

What Is This Memory Useful For? Measuring Task-Conditioned Utility in Long-Term Agent Memory

Chih-Hsuan Yang^{*†}
Argonne National Laboratory
Lemont, Illinois, USA
bellayang@anl.gov

Feng Yan
University of Houston
Houston, Texas, USA
fyan5@uh.edu

Anjir Ahmed Chowdhury^{*}
University of Houston
Houston, Texas, USA

Rajeev Thakur
Argonne National Laboratory
Lemont, Illinois, USA
thakur@anl.gov

Project Website | Code | Dataset

Abstract

A long-horizon agent accumulates more experience than any future task needs and must decide which experiences to retain, retrieve and expose. A single importance score does not reveal how a memory’s effects vary across future tasks. We measure those effects with a task-blind writer and paired evaluations under controlled context: for a memory and a later task, a frozen reader answers the task with identical filler, with and without the target memory, and a ledger records what each memory cost to form, store and expose.

On single-call factual-recall tasks built from LongMemEval-V2 web-agent memories, the effects are strongly task-conditioned. Within the same memory, the mean paired reward gain on answer-bearing task pairs exceeds that on matched-control task pairs by +0.33 (95% interval +0.21 to +0.46). This contrast stays positive in a second session with randomized memory position and under a second reader, while its magnitude and individual memory–task labels change. Among answer-bearing memory units, the median unit bears on one screened task across the tested screen caps. The evaluated task-blind summaries and structured records show smaller reward gains than Raw, including under a fixed total context, and a budget sweep is consistent with retention of answer-bearing content contributing to the gains. Generic retrieval recovers screen-defined answer-bearing memories far above chance but misses most of them. We detect no excess harm for answer-bearing pairs, though the design could detect only a large increase, and we show that sign-flip and harm summaries need conditioned denominators. On a software-engineering and a scientific-coding benchmark, memories show no measurable effect even in raw form. The Conditional Memory Utility Dataset is publicly available on Hugging Face.

CCS Concepts

• **Computing methodologies** → **Knowledge representation and reasoning**; **Intelligent agents**; *Natural language processing*; *Machine learning*.

Keywords

agent memory; long-term memory; counterfactual evaluation; memory utility; memory representation; efficiency; LLM agents

1 Introduction

A long-horizon agent has to decide which past experiences to retain, which to retrieve for a task, and which to expose to the model that answers it. For retention, the relevant quantity is a memory’s expected effect over future tasks; for exposure, it is the effect on the task at hand. Both depend on how a memory’s effects vary across tasks: whether it helps many tasks or few, and whether a memory that helps one task can hurt another. Memory systems already score items [2, 19, 21], and prior work observes that value depends on the task [1, 9, 20]. What is missing is a measurement of those effects with the context held fixed, the writer blind to the task, and the memory’s cost recorded beside its effect. We build one and apply it to web-agent memory on LongMemEval-V2 [16].

Consider a hypothetical past page visit that shows a project’s budget. A task-blind writer stores that visit as a memory without seeing any future question. A later question asks for the budget, and the same reader answers it twice, with the same filler, once with and once without the stored memory; the reward difference is that memory’s effect on that question. Repeating the comparison for other questions and representations yields the memory’s task-conditioned profile.

We estimate these effects through controlled paired evaluation and test their sensitivity to evaluation conditions: the aggregate answer-bearing-versus-matched-control contrast remains positive across the tested sessions, positions and readers, while individual memory–task labels can change.

Scope. Our empirical evidence comes from single-call factual-recall evaluation: the memories are page visits from web-agent trajectories, the future evaluation tasks are questions with deterministically scored answers, and a reader answers each question in one call without acting in an environment. We make no claim about procedural memory, acting agents, or a trained retention policy.

Contributions. (1) **A conditional memory utility protocol** (Section 3): paired evaluation with identical context filler, a memory writer that never observes the future task, and a cost ledger

^{*}These authors are co-first authors and contributed equally to this work.

[†]Corresponding author: Chih-Hsuan Yang (bellayang@anl.gov).

that records formation, storage, retrieval and exposure costs beside each measurement. **(2) Bounded findings for factual recall** (Sections 5.2, 5.3, 5.4; Figures 3 and 4). Within the same memory, answer-bearing task pairs gain +0.33 [+0.21, +0.46] more reward than matched-control pairs. The evaluated task-blind compressed representations show smaller gains than Raw at their own length and at a fixed total context, consistent with reduced retention of answer-bearing text, and generic retrieval recovers only part of the screen-defined answer-bearing set (40% in its top ten, against about 1% by chance). **(3) Measurement-reliability findings** (Sections 3.4, 5.5, 5.6, 6). The headline contrast keeps its direction across a new session with randomized position (+0.50) and a second reader (+0.60), while its magnitude and per-pair labels change; sign-flip and harm summaries are confounded by pairs per memory and baseline correctness, and we show how to condition both away. **(4) A dataset** (Section 6) of per-pair measurements with costs, for reproducing the contrasts, evaluating retrieval against screen-defined labels, and studying variability.

2 Related Work

We organize prior work by the granularity of memory-utility measurement and the evaluation controls it provides. Several systems are designed around the idea that a memory’s value varies across tasks. A design rationale alone does not supply what a retention policy needs: how large the per-task effect is against execution noise, how often it occurs, and what the memory costs.

Measured, then averaged. UMEM scores a memory by a marginal-utility reward over a cosine neighborhood of $N=3$ future queries under GRPO [21]. The counterfactual is real, but it is reduced to its mean, and no dispersion or negative-utility statistic is reported. Their Lemma C.1 argues that the neighborhood captures the queries that retrieve the same memory; it does not report whether the dispersion inside that population is structured, which is what we measure.

One memory, several future tasks. EMBER’s MultiQ split tests whether one frozen memory serves 100 bundles of five questions each [9]. EvoMemEval releases per-cell labels from real execution with and without the memory on held-out tasks [1]; the labels are categorical, so they carry no effect size, and each cell runs once, so no flip separates from execution variance. Its released balance (exactly 100/100/100 by seeded down-sampling) is itself a design choice, so a rate computed on it describes the sample rather than a population. McClear computes leave-one-out and Shapley attributions from re-execution, but over the memories available to a single query [20]. A multi-factor value model notes that task utility is among the factors its scorer leaves out [2], and Memory-R1 trains a memory manager with one task per memory [19].

Constraints on our design. Memory-R2 argues that once rollouts write, update or delete different memories, trajectory-level comparisons become unfair [18]; our frozen memory set answers this in Section 3. Selective retention on clean benchmarks finds retention policies indistinguishable within Wilson 95% intervals [11], which is expected if no memory is harmful. In EvoMemEval’s released sample, pooled over its three datasets, the harmful label covers 22.0% of 537 items; we did not validate these labels. A learned gate over

measurements like ours could build on influence-guided context selection [5] and pre-generation decoding of tool necessity [13]; we propose none. Leave-one-out re-execution is shared with McClear, UMEM and influence-guided selection; context attribution by ablation is developed further in ContextCite [3], and per-item valuation in data Shapley [6]. What we add is the measurement design around it: profiles across tasks under controlled context, formation-cost provenance, and sensitivity checks. Our instability findings also have known causes. Long-context readers are sensitive to where relevant text sits [10], and temperature-0 inference is not reproducible when serving kernels vary with batch composition [7]. What is new is measuring their size on the quantity a memory evaluation reports. On representation, LongMemEval’s original study already varied the granularity at which history is stored and retrieved [17]; our length control asks a narrower question, whether a representation’s effect comes from what it says or from how many tokens it keeps.

