-request, neglecting the full lifecycle of the KV cache in real-world use", while KV reuse "has become widely adopted" in serving frameworks and by providers (Li et al., 2024c). It uses shared-context test examples across 12 tasks and evaluates eight categories of long-context method, including prompt compression. Its finding is that "sub- $O(n)$ memory methods suffer in multi-turn scenarios, while sparse encoding with $O(n)$ memory and sub- $O(n^2)$ pre-filling computation perform robustly" (Li et al., 2024c).

KVzip states the mechanism. Existing eviction methods "exhibit significant performance degradation in multi-query settings, as the retained KV pairs predominantly overfit to initial queries" (Kim et al., 2025a). Its Figure 2 compares SnapKV with a fresh prefill per query, SnapKV reusing the cache compressed for the first question, and KVzip. KVzip itself scores each KV pair by its contribution to reconstructing the original context, and reports 3 to 4x cache reduction "with negligible performance loss" across question answering, retrieval, reasoning and code comprehension. Context lengths reach 170K tokens (Kim et al., 2025a).

EpiCache makes the same observation for long conversations: "query-dependent eviction narrows the cache semantics to a single query, leading to failure cases in multi-turn conversations". Its episodic, clustered compression "achieves near full-cache accuracy under 4-6x compression" on LongMemEval, Realtalk and LoCoMo (Kim et al., 2025b).

4.3 The audit: the protocol decides who wins

Luo et al. (2026) provide the cleanest single measurement. They run six published KV presses against three trivial baselines (random eviction, a key-norm heuristic, and StreamingLLM's attention-sink-plus-recent window; Xiao et al., 2023) on three 7-9B models. Model, ratio, instances and decoding are held fixed, and every record is paired, over 144,300 RULER evaluations and 40,800 LongBench evaluations. The only variable is whether the question is visible when the press scores tokens.

Under the query-aware protocol, four of five backend-comparable presses beat the best trivial baseline on average. SnapKV does so by 0.132, and the audit reproduces the literature's picture. Under the query-agnostic protocol only KeyDiff, whose score "contains no query term at all", beats it consistently: 31 of 36 cells, mean +0.171. SnapKV averages -0.066. AdaKV moves from +0.154 to -0.018 (Luo et al., 2026, Table 2; Figure 1a). In the sharpest single cell (Qwen2.5-7B, retrieval, 90 percent evicted, uncompressed anchor 0.993), KeyDiff scores 0.663 and SnapKV 0.077 when the question is hidden. SnapKV recovers to 0.230 when it is visible.

The audit states its mechanism reading as a hypothesis on ordinal evidence over six methods. The per-

method drop is ordered by how visible the question is to each scoring signal, from +0.198 for SnapKV, whose 64-token observation window holds the question, down to +0.011 for KeyDiff (Luo et al., 2026). SnapKV's own title states the assumption the audit tests: "LLM Knows What You are Looking for Before Generation" (Li et al., 2024b). H2O's ranking is withdrawn by the audit because of an attention-backend confound (Luo et al., 2026; Zhang et al., 2023).

Two further results from the audit bear on this paper. ExpectedAttention, designed to score tokens "by predicting how future queries will attend to them" (Devoto et al., 2025), still loses to the keep-start-plus-recent baseline on the audit's grid (mean -0.051). The authors conclude that "query-agnostic design does not by itself guarantee agnostic-protocol wins". And KeyDiff's monopoly shrinks on natural text. On LongBench, other presses catch up or overtake it, and the authors conclude that "RULER inflated KeyDiff's uniqueness, not its effectiveness" (Luo et al., 2026). The ranking of query-agnostic methods is itself benchmark-dependent.

4.4 The repairs, and what they condition on

The KV line has produced purpose-built query-agnostic methods. Compactor uses approximate leverage scores. With "context-calibrated compression" it "achieves full KV performance on Longbench while reducing the KV memory burden by 68%, on average" (Chari and Van Durme, 2025). KeyDiff keeps keys that are geometrically distinctive (Park et al., 2025). TaskPress builds "a reusable memory representation conditioned on a high-level task guide" (Park and Hwang, 2026). RestoreKV adds a learned, context-conditioned restore cache under the same budget; at a 5 percent budget it raises KVzip from 38.2 to 73.2 on RULER-4K (Baek et al., 2026).

These are real advances, and they refute the reading of the queue premise that says nobody has built for compress-once, query-many. Read by what they condition on, they fall into three groups.

Reconstruction as a proxy for all future queries. KVzip's formal target is that the compressed cache give approximately the same output as the full cache for every query q in a task set T . The practical score is reconstruction of the context (Kim et al., 2025a). Reconstruction is a query-free proxy. Section 8 shows that reconstruction similarity and detail preservation can diverge. **An explicit model of the queries.** ExpectedAttention estimates how future queries will attend from the distributional properties of the model's activations (Devoto et al., 2025). TaskPress conditions on a task guide (Park and Hwang, 2026).

Query-free scores. KeyDiff keeps keys that are distinctive relative to the others, and Compactor uses approximate leverage scores (Park et al., 2025; Chari and Van Durme, 2025). These are the closest to distribution-free. Their evaluations are still RULER and LongBench, whose question generators also defined what the methods were developed and compared on.

RestoreKV sits across these groups: it keeps the base eviction rule and learns a restore mechanism by self-distillation from the full-cache model, "requiring no task-specific tuning" (Baek et al., 2026).

The question-agnostic text compressors follow the same pattern. LLMingua-2 is trained on data distilled from GPT-4's compressions of MeetingBank transcripts (Pan et al., 2024). Its "task-agnostic" label means it does not read the question, not that it was fitted to no task distribution.

This is the point on which the queue premise needs correcting rather than rejecting. The field has a query-agnostic protocol. No located work has a distribution-free one. Every located query-agnostic evaluation draws its test questions from the same benchmark family that fixed what "a question" looks like. For a KV cache over a contract "interrogated by dozens of questions" (Luo et al., 2026), that is a reasonable approximation: the document fixes what can be asked. For a memory of a person's life, it is not.

5. What the agent-memory evidence shows

The agent-memory literature does measure write-time compression directly, though it rarely labels it as such.

5.1 Summaries and extracted facts lose detail; atomic assertions sometimes help

LongMemEval embeds 500 questions in scalable user-assistant histories and compares value representations for the stored memory under a token budget (Wu et al., 2024). Decomposing sessions into rounds helps with GPT-4o as the reader. But "replacing sessions or rounds with extracted summaries or facts negatively impacts QA performance due to information loss". The one exception is multi-session reasoning, where fact decomposition consistently helps. The authors hypothesise that this is because facts put "the same type of information across all sessions in a more uniform and simplified format". Used alone as retrieval keys, condensed forms "do not enhance the memory recall performance". Condensed keys help only when added to the full value as extra retrieval paths. With the session as the stored value on LongMemEval_M, recall@5 is 0.706 when the session itself is the key and 0.572 when its summary is the key. End-to-end accuracy with GPT-4o reading the top five falls from 0.670 to 0.554 (Wu et al., 2024, Table 3).

LoCoMo's conversations average 300 turns and 9K tokens over up to 35 sessions (Maharana et al., 2024). Its RAG experiments compare three stored units read by gpt-3.5-turbo-16k (Table 3). Retrieving the top five raw dialogue turns gives overall F1 31.7 at recall@5 of 58.8. Retrieving the top five "observations", LLM-generated assertions about each speaker, gives F1 41.4 at recall 49.6. Retrieving session summaries gives F1 32.5 at recall 75.1 (top 5) and F1 31.5 at recall 90.7 (top 10). The authors note that summaries do "not significantly improve the performance despite high recall accuracies", "likely due to loss of information during the conversion of dialogs to summaries" (Maharana et al., 2024; Figure 1b).

The two papers agree on the shape. A compression that turns a session into prose (a summary) loses the detail questions ask for. A compression that turns it into short, uniform, attributable assertions (facts, observations) can help some question types. It helped multi-session reasoning in LongMemEval and temporal questions in LoCoMo, where observation F1 is 41.9 against 21.3 for dialogue at top 5. These are not contradictory. They say the form of the compressed record matters more than the ratio, and that the benefit depends on the question type. Both are facts about the question distribution.