3 Conditional Memory Evaluation

3.1 The measured quantity

Figure 1 shows the protocol end to end. We call the model that answers a future evaluation task the *reader*, and reserve “agent” for the motivating setting. A *memory unit* u is one page visit from a past episode; a memory m is the Raw, Summary or Structured text written from u . For a memory m and a later, distinct future task t , the conditional utility of m for t is the change in reward that adding m causes, given the context filler F , the position p at which m is inserted, and the seed s :

$$\Delta U(m, t | F, p, s) = R(t | F \cup_p m; s) - R(t | F; s). \quad (1)$$

A positive value means the memory helped and a negative value that it hurt; a zero difference indicates no measured change in reward, not identical answers or absence of useful information. We call these labels *helped*, *unchanged* and *hurt*; each holds under its evaluation condition. Both arms fix the reader model, serving cluster, decoding, prompt scaffold, output-token budget, and scoring function and version; they differ only in whether m is present. We call them the *with-target-memory* and *without-target-memory* arms (*with* m and *without* m); both contain the filler. Reward stays continuous wherever the benchmark gives partial credit; in the screened run it is binary. The per-cell estimate is the mean over replicates. We write $\Delta U(m, t)$ when F , p and s are fixed, and the conditioning matters: a single inserted item can change the answer for reasons unrelated to its content. The unit of analysis is the *cell*, a memory-task pair (m, t) under one evaluation condition; a memory has no single utility here, and its result is its profile across the tasks it faced. We compare at three levels: the *with- m* versus *without- m* reward difference of each pair; within each memory, the mean difference on answer-bearing task pairs minus that on matched-control task pairs (the headline contrast); and representations written from the same memory unit.

Identical filler, frozen memories. Both arms place the same controlled filler in the reader’s context, and the *WITH* arm adds m . The filler is 4 memory units of 1,500–5,000 tokens each, drawn from 50 trajectories disjoint from every trajectory that hosts a bearing memory, so it never contains another unit that bears on the task.

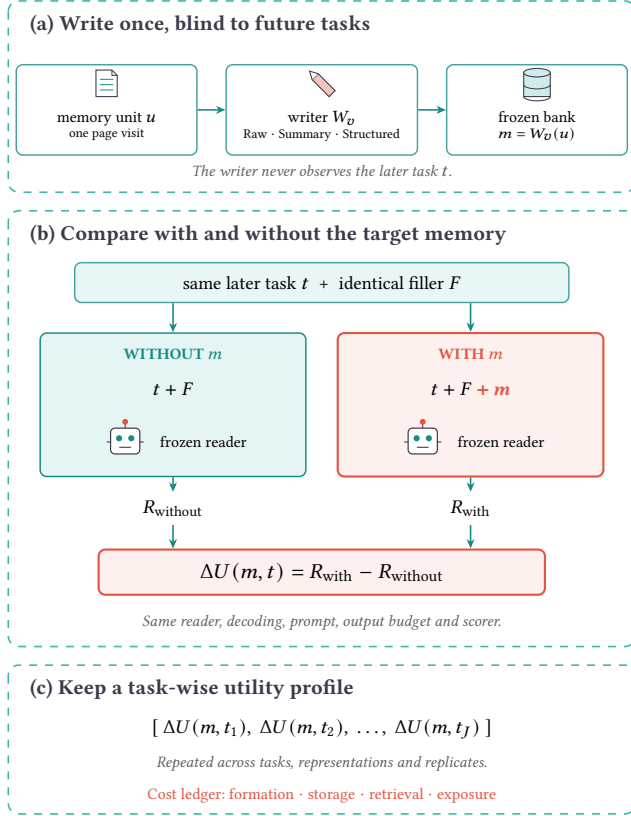


Figure 1: Conditional memory utility evaluation. (a) Page-visit memory units extracted from past episodes are stored as **Raw**, **Summary** or **Structured** representations without exposing the writer to future evaluation tasks. (b) For a memory-task pair, a frozen reader is evaluated with identical filler F , either with or without the target memory m ; the paired reward difference defines $\Delta U(m, t)$ under the recorded evaluation conditions. (c) Repeating this comparison across tasks, representations and replicates yields a task-conditioned utility profile. Formation, storage, retrieval and exposure costs are recorded alongside the measurements (the cost ledger).

The **WITHOUT** arm is therefore not an empty-context arm. On Long-MemEval’s first version [17] the oracle context is entirely evidence, so ablating “the memory” there would delete the whole context. The memory set is written once, frozen and content-hashed before any pairing exists, and no arm writes, updates or deletes a memory. The arms differ by exactly one item from that frozen set, so the divergent-state hazard raised by Memory-R2 [18] does not arise. The position of the added memory is not neutral: moving only that position changed the answer in 8 of 12 cells of a pilot, and 10% of labels at scale (Section 6). In the main runs we hold position fixed across replicates, vary it across cells, and record it on every row (Supplement S1.2). Every contrast we report is therefore conditional on the recorded positions; position variance is not inside our intervals.

The writer never sees the future task. Memory formation is a function of the memory unit alone, $m = W(u)$ and never $m = W(u, t)$. Without this invariant, $\Delta U > 0$ would only show that a model can be told an answer and copy it forward. We enforce it structurally rather than by convention: the writer takes a segment and nothing else, emits a closed field set, and has a single interpolation point. Each of these barriers carries a negative control that plants the defect and requires the check to fire (Supplement S1.1).

3.2 The answer-bearing screen is an instrument, not a retriever

To decide which (memory, task) pairs are worth executing, we use an *answer-bearing screen*. For each pair it reads three inputs: the memory’s text, the task’s **gold answer**, and the task’s evaluation specification. It never reads the question, a reward, or any model output, so the strata it defines are not selected on the outcome. A pair is *answer-bearing* (BEARING) when the benchmark’s own scorer would mark the memory text correct against the gold answer *and* the test singles this memory out of its domain pool, firing on at most eight of the 865–1,024 units available to that task.

Answer-bearing is a screening label based on answer containment, not a measured positive-utility label: paired evaluation measures the effect, and screen-selected pairs can still fail while pairs outside the screen can still help. Our sparsity result is a statement about how answers are distributed over memories. Because it consumes the gold answer, the screen cannot be deployed, and we do not propose it as a retriever. Its role is to define, with the gold answer, the answer-bearing pairs; these screen-defined labels, not exhaustive ground truth for useful memories, are what we evaluate retrieval against (Section 5.3). Every rate we report under the screen is conditional on a screened design.

A screen selected on the gold answer invites a second objection: perhaps the effect is an artifact of selecting pairs with the answer in hand. We tested this with a gold-free screen, pre-registered before execution: the same model as the reader (gpt-oss-120b), asked whether the memory contains the specific information the question asks about, without the gold answer. On 32 cells the gold screen *never* selected, the gold-free picks move the reader from 15.6% correct without the memory to 50.0% with it, while 42 matched-control pairs move from 14.3% to 11.9%. The contrast is +0.37, with a task-clustered 95% interval of [+0.05, +0.70] over 44 tasks and an episode-clustered one of [+0.14, +0.67] over 19 episodes. This answers the selection objection, not the containment one; because the judge is the reader’s own model, its picks may favor pairs that model finds easy. Its picks overlap the gold screen little (Supplement S1.3) yet carry an effect of similar size, so benefits also occur outside the gold screen. A second, lexical gold-free screen did not locate answer-bearing pairs (Supplement S1.3).

3.3 Three representations of one experience

All three representations derive from the *same* memory unit, so a downstream difference is attributable to how the segment is written rather than to which content was selected. **Raw** is deterministic and uses no model: the memory is the segment itself. **Summary** is one writer call per segment at pinned decoding and prompt version. **Structured** is one writer call that emits a validated schema

(memory type, trigger, action, outcome, lesson, scope). A malformed structure is recorded as a failure and never repaired.