5.2 Fact stores written during the conversation

LongMemEval's pilot study had human annotators chat with two commercial assistants that "maintain a set of memorized user facts", session by session, and then ask the question in a new session (Wu et al., 2024). On 97 questions over histories of three to six sessions, about ten times shorter than the benchmark's standard setting, offline reading of the full history by GPT-4o scored 0.9184. ChatGPT's memory scored 0.5773 with GPT-4o and 0.7113 with GPT-4o-mini. Coze scored 0.3299 with GPT-4o and 0.2474 with GPT-3.5-turbo (Figure 1c). The failure modes are named: ChatGPT "tended to overwrite crucial information as the chat continues", and Coze "often failed to record indirectly provided user information" (Wu et al., 2024). The first is a composition failure (Section 7). The second is a write-time selection failure: the writer did not recognise as memorable what a later question would need.

The measurement was taken in the first two weeks of August 2024, on 97 questions, through web interfaces. It describes those systems at that time. It is cited here for the failure modes and the size of the gap, not as a current ranking.

5.3 The same memory system, two benchmarks, two verdicts

Mem0 extracts, consolidates and retrieves salient facts. On LoCoMo its own paper reports an LLM-judge score of 66.88 using 1,764 memory tokens per conversation (68.44 for the graph variant at 3,616), against 72.90 for full context at 26,031 tokens. The authors state that full context "still achieves the highest J score" and report a 91 percent lower p95 latency for Mem0 (Chhikara et al., 2025). On that benchmark a roughly 15x reduction costs six judge points.

MemoryAgentBench feeds the same kinds of systems long inputs chunk by chunk and asks several questions after a single injection (Hu et al., 2025a). On its accurate-retrieval tasks, with GPT-4o-mini as backbone, Mem0 averages 32.6. The long-context GPT-4o-mini agent averages 49.2, and BM25 over raw chunks averages 60.5. On single-hop document QA the figures are 25.0 for Mem0 against 66.0 for BM25 (Hu et al., 2025a, Table 3). The authors note that Mem0 and Cogne perform "significantly lower than that of their backbone, GPT-4o-mini, even when the context length is relatively small". They also disclose that these systems were run at a uniform 4,096-token chunk size for cost reasons.

This paper cannot adjudicate between the two numbers, and it does not try. The chunk-size setting is a live confound. So are the judge metric and the question types. The disagreement is still informative. The same write-time compressor is near full context on a benchmark of personal conversations with fixed question categories, and far below raw-text retrieval on a benchmark built from long documents and synthetic events. That is what a distribution-aware compressor looks like when its evaluation distribution moves.

5.4 Gists with pointers

ReadAgent compresses pages of a long document into "gist memories" before seeing the question, then decides which original pages to look up (Lee et al., 2024). On QuALITY with PaLM 2-L, the full raw content scores 85.83 percent. Gist memory alone, at a compression rate of 85.53 percent, scores 77.52. Gists plus up to five page lookups score 86.83 (Lee et al., 2024, Table 1). The eight-point loss from gist-only reading is the cost of query-agnostic compression on that task. The lookup recovers it because the raw text was retained ($R = \text{yes}$).

5.5 Summary of the memory evidence

The located agent-memory results are consistent with three statements. Write-time compression into prose summaries costs answer quality even when retrieval metrics look good. Write-time compression into atomic assertions can raise answer quality for some question types. And the systems that avoid loss most reliably are those that keep the raw record reachable. None of the located studies varies query visibility or composition depth as a controlled variable. Section 10 turns that absence into a reporting requirement.

6. Property one: the query distribution of a memory write is set by events that have not happened

6.1 What every compressor needs and a memory write does not have

The information bottleneck makes the dependence explicit. It defines "the relevant information in a signal x " as "the information that this signal provides about another signal y ". It casts compression as "finding a short code for X that preserves the maximum information about Y ", a "generalization of rate distortion theory in which the distortion measure ... emerges from the joint statistics of X and Y " (Tishby et al., 2000). Without Y there is no relevant information to preserve, only the choice of what to call relevant. Rate-distortion prompt compression carries the same requirement: the query-agnostic distortion is an expectation over the joint distribution of prompts, queries and answers (Nagle et al., 2024).

So a write-time compressor always assumes a Y . Methods differ only in where the assumption lives. In a query-aware method it lives in the question. In a distribution-aware method it lives in a training set of questions, a task guide, a benchmark's question generator, or a set of failure cases used to tune instructions. In the methods closest to distribution-free it lives in a generic proxy such as reconstruction, perplexity or key geometry, which amounts to a bet that the future Y is everything, weighted as the proxy weights it.

The psychology of memory has long had names for the dependence of an encoding's value on the later test, among them "encoding specificity" (Tulving and Thomson, 1973) and "transfer appropriate processing" (Morris et al., 1977). It has also named the aim of an encoding that serves many tests "robust encoding" (Lockhart, 2002). This paper uses these terms only as vocabulary. It attributes no finding to those papers, and a human-memory analogy is not evidence about a language-model memory.

6.2 Why a benchmark's question generator is a poor stand-in

A memory benchmark fixes the question distribution when it is built. LongMemEval defines five abilities and curates 500 questions to test them (Wu et al., 2024). LoCoMo generates dialogues from personas and temporal event graphs and writes questions in five categories (Maharana et al., 2024). MemoryAgentBench prefaces each input chunk with an instruction such as "Please memorize it and I will ask some questions", and for its conflict-resolution tasks it tells agents that "newer facts have larger serial numbers" (Hu et al., 2025a). Every one of these choices tells the writer, or the person designing the writer, what kind of question is coming. A system tuned to do well on one of them is distribution-aware for that benchmark.

A deployed memory differs in three ways, each stated here as an argument rather than a measurement.

First, the question is authored by events that happen after the write. A user who mentions in March that a colleague is allergic to shellfish is asked about it in October only because a dinner is being booked. Nothing in March makes the allergy salient. The allergy is a detail tangential to the conversation's theme, and that is the category a theme-aligned compressor drops (Section 6.3).

Second, the distribution is nonstationary. What a user asks about shifts with their life, not with the benchmark's categories. A compressor fitted to last quarter's questions carries last quarter's notion of relevance.

Third, the reader is not fixed either. LongMemEval finds that the optimal retrieved-token budget depends on the reader: Llama 3.1 8B "drops sharply beyond 3k retrieved tokens", while GPT-4o "continues to improve even with over 20k" (Wu et al., 2024). A memory written today will be read by a model that may not exist yet.

The one-to-many memory agents of 2026 make the assumption explicit, and that is to their credit. UMA formulates a setting in which "query-agnostic external memory is constructed once from a stream and reused across multiple future QA sessions". It trains the memory policy on the mean reward of question-answering trajectories branching from each memory state, with the query "sampled from $P(q|C)$ " (Zhang et al., 2026b). This is the right training objective for the one-to-many problem when $P(q|C)$ is known. The $P(q|C)$ used is the one that generated the benchmark's questions. BudgetMem argues from the other side that "offline, query-agnostic memory construction ... may discard query-critical information", and moves the expensive processing to query time under a learned budget router (Zhang et al., 2026a). The two designs disagree about where to spend compute, which is the disagreement this paper is about. Both are evaluated on LoCoMo, LongMemEval or similar question sets, so neither evaluation can distinguish a distribution-aware writer from a distribution-free one.

6.3 Compressors keep what fits the theme

A 2024 study of gist-token context compression finds it "near-lossless" on retrieval-augmented generation and long-document QA but struggling on synthetic recall. It names three failure patterns: "lost by the boundary", "lost if surprise" and "lost along the way" (Deng et al., 2024). The second is the one that matters here. "Under constrained memory budgets, the model tends to prioritize retaining detailed information that closely aligns with the overarching theme of the context." The authors test this by inserting needles about the document's subject or about an unrelated "Mr. Tree" (Deng et al., 2024).

The finding bears on memory writes through an inference this paper cannot test, so it is stated as a conjecture, not a finding. If a compressor preferentially keeps theme-aligned detail, and if the details a future question will need are disproportionately the incidental ones, then the compressor's bias and the query distribution are anti-correlated. Theme-aligned material is what a summary would say anyway; incidental material is the allergy, the account number, the date mentioned in passing. No located study measures the rate at which memory questions target incidental versus theme-aligned details. Until one does, the conjecture is unquantified. It is included because it predicts a specific failure that aggregate scores would hide.

6.4 Within-task context compression is not the same problem

Several strong results on agent context compression concern a single task. There the future query is the task itself, and it is known at write time. The Complexity Trap compares SWE-agent strategies on SWE-bench Verified across five model configurations. It finds that simple observation masking "halves cost relative to the raw agent while matching, and sometimes slightly exceeding, the solve rate of LLM summarization" (Lindenbauer et al., 2025). With Qwen3-Coder 480B, masking is 52 percent cheaper than the raw agent and raises the solve rate by 2.6 points. ACON optimises compression guidelines on contrastive task feedback. It collects "tasks where the agent succeeds with H but fails with H'" (full versus compressed history) and asks an optimiser model what the compression lost. It reports 26 to 54 percent lower peak token usage and up to 46 percent improvement for smaller models (Kang et al., 2025). MEM1 trains, with reinforcement learning, a constant-size internal state that discards "the previous context (except for the prompt and initial query)" (Zhou et al., 2025). A single-author report of an agent that prunes its own history on five SWE-bench Lite instances found a 22.7 percent token reduction at unchanged accuracy (Verma, 2026). That is a small sample, cited only as an example of the design.

All four are query-aware in this paper's sense: the task or the initial query is present when the compressor decides what to keep. ACON's guideline optimisation also shows what detecting a compression loss requires: a paired run with and without compression, and a downstream outcome. Those results transfer to within-task working memory. They do not transfer to cross-session memory, where neither the task nor the outcome exists at write time.

7. Property two: memory writes compose

7.1 Compression depth grows with the life of the memory

A prompt is compressed once. A memory is compressed, then merged, then summarised again. Recursive summarisation for dialogue memory "recursively produce[s] new memory using previous memory and following contexts" (Wang et al., 2023). MemGPT keeps "a recursive summary of messages that have been evicted" at the head of its queue (Packer et al., 2023). Compressed context memory "continually compresses the accumulating attention key/value pairs into a compact memory space". Its training "model[s] the recursive compression process as a single parallelized forward computation" (Kim

et al., 2023). A-Mem's new memories "can trigger updates to the contextual representations and attributes of existing historical memories" (Xu et al., 2025). Mem0's update step consolidates new facts against existing ones (Chhikara et al., 2025). In each design, the compression depth of an early detail grows with the number of later writes.

7.2 What is known about iterated rewriting

No located study measures detail preservation in an agent memory as a function of composition depth. Three adjacent literatures bound the question from outside.

Book-length summarisation compares two workflows. "Hierarchically merging chunk-level summaries" and "incrementally updating a running summary" are the same two shapes a memory writer can take. Across 1,193 human annotations on GPT-4 summaries of 100 recent books, "incremental updating yields lower BoookScore but higher level of detail than hierarchical merging" (Chang et al., 2023). The trade-off between coherence and detail is therefore a function of the composition pattern, not only of the ratio.

Faithfulness studies show that abstractive compression adds errors as well as losing detail. A large human evaluation found "substantial amounts of hallucinated content in all model generated summaries" (Maynez et al., 2020). In book-length summaries by 2024 models, "most unfaithful claims relate to events and character states". They "generally require indirect reasoning over the narrative to invalidate", and no LLM rater of faithfulness tested correlates strongly with human annotations (Kim et al., 2024). The same study develops "a typology of omission errors related to crucial narrative elements", so omission is a recognised error class in summary evaluation, not only in memory. A memory record rewritten from an unfaithful summary inherits the error. A reader that does not hold the source cannot detect it.

Transmission-chain experiments show where iterated rewriting drifts. When LLM agents "iteratively receive, produce, and transmit texts from the previous to the next agent in the chain", small biases are amplified and text properties move toward "attractor states". "More open-ended instructions lead to stronger attraction effects" (Perez et al., 2024). A memory writer told to "summarise what matters" is an open-ended instruction applied to its own output.

These results are adjacent evidence, not direct evidence. They measure coherence, faithfulness and stylistic drift, not question-answering from a memory at depth d . They suffice to show that a single-shot compression ratio does not bound the loss after d compressions. They do not say how large that loss is.

7.3 Overwriting is composition with a destructive operator

The ChatGPT failure in the LongMemEval pilot, "overwrite crucial information as the chat continues" (Wu et al., 2024), is the limiting case. An update that replaces a record, rather than appending to it, is a compression whose output discards the prior version entirely. MemoryAgentBench's selective-forgetting tasks test whether agents keep the newest version of a conflicting fact. The best agent in its main table (GPT-5-mini with a 400K context) scores 28.0 on the multi-hop variant, and every other listed agent scores 7.0 or below (Hu et al., 2025a). Those tasks measure whether the right version survives, which is the complement of this paper's concern: whether a detail that should survive does. Both are properties of composition, and no located compression benchmark scores either.

8. The loss is invisible at write time

If write-time compression loses the detail a later question needs, can the writer know? The located evidence says not without the question.

8.1 In the one located test, detecting overflow needed the query

Belikova et al. (2026) define "token overflow" as "a regime in which compressed representations no longer contain sufficient information to answer a given query", and they try to detect it in xRAG's one-token-per-document soft compression (Cheng et al., 2024). Query-agnostic saturation statistics "reliably separate compressed from uncompressed token representations" but show "limited overflow detection capability". Probes over both query and context representations detect overflow with 0.72 AUC-ROC on average across HotpotQA, SQuADv2 and TriviaQA, "demonstrating that incorporating query information improves detection performance" (Belikova et al., 2026). The paper is a student workshop paper on one compressor. It is the only located direct test, and its direction is clear: the signal that a compression lost what matters lives in the question.

8.2 Similarity scores do not see detail loss

Lajewska et al. (2025) evaluate prompt compressors on downstream performance, grounding and information preservation. They reconstruct the original text from xRAG's compressed representation. On unseen data with one token per sample, the reconstruction has BERTScore F1 0.66 against the original, but only 28 percent of the original's named entities survive. For dates the figure is 22 percent and for cardinal numbers 26 percent. PISCO with eight tokens per sample reconstructs at BERTScore 0.89 and preserves 49 percent of entities (Lajewska et al., 2025, Tables 5 and 7). The authors observe "particularly low scores for dates and numerical values", and that xRAG tokens "primarily capture the general topic of the compressed content ... but fail to retain key details".

A reconstruction that is two-thirds similar to its source and keeps a quarter of its dates is what "near-lossless" can mean under a similarity metric. Dates and numbers are the details memory questions ask about. They are the basis of temporal reasoning and knowledge updates, two of LongMemEval's five abilities (Wu et al., 2024). KVzip's reconstruction objective is a better proxy than a single question, but it is still a proxy scored by how much of the context comes back. This measurement shows that reconstruction can come back fluent and on topic while the entities are gone.

8.3 Retrieval metrics count the unit, not the content

LoCoMo's summary units reach 90.7 percent recall at top ten while answer F1 stays at 31.5 (Maharana et al., 2024; Figure 1b). Recall is scored on whether the summary of the evidence session was retrieved. It was. The detail was not in it. Any memory evaluation that reports retrieval recall over compressed units, without answer accuracy on questions targeting specific details, will report success on this failure.

8.4 Consequence

These three results support the queue entry's phrase "the loss stays invisible until a later unanticipated query needs the dropped detail", and they show why it matters for evaluation design. In the located studies, a write-time loss was not detected by query-free signals (8.1), by similarity to the source (8.2) or by retrieval metrics over compressed units (8.3). On that evidence it becomes visible when a question that needs the detail is asked and answered wrongly. An evaluation whose questions do not target the detail the compressor happened to drop cannot see the loss. Each of the three results comes from a single study on particular compressors, and the writer's-signal result (8.1) from a workshop paper on one compressor. Together they make the invisibility argument plausible. They do not establish it in general.