Formation cost belongs to the representation. Raw makes zero writer calls; Summary and Structured make one each per segment. Omitting that cost flatters the model-written representations, so we charge each its own, including writes that failed after consuming a call.

3.4 Noise, labels and cost

Noise is measured, not assumed. Replicates are separate requests (0 of 5,198 calls served from cache), and within a session they agree on all 1,157 primary cells, so the execution floor is zero and a cell’s label (helped, unchanged or hurt) is the sign of ΔU ; the pre-registered floors and label rules are in Supplement S3. Truncated completions are re-issued, not scored. Intervals are over memories, tasks or episodes, never replicates.

A memory shows a **sign flip** when it helps one task and hurts another. A flip rate is descriptive, not evidence of memory $\times$ task interaction: under a purely additive model $\Delta U = \mu + a_m + b_t$, simulated over 20 trials of 200 memories $\times$ 8 tasks, flip rates range from 53.5% to 92.0% with no interaction present (Supplement S3). A flip rate also grows with the number of cells per memory, so it can only be compared across groups at matched cell counts.

The perturbation floor. The noise that matters most here is not seed variation but the reader’s response to inserting *any* item: position alone changes 10% of per-cell labels (Section 6). We ran no placebo arm of length-matched irrelevant content, so per-cell labels include such perturbations. The random-task stratum bounds their size: a memory paired with a randomly selected task helps in 11.8% of the pairs where the reader was wrong without it and hurts in 14.5% of those where it was right. These rates are an upper bound on pure perturbation: the memories are real same-domain pages, and where they contain the answer string they help more often (15.5% against 10.1%). Irrelevant context can distract a reader [12] or even help it [4]. We therefore report effects as paired contrasts between strata, which reduce sensitivity to content-independent perturbation when its mean effect is comparable across the contrasted strata, and never as counts of labeled cells; without a placebo arm we do not claim to isolate every such effect.

Pre-registered and exploratory analyses. The screened design, both gold-free screens, the length control, the second-reader study, the writer $\times$ reader factorial, the reversal rate and the variance decomposition were pre-registered; Supplement S10 lists their anchors and every deviation. All other analyses, including the three follow-up runs, are exploratory, and their intervals are not adjusted for multiplicity.

Every measurement is paired with its cost in seven families, from storage to formation; utility per exposed token is reported beside utility per token charged, including the writer’s (Supplement S4.5).

4 Benchmarks and Setup

LongMemEval-V2 and the page-visit memory unit. Our primary benchmark is the small tier of LongMemEval-V2 [16], where each domain has exactly one haystack, so every memory is available to every question in its domain. A whole trajectory cannot serve as a

memory: its median length is 103,886 tokens. We segment at the *page visit*, the maximal run of consecutive states that share one URL, a boundary drawn by the past agent’s own navigation. This yields **1,889 memory units** with a median of 7,261 tokens. Removing one unit removes a median 3.33% of its trajectory and 0.058% of the domain haystack, and segmentation preserves answerability for 278 of 295 deterministically scored questions (94.2%).

Future tasks and the executed design. A future task is one question with its evaluation specification. Of 451 questions, 295 are deterministically scored; we use the benchmark’s scorer verbatim, and an LLM-judged task raises an error rather than scoring zero. Seeded pairing within domain gives a grid of 15,112 memory–task pairs. We execute a *screened design* over 60 memories, a floor fixed before running by a power study: 40 from the web domain and 20 from the enterprise domain (unit pools of 865 and 1,024), drawn from 35 source episodes. A content-blind size cap fixed before any outcome removed 11 candidate memories longer than 20,000 tokens. The 60 memories are every memory bearing on two or more tasks that passed the cap (35) and 25 that bear on one, so the design over-samples multi-task memories to give reversal a chance. Each memory is paired with three strata of future evaluation tasks: *answer-bearing task pairs* (BEARING); *matched-control task pairs*, tasks of the same family as a bearing task (static environment, dynamic environment or procedure) on which the screen does not fire for that memory; and the *random-task stratum* (RANDOM), unscreened tasks drawn at random, over 218 tasks in all. All three strata contain the identical 60 memories, so contrasts are paired within memory rather than against a different set of control memories. The design has 1,157 cells (Table 1).

Secondary benchmarks. We also ran the protocol on SciCode [15] (reusable helper routines, 154 donor→recipient pairs) and SWE-bench Verified [8] (40 interpretable cells); there memories show no effect even in raw form, so we report them in Supplement S6.

Execution. In the primary arms, the reader and the memory writer are both gpt-oss-120b on one serving cluster, with low reasoning effort, temperature 0 and max_tokens 512; similarity uses SFR-Embedding-Mistral. Section 5.5 changes only the reader, to gemma-4-31B-it. An empty or truncated completion raises and the cell is re-run, because scoring it as zero would push ΔU negative and manufacture harmful memories. The full configuration, caching and saturation checks are in Supplement S3.

5 Results

5.1 Specific answers are sparse over memories

Screening all 1,889 memory units against the 295 deterministically scored questions yields 75 discriminative tasks: questions for which the screen fires on at least one and at most eight units of the domain pool. Over those tasks, 165 memories bear on at least one, across 266 memory–task edges, and the distribution is concentrated at one (Figure 2a): 119 memories bear on exactly one task, 18 on two and 18 on three, with a tail reaching nine. *Among the 165 memory units that bear on at least one of the 75 qualifying tasks, the median bears on one; the other 1,724 units bear on none, so this is not a median over all 1,889 units. Only 46 memories bear on two or more*

Table 1: Every run a reported number comes from. Counts are memory–task pairs times the conditions each pair was evaluated under (representations, arms or replicates, as named). A dataset row is one replicate of one pair under one condition and records both arms. Excluded rows had no producible reward and are dropped, never scored as zero.

run	memories	pairs $\times$ conditions	excluded
<i>LongMemEval-V2</i>			
screened design	60	1,157 pairs	0 rows
representations	57	277×3 representations	3 memories
length control	57	277×8 arms	6 rows
second reader	60	275 of 285 pairs	20 rows
gold-free screen	–	$32 + 42$ pairs	–
second session	60	285×3 replicates	2 rows
fixed context	57	277×5 arms	63 pairs
retrieval	–	75 tasks (screen labels)	–
SciCode	34	391 of 462 pair–repr.	71 pair–repr.
SWE-bench	–	141 of 321 pair–repr.	180 pair–repr.

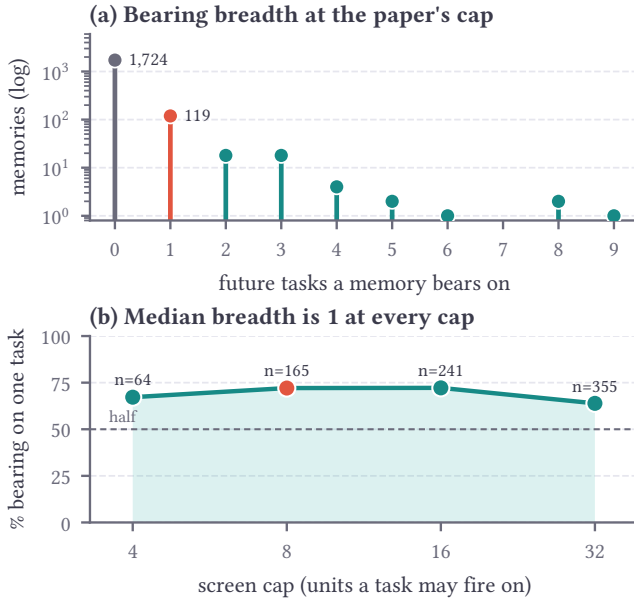


Figure 2: Answer bearing is sparse at every screen cap (exploratory). (a) Number of memory units (of 1,889) that bear on each count of screened future tasks, at the paper’s cap of eight units per task (log axis). (b) Share of bearing memories that bear on exactly one task when the cap is 4, 8, 16 or 32 units; n is the number of bearing memories. The median breadth is one at every cap.

tasks, and they come from 23 source trajectories, so they are not 46 independent draws; we report episode-clustered intervals wherever this matters.

These are screen statistics, not re-execution results. The screen’s cap of eight units is what makes it specific: of the 220 deterministic questions it excludes, 15 fire on no unit and 205 on more than eight,

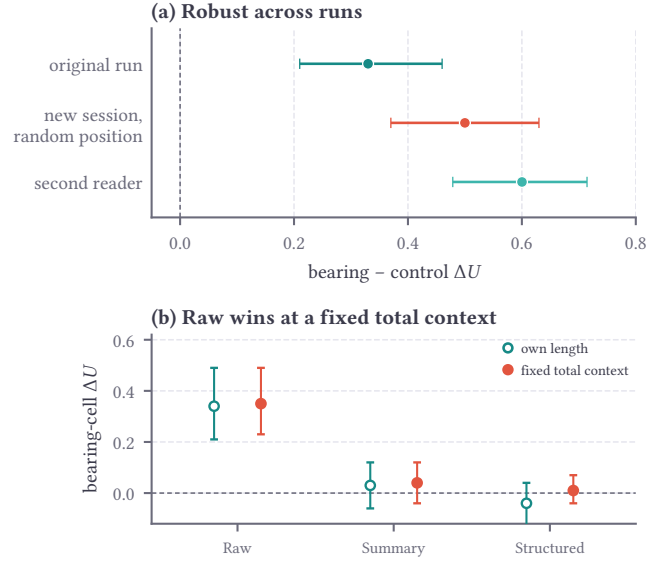


Figure 3: The result holds (LongMemEval-V2). (a) Bearing minus control, paired within memory, in the original run, in a new session with the memory’s position redrawn per replicate, and under a second reader (95% intervals over memories). (b) Bearing-cell ΔU of each representation at its own length ($n = 135$) and with every arm padded to the Raw arm’s token count ($n = 77$, excluding cells exposed to a cache-key defect); task-clustered intervals. Pre-registered: the original run and the second reader in (a); exploratory: the new session in (a) and panel (b).

often because short answers, such as a single number or letter, occur on many pages. Without the cap, string containment alone fires for every one of the 1,889 memories, on 18 to 91 tasks each. Any cap restores sparsity, and the exact value does not matter: at caps of 4, 8, 16 and 32 units the median bearing memory bears on exactly one task, and 64–72% bear on exactly one (Figure 2b). The sparsity we report is therefore the sparsity of *specific* answers over memories, in this benchmark, unit and task set. It is also a lower bound: cells outside BEARING contribute more helped cells (92) than bearing cells do (50), most at the perturbation floor (Section 3.4), and the gold-free screen’s picks lift accuracy from 15.6% to 50.0% on cells the gold screen never selected (Section 3.2). A gold-free definition does not make bearing denser, though: run over all 277,594 (task, unit) pairs under the same cap, the gold-free screen confirms only 42.

Sparsity makes indiscriminate pairing uninformative. When each memory is paired with same-domain questions chosen for question-type spread, without requiring that it bear on them, the reader fails both with and without the memory in 376 of 479 and 402 of 480 cells in two prompt arms, about 80%. Such a cell yields $\Delta U = 0$ whether or not the memory is useful, so a null there is an absence of evidence, not evidence of absence.

Table 2: Screened run on LongMemEval-V2: the same 60 memories paired with three strata of tasks. Both-arms-fail is the fraction of cells in which the reader fails with *and* without the memory; such cells cannot show a memory effect. The criterion for a material reduction (BEARING’s Wilson upper bound below the comparator’s point estimate) was fixed in code before these numbers existed.

stratum	both-arms-fail	Wilson 95%	mean ΔU
BEARING	0.53	[0.44, 0.61]	+0.35
matched control	0.79	[0.71, 0.85]	+0.03
RANDOM	0.70	[0.67, 0.73]	+0.06

5.2 Screening makes the effect measurable

The screened design reduces these structural zeros: cells that fail in both arms fall to 0.53 for BEARING, against 0.79 for the matched control and 0.70 for RANDOM (Table 2). The memory helped in 50 of 139 BEARING cells (36.0%), against 6.8% for the control and 9.4% for RANDOM, and 84.8% of all 1,157 cells are exact zeros.

The headline contrast. For each memory, we compare its mean paired reward gain on answer-bearing task pairs with its mean gain on matched-control task pairs. Averaged over the 60 memories, this contrast is +0.33 [+0.21, +0.46] (episode-clustered [+0.20, +0.47]; two-way over memories and tasks [+0.17, +0.47]; Figure 3a). It is similar for single-task (+0.36) and multi-task (+0.32) memories, and +0.35 [+0.17, +0.53] reweighted to all 165 bearing memories. It differs from the bearing stratum’s own mean ΔU in Table 2, which is not paired within memory. Because the screen selects pairs whose memory contains the answer, this result first validates the instrument: the protocol detects the effect it should.

The contrast holds across session, position and reader. Re-running all 285 bearing and control cells in a new session, three replicates each with the memory’s position redrawn (1,708 calls, none from cache), gives +0.50 [+0.37, +0.63]; under a second reader it is +0.60 (Section 5.5). The contrast keeps its direction in every tested condition, while its magnitude moves with session and reader, as Section 6 anticipates.

Two checks address alternative explanations. First, the screen did not select easier tasks. Because reward is binary, Table 2 and the cell counts fix each stratum’s accuracy without the memory: 11.5% for BEARING, 14.4% for the control and 20.5% for RANDOM. Bearing tasks are the hardest without the memory and the easiest with it (46.0%). The 16 bearing cells answered correctly without the memory (14 of them also with it) show that the answer is sometimes recoverable from the filler or from the reader’s priors. Second, the reader can retrieve planted content: a unique code planted inside the target memory, in the full ~27k-token context, is recovered in 22 of 22 cells and in 0 of 17 when nothing is planted (Supplement S4.3). This makes a simple inability to access the long context less likely as the cause of low accuracy on unscreened pairs.

With the answer in context, the reader still fails on 54% of bearing cells. The planted-code test makes a simple inability to access the context less likely, while failures to localize or use natural evidence remain possible. It is spread across question families (Supplement S4.4). Neither scorer leniency nor truncation explains

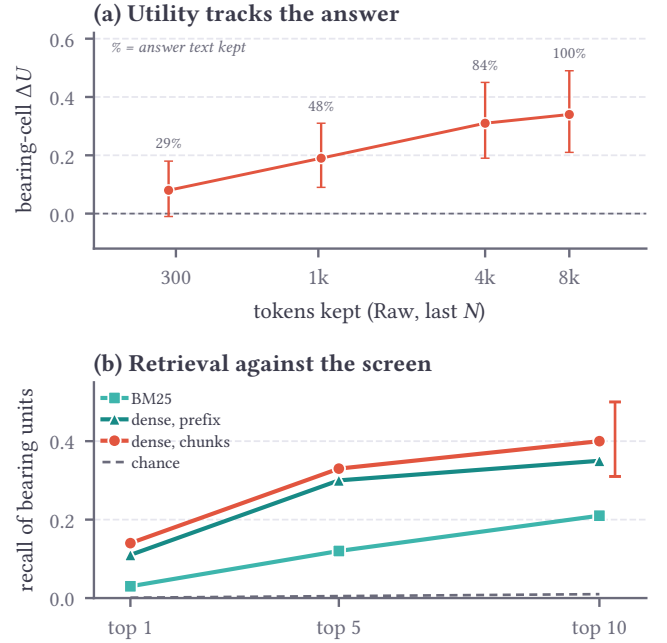


Figure 4: What carries utility (LongMemEval-V2). (a) Bearing-cell ΔU as Raw is cut to its last N tokens (135 cells, task-clustered intervals); labels give the share of cells whose text still contains the answer. (b) Share of a task’s bearing units that a retriever ranks in its top 1, 5 and 10 of the domain pool (75 tasks; task-clustered interval shown at top 10); the dashed line is chance. Both panels are exploratory; the matched-length contrast in (a) was pre-registered.

it: every bearing pair matches the gold answer exactly (57) or after normalization (82), none by loose phrase containment, and exact matches fail more often (60% against 50%). Truncated outputs are re-run, not scored (1 of 2,599 rows).