9. Counter-evidence: where compression at write time helps, and where the worry dissolves

This section is placed before the recommendations so that they are read against it.

Compression can raise answer quality. LoCoMo's observations beat raw dialogue by almost ten F1

points at top five, and by twenty points on temporal questions (Maharana et al., 2024). LongMemEval's fact decomposition consistently helps multi-session reasoning (Wu et al., 2024). SeCom applies LLMlingua-2 to memory units before retrieval "as a denoising mechanism". It reports that this "consistently improves retrieval recall given different retrieval budgets K ... when the compression rate exceeds 50%", raising query similarity to relevant segments and lowering it to irrelevant ones (Pan et al., 2025). Observation masking matches LLM summarisation at lower cost in software agents (Lindenbauer et al., 2025), and ACON raises smaller models' success (Kang et al., 2025).

These results are real, and they bound this paper's claim. Four points about them are relevant. SeCom's compression keeps most of each unit. The ratio it uses retains 75 percent of tokens, and its gain is in retrieval, where removing redundancy sharpens similarity. Observations and facts help by changing the form of the record to short, uniform assertions, and the gain is concentrated in particular question types. The within-task results are query-aware (Section 6.4). And every one of them is measured on the benchmark whose questions the design was developed against. None of them shows that write-time compression preserves details that no one designing the system anticipated would be asked for.

Mild compression is often close to free. KVzip's 3 to 4x, Compactor's 68 percent reduction at full LongBench performance, EpiCache's near full-cache accuracy at 4 to 6x, and CCM's full-context performance "with 5x smaller context memory size" are all reported without the question at compression time (Kim et al., 2025a; Chari and Van Durme, 2025; Kim et al., 2025b; Kim et al., 2023). At these ratios the distribution-aware caveat may be small in practice. This paper claims only that it has not been measured.

Raw retention dissolves most of the problem. When the raw record survives and the reader can reach it, write-time compression is an index, and an index that loses detail costs recall rather than truth. ReadAgent's lookups recover the eight points its gists lose (Lee et al., 2024). MemGPT keeps the full message history in recall storage (Packer et al., 2023). LongMemEval's best key designs keep the value and add condensed keys beside it (Wu et al., 2024). For systems built this way the relevant question is whether retrieval over the compressed index finds the raw record. The LoCoMo recall figures show that it often does. The paper's concern applies with full force only where $R = \text{no}$. That covers fixed-size compressed memories, fact stores that replace the conversation, and deployments that discard raw logs for storage, cost or privacy reasons. It also covers any design whose reader is given only the compressed record, as happens whenever raw retrieval is disabled to save tokens.

Latency is a legitimate reason to compress. Mem0 reports a 91 percent lower p95 latency than full context at a cost of six judge points on LoCoMo (Chhikara et al., 2025). A deployment may rationally accept that. The objection here is to accepting it on the strength of a prompt-compression ratio measured under a different protocol, not to the trade.

10. What a write-time compression claim would have to report

The following is this paper's proposal, not a finding. Each item exists because a result in Sections 3 to 8 would have been misread without it.

R1. The query-visibility regime at compression time. State whether the compressor saw the question (QA), was designed, trained or tuned against the evaluation's question distribution (DA), or neither (DF). Where a method can run in more than one regime, report it in each, under a matched budget. The Luo et al. (2026) audit is the template: one variable, paired instances, trivial baselines. Without R1 a 4x ratio from LongLLMLingua and a 4x ratio from a memory writer look like the same claim. They are measured under the two protocols whose limits Nagle et al. (2024) computed separately.

R2. Where the evaluation questions came from, relative to the writer's design. State whether the questions were written by the same process that shaped the compressor: the same benchmark, the same

question categories, the same generator, or tuning on its failures. At least one question pool should be independent of all of these. Section 12 proposes one.

R3. Detail preservation by type, not similarity. Report the fraction of named entities, dates and numbers in the source that survive in the compressed record, or that can be answered from it. Report theme-aligned and incidental details separately where the source allows. Similarity metrics such as BERTScore should not be the only preservation measure, because they can read 0.66 while 28 percent of entities survive (Lajewska et al., 2025).

R4. Composition depth. For memories that are rewritten, report accuracy on a fixed set of details as a function of the number of compression or update passes since the detail was written. A single-pass ratio does not bound the loss at depth d (Section 7).

R5. Raw retention and reader access. State whether the raw record survives and whether the reader can reach it at answer time. Report accuracy with the compressed record alone as well as with raw access. The gap between ReadAgent's 77.52 and 86.83 (Lee et al., 2024) is exactly this number.

R6. Answer accuracy, not retrieval recall over compressed units. Retrieval recall on summary units measures whether the unit was found, not whether the detail was in it (Maharana et al., 2024).

R7. Trivial baselines under the same protocol. Include "keep the most recent", observation masking, and raw-text BM25 retrieval. Under the query-agnostic protocol a widely used KV press lost to keep-start-plus-recent (Luo et al., 2026). Observation masking matched LLM summarisation in software agents (Lindenbauer et al., 2025). BM25 over raw chunks beat every compressing memory system on MemoryAgentBench's accurate-retrieval tasks (Hu et al., 2025a).

R8. What was compressed. Report the redundancy of the input: the number of near-duplicate demonstrations, repeated turns or boilerplate blocks. On GSM8K, dropping to a single demonstration already yields 6x (Jiang et al., 2023a). A ratio on a redundant input is not evidence about a ratio on a non-redundant one.

None of R1 to R8 requires a new model. All of them require a benchmark designer or a method author to report something that is currently computed and discarded, or not computed. No located memory or compression paper reports all eight. Several of the agent-memory papers located report R6 and R7 in part, and the KV audit reports R1 and R7.

11. The numbers and the decision structure, in one place

Table 1 consolidates the measured values the argument rests on, one row per cited result, with the source named on the row. Algorithm 1 is this paper's own synthesis, not a result from any cited source. It states how a published compression result should be classified before it is used to justify a memory write, and the write-time decision structure that classification implies. Figure 1 plots rows 6, 15 and 14 of the table.

Table 1. Published values behind this paper's argument, one row per cited result. QA, DA and DF are this paper's regimes (Section 2), assigned by the author from each source's method description; the regime is not a label the sources use. All values are as reported by the source; no value was measured for this paper.

#	Result	Published value	Regime	Source
1	GSM8K, 8-shot CoT prompt vs compressed prompts	full 78.85 EM at 2,366 tokens; LLMingua 79.08 at 446 (5x); one demonstration kept 77.10 at 422 (6x)	QA (question in prompt)	Jiang et al., 2023a