5.3 Retrievers find answer-bearing memories, but miss most

The screen defines which units of a domain pool are answer-bearing for each of the 75 tasks, so these screen-defined labels can score retrievers that never see the answer (Figure 4b). Recall@10 measures recovery of these units, not of every helpful memory or the accuracy of a complete retrieval system. Ranking the whole pool of 865 or 1,024 units by the question, dense retrieval with SFR-Embedding-Mistral ranks, on average, 0.40 [0.31, 0.50] of a task’s bearing units in its top ten (recall@10) when units are split into 512-token chunks, and 0.35 when only a 2,000-token prefix is embedded, against about 0.01 by chance. The lexical ranker BM25 reaches 0.21, with an area under the ROC curve (AUROC) of 0.86. Chunked retrieval places the first bearing unit at a mean reciprocal rank of 0.41, and chunking gains only modestly over the prefix, so truncating the embedded text costs little. Retrieval thus locates bearing memories far above chance yet misses most of them in its top ten, which is why a deployable locator remains open. Similarity is a weaker

guide to how *much* a memory helps: across executed cells, cosine similarity correlates with ΔU at only $\rho = +0.09$ (Supplement S4.2).

5.4 Evaluated compressed representations gain less than Raw

We repeat the measurement on the same experiences and tasks, changing only how each experience is written down. On the 277 cells evaluated under all three representations, the paired BEARING-control effect is +0.32 for Raw (95% CI [+0.19, +0.45]), +0.06 for Summary ([−0.04, +0.16]) and −0.05 for Structured ([−0.14, +0.04]) (Figure S5d in the supplement). Only Raw excludes zero. Per cell, Summary exposes a mean 323 tokens, Structured 238 and Raw 9,228. This Raw contrast comes from a different run and memory set than the screened one (Table 1); the intervals overlap.

The pre-registered length control. A content-blind *extractive truncation* keeps the final N tokens of the same segment, with N set per cell to that cell’s Summary length. On the 135 bearing cells (task-clustered over 55 tasks), the extractive and Summary arms differ by +0.05 [−0.07, +0.17], a failure to reject rather than an equivalence, and Raw beats the matched extractive arm by +0.26 [+0.15, +0.39]. Keeping the first N tokens instead of the last, the contrast registered as decisive, retains the bearing content in 0.7% of cells against 28.9%; last- N exceeds first- N by +0.10 [+0.01, +0.21].

A budget sweep. The same run also truncated Raw to its last 4,096 and 1,024 tokens (Figure 4a). Bearing-pair gains fall with the budget, +0.34, +0.31, +0.19 and +0.08 from full length down to matched length, and the share of cells whose text still contains the bearing content falls with it: 100%, 84%, 48% and 29%. Splitting every arm by whether its text kept that content (Figure S2b), the arms look alike: cells that kept it gain +0.32 [+0.17, +0.49] pooled over the compressed arms, Summary included at +0.25, and cells that lost it gain −0.02 [−0.08, +0.03]. The cells that kept the content are not simply easier: on the very same cells, Raw gains +0.38 where Summary kept it and +0.34 where Summary lost it. Structured is the exception, at +0.00 on 17 cells, but those are cells where even Raw gains only +0.18. Within Raw, ΔU correlates weakly with memory length across bearing cells ($r = -0.13$), which does not by itself establish the absence of a length effect. Under the evaluated task-blind writing strategies, compressed representations often omit answer-bearing text and show smaller reward gains than Raw; because the sweep changes content and length together, it is consistent with, rather than proof of, retention of answer-bearing content driving the gains. These analyses are exploratory (session caveats in Supplement S4.4). The phrase matcher can miss paraphrased answers, but little: Summary cells it calls lost still help in only 6 of 111. The result describes one task-blind writer and prompt, whose structured schema targets procedural lessons; a writer told to keep values and identifiers might keep more bearing content.

Raw still wins at a fixed total context. In deployment the main argument for compression is that a long raw memory displaces other context. We re-ran the 277 representation cells with every arm of a cell given the same token count: the compressed arms and the no-memory arm were padded with further bearing-disjoint filler (median 8,065 tokens) up to the Raw arm’s total, the memory kept at its original position. On bearing cells Raw gains +0.35

[+0.23, +0.49], while Summary gains +0.04 and Structured +0.01; against Raw on the same cells they lose −0.31 [−0.44, −0.18] and −0.34 [−0.47, −0.21]. Including cells exposed to a cache-key defect changes no conclusion (Supplement S11; Figure 3b). Costs depend on the accounting: amortizing the writer’s one-time cost over this design’s 4.86 evaluated uses per memory removes compression’s token advantage, and Summary overtakes Raw on point estimates only above about 5.5 uses per write, with a utility interval spanning zero throughout (Supplement S4.5).

5.5 The contrast keeps its direction under a second reader

One concern is that the Raw effect is specific to one reader. We re-executed the 285 Raw bearing and control cells with a second, unrelated reader (gemma-4-31B-it) on the same memory text. The memory file, prompt, filler, every rendered request and each target position were hash-asserted against the first run, and the text re-verified byte-identical on 1,694 rows. Because this endpoint is not deterministic across sessions (Section 6), we ran both readers in one session and quote that measurement.

The aggregate contrast keeps its direction and is larger: over 275 matched cells on 123 tasks, the second reader’s mean ΔU is higher by +0.10 [+0.02, +0.17], and the within-memory BEARING-control contrast is +0.60 (Figure 3a). Part of the gain comes from a weaker no-memory arm under the second reader (−0.07) rather than from a better memory.

The aggregate contrast retains its direction across the two readers, while per-pair labels show partial agreement. Over the 275 cells, the two readers’ three-way labels (helped, unchanged, hurt) agree at Cohen’s $\kappa = 0.58$. Of the 105 cells that move under either reader, both mark 55 as helped (Figure S5f). Too few cells are harmed (6 and 2) to tell whether harm labels reproduce. Whether a reader favors memories its own model wrote is tested by a pre-registered writer $\times$ reader factorial crossing both models as writers and readers. On Summary and Structured, whose utility is already near zero, the factorial excludes writer and same-family effects above 0.038 (Supplement S5).

5.6 What the instrument did not find

The study was designed to measure sign reversal: one memory helping one task and harming another. Two naive comparisons mislead. Flip rates depend on cells per memory (2.3 for BEARING, 14.5 for RANDOM): at matched counts and no-memory outcomes, RANDOM also shows no flip in 70% of resamples. And harm shares depend on baseline accuracy, since a memory can only hurt where the reader was right without it. Conditioning on that opportunity (Figure 5b; Supplement S7), answer-bearing pairs hurt in 12.5% [3.5, 36.0] of the 16 baseline-correct pairs where they could, random-task pairs in 14.5% [10.1, 20.4]: *we detect no excess harm*, but with 16 pairs the design could detect only a bearing harm rate of 44% or more, so this does not establish the absence of harm. Matched-control pairs hurt in 28.6% [13.8, 50.0], an interval that overlaps random’s. The strata differ in help: 40.7% [32.4, 49.5] of the cells where the reader was wrong, against 11.8% for RANDOM. The planned variance decomposition is not identifiable in the main run (exploratory estimate in Supplement S11).