2	LongBench average at a 2,000-token constraint (5x)	uncompressed 44.0 (10,295 tokens); LongLLMLingua 48.0; LLMLingua-2 39.1; LLMLingua 34.6; Selective-Context 24.8	QA vs DA	Pan et al., 2024
3	LongBench average at a 3,000-token constraint (3x)	LongLLMLingua 48.8; LLMLingua-2 42.4; LLMLingua 37.4; Selective-Context 32.0	QA vs DA	Pan et al., 2024
4	Optimal distortion-rate curves, synthetic prompts	a query-aware practical method is the only method to beat the optimal query-agnostic curve	QA vs DA (limit)	Nagle et al., 2024
5	KVzip, reusable cache	3 to 4x cache reduction with negligible loss; query-aware eviction degrades even at a 90 percent budget in multi-query use	DA	Kim et al., 2025a
6	KV presses vs best-of-3 trivial baseline, 36 RULER cells	SnapKV +0.132 visible, -0.066 hidden; AdaKV +0.154 vs -0.018; TOVA +0.198 vs +0.095; KeyDiff +0.183 vs +0.171 (wins 31 of 36 hidden)	QA vs DA	Luo et al., 2026
7	Sharpest audit cell (Qwen2.5-7B, retrieval, 90 percent evicted, anchor 0.993)	hidden: KeyDiff 0.663, SnapKV 0.077; visible: SnapKV 0.230	QA vs DA	Luo et al., 2026
8	Episodic KV compression for long conversations	near full-cache accuracy at 4 to 6x; up to 30 percent accuracy gain over baselines	DA	Kim et al., 2025b
9	Context-calibrated leverage-score compression	full KV performance on LongBench at 68 percent average KV memory reduction	DA	Chari and Van Durme, 2025
10	Learned restore cache under the same budget	KVzip 38.2 to 73.2 on RULER-4K at a 5 percent budget (Qwen3-4B)	DA	Baek et al., 2026
11	Soft compression, reconstruction vs entity survival (unseen data)	xRAG BERTScore 0.66, entities 0.28, dates 0.22, cardinals 0.26; PISCO BERTScore 0.89, entities 0.49	DA	Lajewska et al., 2025
12	Detecting when a soft compression lost what a query needs	query-agnostic statistics: limited detection; query-plus-context probes: 0.72 AUC-ROC average	-	Belikova et al., 2026
13	LongMemEval_M, session value, key = session vs key = summary	recall@5 0.706 vs 0.572; GPT-4o top-5 accuracy 0.670 vs 0.554	DA	Wu et al., 2024
14	Commercial memory assistants vs offline reading (97 questions, Aug 2024)	offline GPT-4o 0.918; ChatGPT 0.577 (GPT-4o), 0.711 (GPT-4o-mini); Coze 0.330 (GPT-4o), 0.247 (GPT-3.5-turbo)	write-time, regime unknown	Wu et al., 2024

15 LoCoMo RAG, overall F1 and recall@k by stored unit	dialogue top-5 31.7 F1, 58.8 recall; observations top-5 41.4, 49.6; summaries top-5 32.5, 75.1; summaries top-10 31.5, 90.7	DA	Maharana et al., 2024
16 Mem0 on LoCoMo, LLM-judge score	Mem0 66.88 at 1,764 tokens; graph variant 68.44 at 3,616; full context 72.90 at 26,031	DA	Chhikara et al., 2025
17 MemoryAgentBench accurate-retrieval average (GPT-4o-mini backbone)	Mem0 32.6; Zep 37.5; MemGPT 34.3; long-context agent 49.2; BM25 over raw chunks 60.5	DA	Hu et al., 2025a
18 QuALITY with PaLM 2-L	full text 85.83; gists only 77.52 at 85.53 percent compression; gists plus up to 5 lookups 86.83	DF gist, then QA lookup	Lee et al., 2024
19 SWE-bench Verified, Qwen3-Coder 480B	observation masking 52 percent cheaper than raw agent, solve rate +2.6 points	QA (within task)	Lindenbauer et al., 2025
20 Agent context compression with optimised guidelines	26 to 54 percent lower peak tokens; up to 46 percent improvement for smaller models	QA (within task), DA guidelines	Kang et al., 2025
21 Soft compression trained on document questions	16x with 0 to 3 percent accuracy loss on RAG QA	DA	Louis et al., 2025
22 Compression as denoising of memory units	LLMLingua-2 at 75 percent token retention; recall improves across budgets when compression rate exceeds 50 percent	DA	Pan et al., 2025

Rows 1 to 4 are Section 3's evidence that the headline ratios are query-aware or measured on redundant inputs. Rows 4 to 10 are Section 4's query-agnostic line, and they are the reason the queue premise does not survive in its original form. Rows 11 and 12 are Section 8's evidence that the loss is invisible at write time. Rows 13 to 18 are Section 5's agent-memory evidence. Rows 19 to 22 are Section 9's counter-evidence, and they are the rows a reader should weigh against this paper. Every "DA" in the regime column records that the evaluation questions came from the same benchmark family the method was developed against. No row is DF in its evaluation, which is this paper's central observation.

Algorithm 1. Classifying a published compression result before it is used to justify a memory write, and the write-time decision it implies. This is the author's own decomposition, offered to make the argument checkable. It is not a method, not a result, and not reproduced from any cited source.

Input: a published result (method M, ratio k, score s, benchmark B)

a target memory deployment D with reader model r

Output: what the result licenses for D

```
# Step 1: regime at compression time (Section 2)
if question visible to M when compressing: regime <- QA
elif M trained / tuned / calibrated on B-like questions,
    or B's questions share a generator with M's design data: regime <- DA
else: regime <- DF
```

```

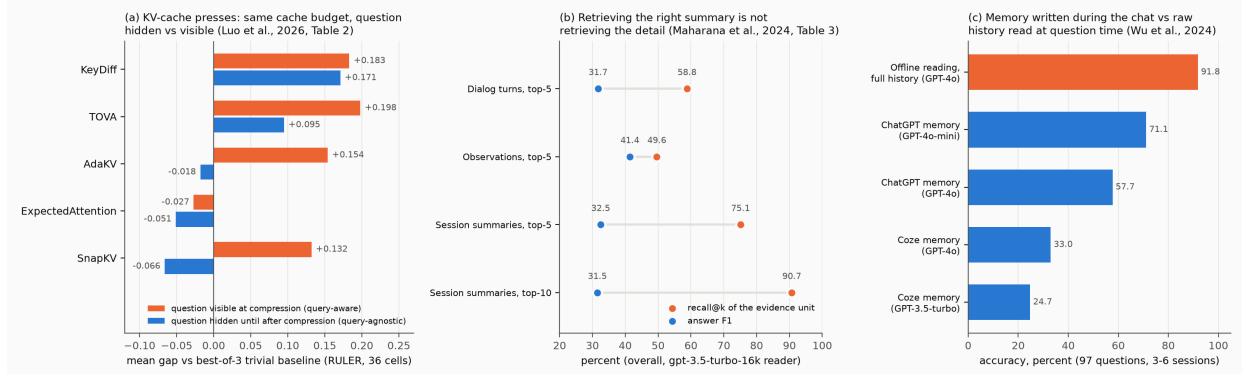
# Step 2: what a QA or DA result can and cannot say about a memory write
if regime == QA:
    return "read-time evidence only; part of s may be relevance
           filtering (s can exceed the uncompressed score)" # Sec 3.2
if regime == DA and question distribution of D != that of B:
    licensed <- "none until R2 is reported" # Sec 6.2
else:
    licensed <- "ratio k at score s, for depth d = 1 only" # Sec 7

# Step 3: at the write itself, in deployment D
for each item x arriving at time t:
    m <- compress(x) # no question exists yet
    if raw retention available and affordable:
        store (m as index, x as value) # R = yes: loss costs recall only
    else:
        store m # R = no: loss is permanent
        log the detail types in x absent from m # R3; cheap at write time
    on each later update touching m:
        depth(m) += 1 # R4; compounding is untracked today

# Step 4: the only signal that exposes a write-time loss (Section 8)
on question q at time t' > t:
    answer from m (and x if R = yes)
    a loss is observable only if q targets a detail absent from m
    -> evaluation must include questions that target such details (R2, R3)

```

Algorithm 1 makes the paper's claims locatable. Step 1 is Section 2's stipulation. The QA branch of Step 2 is Section 3. The DA branch is Sections 4 and 6. The depth qualifier is Section 7. Step 3's two logging lines are R3 and R4 of Section 10, and both are cheap to compute at write time, because the writer holds x and m together then and never again. Step 4 restates Section 8.



12. What would settle the open part

The open question is whether distribution-aware compressors lose more on questions their design did not anticipate than on the benchmark's own questions, and whether that loss grows with composition depth. It can be settled with existing systems and existing data. The experiment below is a proposal; nothing in it has been run.

Uniform detail probes as a distribution-free question pool. Take a long-conversation corpus such as LoCoMo or LongMemEval. Run a named-entity and temporal tagger over the raw history. Sample detail spans (entities, dates, quantities) uniformly at random, and turn each into a question whose answer is that span, verified against the raw history. The sampling process knows nothing about any compressor,

any summary, or the benchmark's question categories. It therefore approximates the DF regime for the purpose of detecting loss. It is not an estimate of what users ask. It is a stress test that no writer can have been fitted to.

Design. For each memory writer (a summariser, a fact extractor such as Mem0, an observation writer, a KV compressor such as KVzip or KeyDiff, and a raw-retention baseline), build the memory without question visibility. Answer three pools: the benchmark's own questions, the uniform detail probes, and questions from a second benchmark's generator applied to the same histories. Vary composition depth by running the writer's update step over the same history split into more and fewer sessions. Report R1 to R8.

What would count as each answer. If a writer's accuracy on uniform probes is close to its accuracy on the benchmark's questions, and the gap does not grow with depth, the queue's worry is empty for that writer and this paper's distinction is academic. If the gap is large for distribution-aware writers, small for the raw-retention baseline, and grows with depth, the prompt-compression ratios do not transfer and the write-time regime needs its own benchmark. A mixed result, for example large gaps for prose summaries and small ones for atomic facts, would locate the problem in the form of the record rather than in compression as such. That is the reading Section 5.1 already suggests.

A second, cheaper test. Re-score an existing memory benchmark's published system outputs by question type, separating questions whose answer is a theme-aligned detail from those whose answer is incidental to the session's topic. This needs annotation but no new runs. It would measure the anti-correlation conjectured in Section 6.3.

13. The strongest case against this paper

"The deployment distribution is not unknown; it is logged." A provider sees millions of memory queries and can fit $P(q)$ to them. On this view DA is the correct target, DF is a strawman, and the recommendation should be "evaluate on deployment logs", not "evaluate on uniform probes". The objection is partly right. A DA evaluation on the deployment's own logged questions, with a temporal holdout, would answer most of this paper's concern for that deployment. Two points remain. Logged questions are conditional on what the memory could answer; a detail that was dropped produces a failed or unasked question, not a logged one. And the logged distribution drifts (Section 6.2). The paper's claim is that published evaluations are DA with respect to benchmark generators, not deployment logs, and that no located one is DF. Both points stand under the objection.

"Nobody discards the raw record." Storage is cheap, and MemGPT, ReadAgent and LongMemEval's best designs keep it (Section 9). Where that is true the paper's concern reduces to retrieval quality over a lossy index, which is a smaller problem. But fixed-size compressed memories discard by construction (Kim et al., 2023). The commercial systems in the LongMemEval pilot answered from their fact stores (Wu et al., 2024). Privacy and retention policies can require discarding raw logs. The paper applies where $R = \text{no}$, and it asks that R be reported so readers can tell.

"Uniform span probes overweight trivia no user asks about." Yes. That is why Section 12 proposes them as one pool of three and calls them a stress test rather than an estimate. A writer that scores well on the benchmark and badly on uniform probes may still serve real users well, if real users never ask about incidental details. The paper conjectures that they do, and says in Section 6.3 that the conjecture is unmeasured.

"KV-cache evidence does not transfer to text memories either." Also true. A KV cache is compressed over one document for questions about that document. A text memory is compressed over a life for questions about anything. The KV results are used here for two things only: to show that query visibility changes rankings when everything else is held fixed, and to show that the query-agnostic protocol has

been built. Neither use needs the KV numbers to transfer in magnitude.

"The distinction between DA and DF is a matter of degree." It is. Any real compressor embodies some prior about what matters, including KeyDiff's key geometry and KVzip's reconstruction target. The stipulation is useful only because it names which prior a result was fitted to. If a reader prefers to treat regime as continuous, R2 still asks the same question: how far was the evaluation's question source from the source that shaped the compressor?

14. What is not known

No located study measures question-answering accuracy from an agent memory as a function of composition depth. No located study evaluates a memory writer on questions drawn independently of every process that shaped it. The rate at which real memory questions target incidental rather than theme-aligned details has not been measured. It is not known whether the Mem0 disagreement between LoCoMo and MemoryAgentBench is driven by chunk size, question type, metric or domain. It is not known whether the query-free KV methods that survive Luo et al.'s audit keep their advantage on conversational memory or at 32K to 128K contexts; the audit's authors say the latter is unmeasured. It is not known whether a query-aware overflow detector can be approximated at write time by a model of future questions. The only located detector needs the actual question.

15. Limitations

The hostile reviewer's first objection is that this paper's central observation, that no evaluation is distribution-free, is an absence claim over a literature too large to search exhaustively. The sweep covered prompt compression, KV-cache compression, soft context compression, agent memory systems and benchmarks, and summarisation faithfulness, through September 2026. It verified the 62 sources cited here. A DF evaluation could exist under vocabulary the search did not use. The claim is scoped accordingly: no located evaluation.

The regime labels in Table 1 are the author's assignments from method descriptions, not labels the sources use. A reader could reasonably move some rows. KVzip's reconstruction objective, in particular, is arguably closer to DF than to DA.

Several load-bearing numbers come from single papers: the audit (Luo et al., 2026), the overflow detector (Belikova et al., 2026), the commercial-assistant pilot on 97 questions (Wu et al., 2024) and the entity-survival measurements (Lajewska et al., 2025). Two of those are 2026 preprints. The Section 6.3 conjecture about theme-aligned versus incidental details is untested. The three psychology references are cited for their terms only.

The paper treats textual and KV memories together where the argument allows. Their mechanisms differ, and the magnitudes reported for one should not be read as estimates for the other.

16. Conclusion

The worry that prompt-compression results do not transfer to agent memory is correct, but the reason usually given for it is out of date. The question-visible protocol behind the headline ratios is real, and one of its gains is relevance filtering that only a read can perform. The field has not ignored this, however. The KV-cache line built query-agnostic compressors, reuse benchmarks and a matched-budget audit, and the audit shows the protocol decides which method wins.

What that line did not build is a distribution-free evaluation. Every located query-agnostic result is scored on questions from the family that shaped the compressor. A memory write differs in two ways no located protocol scores. Its questions are authored by events that have not happened yet, and its

compressions compose as records are summarised, merged and overwritten. The agent-memory evidence is consistent with a loss that similarity scores and retrieval recall do not show. Summaries are retrieved at 90.7 percent recall and answer at 31.5 F1. A reconstruction that scores 0.66 BERTScore against its source keeps 28 percent of the source's named entities. Assistants writing fact stores in 2024 scored well below a reader holding the raw history. In the one located test, detecting such a loss needed the question.

The recommendation is a reporting standard, R1 to R8, and an experiment that samples detail probes independently of every writer. Until a write-time compression claim reports its regime, its question source, its depth and whether the raw record survived, a 20x ratio measured on a prompt says nothing about a memory.