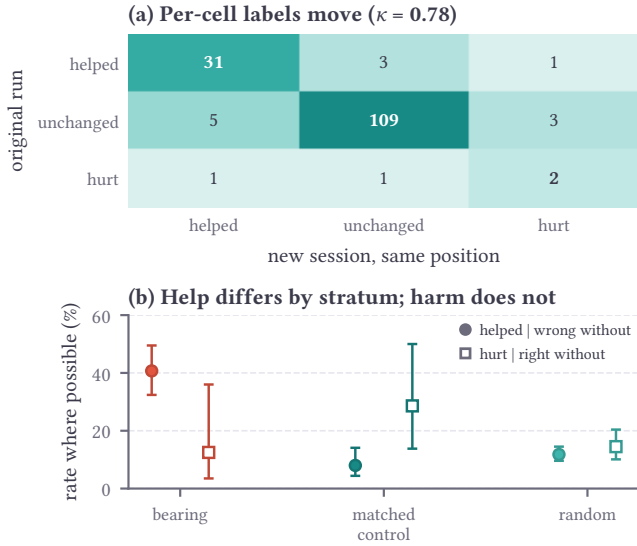


Figure 5: What single cells can and cannot show (exploratory). (a) Per-cell labels (helped, unchanged, hurt) in the original run against a new session at the same memory position (156 replicate cells; diagonal agrees). At a new position κ falls to 0.65, and between two readers it is 0.58. (b) Help among cells the reader got wrong without the memory (filled) and harm among cells it got right (hollow), by stratum, Wilson 95% intervals: help differs sharply, harm does not detectably.

6 Discussion and Limitations

Our measurements inform two memory-manager decisions without implementing either: what to keep after an episode and what to show at each later step. *Retention* would require combining task-conditioned utility profiles with an anticipated future-task distribution and memory costs; our measurements characterize a defined evaluation task set, not a deployed agent’s future return. *Exposure* is the standard retrieval problem: generic retrieval recovers screen-defined answer-bearing memories far above chance but usually not in its top ten (Section 5.3), so a reliable locator remains open. For representation, the evaluated compressed representations retain less answer-bearing text and gain less than Raw, including when total context is fixed.

The pairing is constructed. The memories are real trajectory spans, but their pairing to tasks is a seeded rule fixed before measurement, so every rate is over a defined task set, never a natural base rate, and we report no harmful-memory rate.

Session and position variance. Re-issuing 1,124 byte-identical no-memory requests a day later changed 82 rewards, so we compare only arms executed in one session. We then measured both sources, session and position, directly by re-running the headline cells in a new session with position redrawn per replicate (Section 5.2). The aggregate contrast keeps its direction. Single cells move (Figure 5a). Position alone changes 10% of labels and accounts for 11% of the variance of ΔU , and the bearing effect is weaker in the middle of the context (+0.30) than at its ends (+0.43; Figure 6), echoing [10].

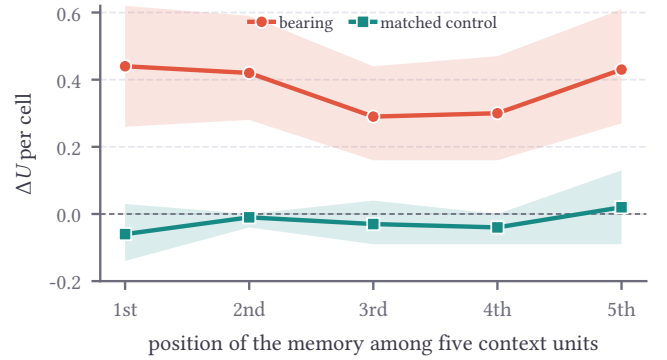


Figure 6: Position affects per-pair measurements; the aggregate bearing-control contrast remains positive after position standardization (exploratory). Per-cell ΔU by the memory’s position among the five context units in the second-session run, with task-clustered 95% bands: the bearing effect dips in the middle and recovers at the end, while the control stays near zero at every position.

Labels agree with the original run at $\kappa = 0.78$ at the same position and 0.65 at a new one, and changes almost always move to or from *unchanged* (Figure 5a). Standardizing the published contrast over position leaves it at +0.33.

Scope. The results concern a single-call factual-recall reader, not acting agents. Every cell runs on one serving cluster and one benchmark, in two domains, and two reader checkpoints establish nothing about model families. Single-item insertion cannot see redundancy or complementarity between units, which Shapley-style attribution could [20]. Our filler comes from trajectories that host no bearing memory, so it never contains competing evidence; realistic retrieval would, which likely makes our bearing effect larger than a deployed system would see. Our leakage-audit rates are lower bounds (Supplement S9).

Release and reproducibility. The Conditional Memory Utility Dataset is publicly available on Hugging Face. The frozen v0.6-length-control snapshot contains 16,816 evaluation records, with paired outcomes, costs and available text, plus schema and validation code. The data card documents the included conditions, missing records and text withheld under upstream restrictions. This versioned snapshot does not cover every later manuscript analysis, including the fixed-total-context experiment. Per-pair labels depend on the evaluation condition; they are observations, not unconditional targets for training a memory gate. See Supplement S8 for coverage and reuse details.

Conclusion. On LongMemEval-V2 factual-recall tasks, answer-bearing pairs gain more than matched controls, and the evaluated compressed representations gain less, consistent with reduced retention of relevant content. The headline contrast stays positive across the tested sessions, positions and readers, while individual labels remain condition-dependent. Memory should be evaluated through task-conditioned effects, content preservation and measurement sensitivity.

Use of AI-Assisted Technologies

We follow the stronger clause of the AAMAS policy. AI assistants helped with the literature review, coding, analysis, and paper writing under human guidance. The authors made the research decisions, interpreted the results, verified all numbers, and checked all citations.

Acknowledgments

This research used resources of the Argonne Leadership Computing Facility, a U.S. Department of Energy (DOE) Office of Science user facility at Argonne National Laboratory (ANL) operated under Contract No. DE-AC02-06CH11357. The work was also supported under the same contract by the DOE Office of Science’s Advanced Scientific Computing Research Program and by Laboratory Directed Research and Development (LDRD) funding from ANL, provided by the Director, DOE Office of Science. The work of the University of Houston coauthors was supported in part by the National Science Foundation under grants CAREER-2305491 and CNS-2504762.

References

- [1] Anonymous. 2026. EvoMemEval: Evaluating Large Language Models at Evaluating Memory Utility in Self-Evolving Agents. ACL ARR 2026 May Submission, OpenReview n3GMb0bXnc. Under review.
- [2] Zhibao Chen and Qian Cheng. 2026. Learning What to Remember: A Cognitively Grounded Multi-Factor Value Model for Agentic Memory. arXiv:2606.12945 [cs.CL]
- [3] Benjamin Cohen-Wang, Harshay Shah, Kristian Georgiev, and Aleksander Mądry. 2024. ContextCite: Attributing Model Generation to Context. In *Advances in Neural Information Processing Systems (NeurIPS)*, Vol. 37. doi:10.52202/079017-3035
- [4] Florin Cuconasu, Giovanni Trappolini, Federico Siciliano, Simone Filice, Cesare Campagnano, Yoelle Maarek, Nicola Tonellotto, and Fabrizio Silvestri. 2024. The Power of Noise: Redefining Retrieval for RAG Systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, USA, 719–729. doi:10.1145/3626772.3657834
- [5] Jiale Deng, Yanyan Shen, Ziyuan Pei, Youmin Chen, and Linpeng Huang. 2025. Influence Guided Context Selection for Effective Retrieval-Augmented Generation. arXiv:2509.21359 [cs.CL]
- [6] Amirata Ghorbani and James Zou. 2019. Data Shapley: Equitable Valuation of Data for Machine Learning. In *Proceedings of the 36th International Conference on Machine Learning (ICML) (PMLR, Vol. 97)*. PMLR, 2242–2251.
- [7] Horace He and Thinking Machines Lab. 2025. Defeating Nondeterminism in LLM Inference. Thinking Machines Lab: Connectionism (blog). <https://thinkingmachines.ai/blog/defeating-nondeterminism-in-llm-inference/>
- [8] Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2024. SWE-bench: Can Language Models Resolve Real-World GitHub Issues? In *International Conference on Learning Representations (ICLR)*.
- [9] Yilong Li, Suman Banerjee, and Tong Che. 2026. EMBER: Efficient Memory via Budgeted Evidence Retention for Long-Horizon Agents. arXiv:2606.05894 [cs.CL]
- [10] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranajpe, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the Middle: How Language Models Use Long Contexts. *Transactions of the Association for Computational Linguistics* 12 (2024), 157–173. doi:10.1162/tacl_a_00638
- [11] Pranath Reddy. 2026. Selective Memory Retention for Long-Horizon LLM Agents. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*. arXiv:2606.29178.
- [12] Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H. Chi, Nathanael Schärli, and Denny Zhou. 2023. Large Language Models Can Be Easily Distracted by Irrelevant Context. In *Proceedings of the 40th International Conference on Machine Learning (ICML) (PMLR, Vol. 202)*. PMLR, 31210–31227.
- [13] Chung-En Sun, Linbo Liu, Ge Yan, Zimo Wang, and Tsui-Wei Weng. 2026. LLM Agents Already Know When to Call Tools – Even Without Reasoning. arXiv:2605.09252 [cs.CL]
- [14] Mohammad Tavakoli, Alireza Salemi, Carrie Ye, Mohamed Abdalla, Hamed Zamani, and J Ross Mitchell. 2026. Beyond a Million Tokens: Benchmarking and Enhancing Long-Term Memory in LLMs. In *International Conference on Learning Representations (ICLR)*. arXiv:2510.27246.
- [15] Minyang Tian, Luyu Gao, Shizhuo Dylan Zhang, Xinan Chen, Cunwei Fan, Xuefei Guo, Roland Haas, Pan Ji, Kittithat Krongchon, Yao Li, Shengyan Liu, Di Luo, Yutao Ma, Hao Tong, Kha Trinh, Chenyu Tian, Zihan Wang, Bohao Wu, Yanyu Xiong, Shengzhu Yin, Minhui Zhu, Kilian Lieret, Yanxin Lu, Genglin Liu, Yufeng Du, Tianhua Tao, Ofir Press, Jamie Callan, Eliu Huerta, and Hao Peng. 2024. SciCode: A Research Coding Benchmark Curated by Scientists. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*. arXiv:2407.13168.
- [16] Di Wu, Zixiang Ji, Asmi Kawatkar, Bryan Kwan, Jia-Chen Gu, Nanyun Peng, and Kai-Wei Chang. 2026. LongMemEval-V2: Evaluating Long-Term Agent Memory Toward Experienced Colleagues. arXiv:2605.12493 [cs.CL]
- [17] Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. 2025. LongMemEval: Benchmarking Chat Assistants on Long-Term Interactive Memory. In *International Conference on Learning Representations (ICLR)*. arXiv:2410.10813.
- [18] Sikuan Yan, Ahmed Bahloul, Ercong Nie, Susanna Schwarzmann, Riccardo Trivisonno, Volker Tresp, and Yunpu Ma. 2026. Memory-R2: Fair Credit Assignment for Long-Horizon Memory-Augmented LLM Agents. arXiv:2605.21768 [cs.CL]
- [19] Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Jinhe Bi, Kristian Kersting, Jeff Z. Pan, Hinrich Schütze, Volker Tresp, and Yunpu Ma. 2025. Memory-R1: Enhancing Large Language Model Agents to Manage and Utilize Memories via Reinforcement Learning. arXiv:2508.19828 [cs.CL]
- [20] Boyu Yang, Jiazheng Sun, Zilong Lu, Zhi Qiu, Xin Peng, and Jun Zheng. 2026. MeClear: Cooperative Game-Theoretic Attribution and Risk-Aware Memory Clearance for Long-Horizon LLM Agents. arXiv:2609.09115 [cs.CL]
- [21] Yongshi Ye, Hui Jiang, Feihu Jiang, Tian Lan, Yichao Du, Biao Fu, Xiaodong Shi, Qianghuai Jia, Longyue Wang, and Weihua Luo. 2026. UMEM: Unified Memory Extraction and Management Framework for Generalizable Memory. arXiv:2602.10652 [cs.CL]

Supplementary Material

Guide to this supplement. The paper is self-contained, and this document holds the detail behind it. Section S1 gives the full protocol: how the writer is kept blind to future tasks, how target position is controlled, both gold-free screens, and the leakage audit. Section S2 describes the benchmark adapters, the scorer, pairing and power, and why each secondary benchmark has the yield it has. Section S3 lists the frozen execution configuration, the noise floors, the label definitions, and why the additive null and the variance decomposition matter. Section S4 adds LongMemEval-V2 results: the cell-level sparsity map, the reader capability check, the length-control design, and the formation-cost sensitivity. Section S5 reports the second-reader study and the writer-by-reader factorial in full, and Section S6 gives the cross-benchmark contrasts as a table. Section S7 reports the sign-reversal analysis the study was designed around, including why naive sign comparisons mislead and the transition analysis that replaces them. Section S8 covers data, code and prompt availability, licensing and reproducibility, and Section S9 the limitations not needed to read the main result. Section S10 is the pre-registration record: every pre-registered analysis, its anchor, and every recorded deviation. Section S11 reports three follow-up runs: a second session with randomized position, a fixed total context, and a retrieval evaluation. Every table and figure states its benchmark, population and denominator.

S1 Protocol details

S1.1 Keeping the writer blind to future tasks

Memory formation must be a function of the past episode alone, $m = W(e)$. If the writer could see a future task, a positive ΔU would show only that a model can be told the answer and copy it forward. We enforce this with four structural barriers rather than by convention. (1) *Arity*: the writer function takes a segment and nothing else. (2) *A closed field set*: the writer emits only the fields of its schema. (3) *A metadata denylist*: the weakest barrier, present to make a plumbing mistake fail loudly. (4) *A single interpolation point*: the prompt has one placeholder, the segment text. Each barrier carries a negative control that plants the defect and requires the check to fire. Prompt identity is derived from the prompt file’s bytes, not from a hand-maintained version constant that could report a stale version.

S1.2 Target position is held fixed

The WITH arm holds one more item than the WITHOUT arm, so some rule must place it, and that rule is not neutral. Holding the memory, question, filler, model, decoding and seed fixed and varying only the position of the target memory changes the answer in 8 of 12 probe cells. Measured at scale in the second-session run (Section S11), position changes 10% of labels. A comparison that does not control position therefore measures position together with presence. We hold position fixed across replicates, vary it across cells, and record it on every row. This is also what makes the second-reader comparison (main paper, Section 5.5) a comparison of readers and not of positions. The probe is small ($n=12$), and we treat the rate as provisional.

S1.3 The bearing screen and both gold-free screens

The bearing screen reads the memory text, the task’s gold answer, and the task’s evaluation specification; it never reads the question. A pair is BEARING only when the benchmark’s scorer marks the memory text correct against the gold answer *and* the test fires on at most eight of the 865–1,024 units in the task’s domain pool. The second clause matters. An earlier, non-discriminative version of the flag fired on nearly every candidate: its rate on paired versus unpaired memories was in a ratio of 0.995, so it said nothing about any one of them.