References

- Baek, C., Shin, S. & Kong, K. (2026). RestoreKV: Recovering Full-Cache Behavior Under Aggressive Query-Agnostic KV Cache Eviction. arXiv:2608.01247. doi:10.48550/arXiv.2608.01247. [PREPRINT]
- Belikova, J., Rozhevskii, D., Svirin, D., Polev, K. & Panchenko, A. (2026). Detecting Overflow in Compressed Token Representations for Retrieval-Augmented Generation. EACL 2026 Student Research Workshop. arXiv:2602.12235. doi:10.48550/arXiv.2602.12235.
- Chang, Y., Lo, K., Goyal, T. & Iyyer, M. (2023). BoookScore: A systematic exploration of book-length summarization in the era of LLMs. ICLR 2024. arXiv:2310.00785. doi:10.48550/arXiv.2310.00785.
- Chari, V. & Van Durme, B. (2025). Compactor: Calibrated Query-Agnostic KV Cache Compression with Approximate Leverage Scores. arXiv:2507.08143. doi:10.48550/arXiv.2507.08143. [PREPRINT]
- Cheng, X., Wang, X., Zhang, X., Ge, T., Chen, S.-Q., Wei, F. et al. (2024). xRAG: Extreme Context Compression for Retrieval-augmented Generation with One Token. NeurIPS 2024. arXiv:2405.13792. doi:10.48550/arXiv.2405.13792.
- 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. doi:10.48550/arXiv.2504.19413. [PREPRINT]
- Deng, C., Zhang, Z., Mao, K., Li, S., Huang, X., Yu, D. et al. (2024). A Silver Bullet or a Compromise for Full Attention? A Comprehensive Study of Gist Token-based Context Compression. arXiv:2412.17483. doi:10.48550/arXiv.2412.17483. [PREPRINT]
- Devoto, A., Jeblick, M. & Jegou, S. (2025). Expected Attention: KV Cache Compression by Estimating Attention from Future Queries Distribution. arXiv:2510.00636. doi:10.48550/arXiv.2510.00636. [PREPRINT]
- Du, Y., Huang, W., Zheng, D., Wang, Z., Montella, S., Lapata, M. et al. (2025). Rethinking Memory in LLM based Agents: Representations, Operations, and Emerging Topics. arXiv:2505.00675. doi:10.48550/arXiv.2505.00675. [PREPRINT]
- Ge, T., Hu, J., Wang, L., Wang, X., Chen, S.-Q. & Wei, F. (2023). In-context Autoencoder for Context Compression in a Large Language Model. ICLR 2024. arXiv:2307.06945. doi:10.48550/arXiv.2307.06945.
- Hu, Y., Wang, Y. & McAuley, J. (2025a). Evaluating Memory in LLM Agents via Incremental Multi-Turn Interactions. arXiv:2507.05257. doi:10.48550/arXiv.2507.05257. [PREPRINT]
- Hu, Y., Liu, S., Yue, Y., Zhang, G., Liu, B., Zhu, F. et al. (2025b). Memory in the Age of AI Agents. arXiv:2512.13564. doi:10.48550/arXiv.2512.13564. [PREPRINT]
- Jiang, H., Wu, Q., Lin, C.-Y., Yang, Y. & Qiu, L. (2023a). LLMlingua: Compressing Prompts for Accelerated Inference of Large Language Models. EMNLP 2023. arXiv:2310.05736. doi:10.48550/arXiv.2310.05736.
- Jiang, H., Wu, Q., Luo, X., Li, D., Lin, C.-Y., Yang, Y. et al. (2023b). LongLLMLingua: Accelerating and Enhancing LLMs in Long Context Scenarios via Prompt Compression. ACL 2024. arXiv:2310.06839. doi:10.48550/arXiv.2310.06839.
- Kang, M., Chen, W.-N., Han, D., Inan, H. A., Wutschitz, L., Chen, Y. et al. (2025). ACON: Optimizing Context Compression for Long-horizon LLM Agents. ICML 2026. arXiv:2510.00615. doi:10.48550/arXiv.2510.00615.
- Kim, J.-H., Yeom, J., Yun, S. & Song, H. O. (2023). Compressed Context Memory For Online Language Model Interaction. ICLR 2024. arXiv:2312.03414. doi:10.48550/arXiv.2312.03414.

- Kim, J.-H., Kim, J., Kwon, S., Lee, J. W., Yun, S. & Song, H. O. (2025a). KVzip: Query-Agnostic KV Cache Compression with Context Reconstruction. NeurIPS 2025. arXiv:2505.23416. doi:10.48550/arXiv.2505.23416.
- Kim, M., Kundu, A., Kim, H.-B., Dixit, R. & Cho, M. (2025b). EpiCache: Episodic KV Cache Management for Long-Term Conversation on Resource-Constrained Environments. ICML 2026. arXiv:2509.17396. doi:10.48550/arXiv.2509.17396.
- Kim, Y., Chang, Y., Karpinska, M., Garimella, A., Manjunatha, V., Lo, K. et al. (2024). FABLES: Evaluating faithfulness and content selection in book-length summarization. COLM 2024. arXiv:2404.01261. doi:10.48550/arXiv.2404.01261.
- Kummer, C., Jurkschat, L., Farber, M. & Vahdati, S. (2026). Prompt Compression in the Wild: Measuring Latency, Rate Adherence, and Quality for Faster LLM Inference. ECIR 2026. arXiv:2604.02985. doi:10.48550/arXiv.2604.02985.
- Lajewska, W., Hardalov, M., Aina, L., John, N. A., Su, H. & Marquez, L. (2025). Understanding and Improving Information Preservation in Prompt Compression for LLMs. Findings of EMNLP 2025. arXiv:2503.19114. doi:10.48550/arXiv.2503.19114.
- Lee, K.-H., Chen, X., Furuta, H., Canny, J. & Fischer, I. (2024). A Human-Inspired Reading Agent with Gist Memory of Very Long Contexts. arXiv:2402.09727. doi:10.48550/arXiv.2402.09727. [PREPRINT]
- Li, Y., Dong, B., Lin, C. & Guerin, F. (2023). Compressing Context to Enhance Inference Efficiency of Large Language Models. EMNLP 2023. arXiv:2310.06201. doi:10.48550/arXiv.2310.06201.
- Li, Z., Liu, Y., Su, Y. & Collier, N. (2024a). Prompt Compression for Large Language Models: A Survey. arXiv:2410.12388. doi:10.48550/arXiv.2410.12388. [PREPRINT]
- Li, Y., Huang, Y., Yang, B., Venkitesh, B., Locatelli, A., Ye, H. et al. (2024b). SnapKV: LLM Knows What You are Looking for Before Generation. arXiv:2404.14469. doi:10.48550/arXiv.2404.14469. [PREPRINT]
- Li, Y., Jiang, H., Wu, Q., Luo, X., Ahn, S., Zhang, C. et al. (2024c). SCBench: A KV Cache-Centric Analysis of Long-Context Methods. ICLR 2025. arXiv:2412.10319. doi:10.48550/arXiv.2412.10319.
- Lindenbauer, T., Slinko, I., Felder, L., Bogomolov, E. & Zharov, Y. (2025). The Complexity Trap: Simple Observation Masking Is as Efficient as LLM Summarization for Agent Context Management. DL4C workshop at NeurIPS 2025. arXiv:2508.21433. doi:10.48550/arXiv.2508.21433.
- Liskavets, B., Roy, S., Ushakov, M., Klibanov, M., Etemad, A. & Luke, S. (2025). Task-agnostic Prompt Compression with Context-aware Sentence Embedding and Reward-guided Task Descriptor. arXiv:2502.13374. doi:10.48550/arXiv.2502.13374. [PREPRINT]
- Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F. et al. (2023). Lost in the Middle: How Language Models Use Long Contexts. TACL. arXiv:2307.03172. doi:10.48550/arXiv.2307.03172.
- Lockhart, R. S. (2002). Levels of processing, transfer-appropriate processing, and the concept of robust encoding. Memory. doi:10.1080/09658210244000225.
- Louis, M., Dejean, H. & Clinchant, S. (2025). PISCO: Pretty Simple Compression for Retrieval-Augmented Generation. arXiv:2501.16075. doi:10.48550/arXiv.2501.16075. [PREPRINT]
- Luo, D., Liang, C. & Xuan, J. (2026). How Query Visibility Changes KV-Cache Compression Rankings: A Matched-Budget Audit. arXiv:2607.11942. doi:10.48550/arXiv.2607.11942. [PREPRINT]
- Maharana, A., Lee, D.-H., Tulyakov, S., Bansal, M., Barbieri, F. & Fang, Y. (2024). Evaluating Very Long-Term Conversational Memory of LLM Agents. arXiv:2402.17753. doi:10.48550/arXiv.2402.17753. [PREPRINT]
- Maynez, J., Narayan, S., Bohnet, B. & McDonald, R. (2020). On Faithfulness and Factuality in Abstractive Summarization. ACL 2020. arXiv:2005.00661. doi:10.48550/arXiv.2005.00661.
- Mei, L., Yao, J., Ge, Y., Wang, Y., Bi, B., Cai, Y. et al. (2025). A Survey of Context Engineering for Large Language Models. arXiv:2507.13334. doi:10.48550/arXiv.2507.13334. [PREPRINT]
- Morris, C. D., Bransford, J. D. & Franks, J. J. (1977). Levels of processing versus transfer appropriate processing. Journal of Verbal Learning and Verbal Behavior. doi:10.1016/s0022-5371(77)80016-9.
- Mu, J., Li, X. L. & Goodman, N. (2023). Learning to Compress Prompts with Gist Tokens. NeurIPS 2023. arXiv:2304.08467. doi:10.48550/arXiv.2304.08467.
- Nagle, A., Girish, A., Bondaschi, M., Gastpar, M., Makkuva, A. V. & Kim, H. (2024). Fundamental Limits of Prompt

Compression: A Rate-Distortion Framework for Black-Box Language Models. NeurIPS 2024. arXiv:2407.15504. doi:10.48550/arXiv.2407.15504.

Packer, C., Wooders, S., Lin, K., Fang, V., Patil, S. G., Stoica, I. et al. (2023). MemGPT: Towards LLMs as Operating Systems. arXiv:2310.08560. doi:10.48550/arXiv.2310.08560. [PREPRINT]

Pan, Z., Wu, Q., Jiang, H., Xia, M., Luo, X., Zhang, J. et al. (2024). LLMingua-2: Data Distillation for Efficient and Faithful Task-Agnostic Prompt Compression. Findings of ACL 2024. arXiv:2403.12968. doi:10.48550/arXiv.2403.12968.

Pan, Z., Wu, Q., Jiang, H., Luo, X., Cheng, H., Li, D. et al. (2025). On Memory Construction and Retrieval for Personalized Conversational Agents. arXiv:2502.05589. doi:10.48550/arXiv.2502.05589. [PREPRINT]

Park, J. S., O'Brien, J. C., Cai, C. J., Morris, M. R., Liang, P. & Bernstein, M. S. (2023). Generative Agents: Interactive Simulacra of Human Behavior. arXiv:2304.03442. doi:10.48550/arXiv.2304.03442. [PREPRINT]

Park, J., Jones, D., Morse, M. J., Goel, R., Lee, M. & Lott, C. (2025). KeyDiff: Key Similarity-Based KV Cache Eviction for Long-Context LLM Inference in Resource-Constrained Environments. NeurIPS 2025. arXiv:2504.15364. doi:10.48550/arXiv.2504.15364.

Park, W. & Hwang, S. (2026). TaskPress: Query-Agnostic KV Cache Compression via Task-Guided Pruning. EMNLP 2026. arXiv:2608.03276. doi:10.48550/arXiv.2608.03276.

Perez, J., Kovac, G., Leger, C., Colas, C., Molinaro, G., Derex, M. et al. (2024). When LLMs Play the Telephone Game: Cultural Attractors as Conceptual Tools to Evaluate LLMs in Multi-turn Settings. arXiv:2407.04503. doi:10.48550/arXiv.2407.04503. [PREPRINT]

Rasmussen, P., Paliychuk, P., Beauvais, T., Ryan, J. & Chalef, D. (2025). Zep: A Temporal Knowledge Graph Architecture for Agent Memory. arXiv:2501.13956. doi:10.48550/arXiv.2501.13956. [PREPRINT]

Tan, S., Li, X., Patil, S., Wu, Z., Zhang, T., Keutzer, K. et al. (2024). LLoCO: Learning Long Contexts Offline. EMNLP 2024. arXiv:2404.07979. doi:10.48550/arXiv.2404.07979.

Tishby, N., Pereira, F. C. & Bialek, W. (2000). The information bottleneck method. arXiv:physics/0004057. doi:10.48550/arXiv.physics/0004057. [PREPRINT]

Tulving, E. & Thomson, D. M. (1973). Encoding specificity and retrieval processes in episodic memory. Psychological Review. doi:10.1037/h0020071.

Verma, N. (2026). Active Context Compression: Autonomous Memory Management in LLM Agents. arXiv:2601.07190. doi:10.48550/arXiv.2601.07190. [PREPRINT]

Wang, Q., Fu, Y., Cao, Y., Wang, S., Tian, Z. & Ding, L. (2023). Recursively Summarizing Enables Long-Term Dialogue Memory in Large Language Models. Neurocomputing. arXiv:2308.15022. doi:10.48550/arXiv.2308.15022.

Wu, D., Wang, H., Yu, W., Zhang, Y., Chang, K.-W. & Yu, D. (2024). LongMemEval: Benchmarking Chat Assistants on Long-Term Interactive Memory. ICLR 2025. arXiv:2410.10813. doi:10.48550/arXiv.2410.10813.

Xiao, G., Tian, Y., Chen, B., Han, S. & Lewis, M. (2023). Efficient Streaming Language Models with Attention Sinks. ICLR 2024. arXiv:2309.17453. doi:10.48550/arXiv.2309.17453.

Xu, F., Shi, W. & Choi, E. (2023). RECOMP: Improving Retrieval-Augmented LMs with Compression and Selective Augmentation. arXiv:2310.04408. doi:10.48550/arXiv.2310.04408. [PREPRINT]

Xu, W., Liang, Z., Mei, K., Gao, H., Tan, J. & Zhang, Y. (2025). A-MEM: Agentic Memory for LLM Agents. NeurIPS 2025. arXiv:2502.12110. doi:10.48550/arXiv.2502.12110.

Zhang, Z., Sheng, Y., Zhou, T., Chen, T., Zheng, L., Cai, R. et al. (2023). H2O: Heavy-Hitter Oracle for Efficient Generative Inference of Large Language Models. arXiv:2306.14048. doi:10.48550/arXiv.2306.14048. [PREPRINT]

Zhang, Z., Bo, X., Ma, C., Li, R., Chen, X., Dai, Q. et al. (2024). A Survey on the Memory Mechanism of Large Language Model based Agents. arXiv:2404.13501. doi:10.48550/arXiv.2404.13501. [PREPRINT]

Zhang, H., Yue, H., Feng, T., Long, Q., Bao, J., Jin, B. et al. (2026a). Learning Query-Aware Budget-Tier Routing for Runtime Agent Memory. ICML 2026. arXiv:2602.06025. doi:10.48550/arXiv.2602.06025.

Zhang, K., Gui, S., Yang, S., Chen, W. & Feng, Y. (2026b). Learning to Remember: End-to-End Training of Memory Agents for Long-Context Reasoning. EMNLP 2026. arXiv:2602.18493. doi:10.48550/arXiv.2602.18493.

Zhang, X., Wang, G., Cui, Y., Qiu, W., Li, Z., Zhu, B. et al. (2026c). Experience Compression Spectrum: Unifying Memory, Skills, and Rules in LLM Agents. arXiv:2604.15877. doi:10.48550/arXiv.2604.15877. [PREPRINT]

Zhong, W., Guo, L., Gao, Q., Ye, H. & Wang, Y. (2023). MemoryBank: Enhancing Large Language Models with Long-Term Memory. arXiv:2305.10250. doi:10.48550/arXiv.2305.10250. [PREPRINT]

Zhou, Z., Qu, A., Wu, Z., Kim, S., Prakash, A., Rus, D. et al. (2025). MEM1: Learning to Synergize Memory and Reasoning for Efficient Long-Horizon Agents. arXiv:2506.15841. doi:10.48550/arXiv.2506.15841. [PREPRINT]

(c) 2026 Pranay Mahendrakar. Licensed under CC BY 4.0.

The literature search, drafting and citation verification for this paper were carried out with AI assistance under the author's direction. Every arXiv citation was machine-verified against its live arXiv Atom API record, and every other citation against its DOI record, during drafting (title and author list checked against the record returned). Every quantitative claim is taken from the abstract, full text or a table of the source credited with it; full-text numbers were read from the sources' own HTML renderings rather than from summaries. No experiment was run and no number in this paper was measured by its author. Table 1 re-presents numbers published by the cited papers, each named on its row; Figure 1 plots published values from three cited papers with no transformation beyond rounding. The three-regime classification, Algorithm 1 and the reporting protocol in Section 10 are original conceptual synthesis by the author, not empirical results, and are presented as such. Three psychology references are cited for terminology only; their full texts were not available to the verification step and no finding is attributed to them.