We pre-registered two gold-free screens before execution. Each reads the memory, the question and the same-domain pool, and none of the gold answer, the reward, any model output, or the evaluation specification. We report both, because reporting only the screen that worked would select an instrument on its outcome.

- *Model-based screen.* The judge is *gpt-oss-120b*, the reader’s own model, at temperature 0, asked whether the memory contains the specific information the question asks about. On cells the gold screen never selected, its picks move the reader from 15.6% correct without the memory to 50.0% with it (64 rows, 32 cells); its matched control moves from 14.3% to 11.9% (84 rows, 42 cells). The contrast of +0.3676 has a task-clustered interval of [+0.05, +0.70] (44 tasks), an episode-clustered one of [+0.14, +0.67] (19 episodes), and a paired-by-memory difference of +0.49 over 24 memories. Against the gold screen it recovers 10 of 266 gold-bearing pairs (3.8% recall), and 10 of its 42 picks are gold-bearing (23.8%). The run is in one session, separate from the gold run.
- *Lexical screen.* It returns no effect. A diagnostic independent of any utility explains why: its bearing cells fail in both arms at a rate of 0.744, worse than unscreened random pairing, against 0.310 for the model-based screen. A criterion whose selections are harder than a random draw has not located bearing pairs.

S1.4 Leakage audit

The writer invariant excludes *task-directed writing*, not a naturally occurring memory that happens to contain a later answer. Such a memory is genuinely useful, and removing it would condition the cell set on the outcome. We therefore audit every cell rather than filter it. The detector reuses the benchmark’s own scorer and asks whether the rendered memory text alone would score correct against the gold answer. For BEARING cells the answer is yes by construction, since that is half the bearing definition. The audit carries information for the control and RANDOM strata and for the Summary and Structured texts, where containment is not guaranteed and can be conditioned on in analysis. It is validated by 17 controls (plants it must catch, clean cases it must not fire on, and two pinned known false negatives) and aborts if any control fails.

S1.5 Three kinds of zero in the cost ledger

Raw makes zero writer calls; Summary and Structured make one each per segment. The ledger distinguishes three zeros that are routinely merged: a measured zero (Raw makes zero extraction

calls), an absent measurement (no call was made, so the cell leaves that denominator), and an unreported field (the endpoint exposes no counter). Failed writes count: a write that failed after consuming a call still cost that call.

S2 Benchmark adapters and coverage

LongMemEval-V2 haystacks. On the small tier each domain has exactly one haystack, identical across questions even as an ordered tuple, so every memory is available to every question in its domain and there is no availability confound between arms. The same test returns 222 and 211 distinct haystacks on medium, a positive control showing that it distinguishes the two cases. Trajectory lengths were measured with the benchmark’s own tokenizer over 200 trajectories and 5,095 states.

Scorer dispatch. There are eight distinct evaluation-specification strings, not six: two phrase-match variants differ only in their separator. The adapter therefore dispatches on the full string, never on the function name, and calls the benchmark’s scorer verbatim. Its phrase matcher tests gold-phrase *containment*, not set equality, which a name-based reimplementation gets wrong. An LLM-judged task raises an error rather than returning a default; a silent False would turn “cannot evaluate” into “wrong answer” for 156 of 451 questions in both arms.

Pairing and power. Seeded deterministic pairing within domain, stratified by the benchmark’s own question-type taxonomy, gives 1,889 memories $\times$ 8 tasks = 15,112 cells, and every memory was verified to receive exactly 8. The build asserts that the same seed reproduces the grid byte-identically *and* that a different seed changes it, so the reproducibility check is not vacuous. That grid is the available pairing, not the executed design. The floor of 60 memories was fixed before running from a recovery study on synthetic data with known planted components; the same study showed that widening the task axis helps more than adding memories.

BEAM. BEAM [14] was in our original plan as a multi-domain check but cannot supply the task axis: 92.6% of its cited turns are cited by exactly one question. Being CC BY-SA 4.0, it would also have to ship separately, since ShareAlike would otherwise relicense the rest of the release.

SciCode. The memory unit is a reusable helper routine with its procedure, which transfers across problems. Against an early feasibility estimate of 8–12 pairs, the full candidate pool measured on the built adapter is 154 donor→recipient pairs (34 memories, 40 tasks), and we ran all of it.

SWE-bench Verified. Pairing is plentiful but the *measurable* yield is low. Across 2 repositories, after a localizer that misses the edit region on 38 cells and edits that neither apply nor produce a test verdict, 40 cells are interpretable under all three representations, and 92% come from one repository. A scoring defect was corrected before these counts: a test suite that printed no verdict at all (an import or collection error) had been recorded as a reward of zero. Those 37 arm-reads (27 django, 10 sympy) are now unscorable, and every contrast includes zero after the correction. The matched set of 40 cells is pooled from four runs across the two repositories; it grew as runs were added, not through the correction, which can

only remove cells. Table 1 of the main paper counts SWE-bench in (memory, task, representation) cells; counts per recipient task (107/37/70) describe a different unit.

S3 Execution configuration and statistics

Frozen configuration. Cluster and serving configuration are part of a cell’s experimental identity, and every row records the full pinned set: reader and writer gpt-oss-120b on one cluster, low reasoning effort, temperature 0, max_tokens 512, 16 concurrent workers, SFR-Embedding-Mistral at dimension 4096, and the benchmark’s own tokenizer. A change to any of them defines a new arm rather than additional data. Three settings were fixed by measurement. An empty or truncated completion raises and the cell is re-run, since scoring it as zero would push ΔU negative and manufacture harmful memories. Concurrency sits at a measured saturation ceiling, with the ladder pushed past the chosen value. Responses are cached on all twelve fields of a cell’s identity, verified by varying one field and confirming a cache miss. The embedding model’s determinism was checked directly: the cosine of identical texts is exactly 1.0.

Noise floors. F_{exec} is outcome variation across seeds with everything else fixed, measured per benchmark with a bootstrap interval and within a single execution session. F_{empty} is the variation of the no-memory arm, available at no extra cost because that arm is identical for every memory paired with a task. F_{judge} is scorer disagreement on byte-identical outputs, from re-scoring stored generations; we store full output text rather than a digest so that this re-scoring is possible. The floor is defined on the ΔU scale, since $\text{var}(\Delta U) = \text{var}_{\text{with}} + \text{var}_{\text{without}} - 2 \text{cov}$; per-arm floors are reported alongside. One hazard would fabricate the whole result: the WITH-OUT arm’s inputs are identical across memories, so a cache serving it would reuse one generation everywhere and let every ΔU clear a zero floor. That arm is re-generated per (task, replicate) rather than served from the response cache; in the screened run all 5,198 calls reached the endpoint, with 0 cache hits.

Labels and flips. A cell is robust useful when $\text{CI}_{\text{lower}} > 0$ and $\overline{\Delta U} > F$, robust harmful when $\text{CI}_{\text{upper}} < 0$ and $|\overline{\Delta U}| > F$, and neutral otherwise, so a cell can be statistically non-zero and still neutral. A memory shows a robust sign flip only if two of its cells independently meet the useful and harmful criteria. Every flip rate carries its denominator (memories on at least two tasks) and a Wilson interval beside the naive rate, and a cell needs at least two replicates to enter one. Because replicates agree on every primary cell, a robust label reduces in practice to the sign of ΔU , and flip rates are comparable only at matched cells per memory (Section S7).

The additive null. A flip is a property of ΔU ’s sign; interaction is a property of its variance. Under $\Delta U(m, t) = \mu + a_m + b_t$ with no interaction, a memory still helps one task and harms another whenever b_t crosses $-(\mu + a_m)$. Simulating that null over 20 trials of 200 memories $\times$ 8 tasks gives flip rates of 53.5%–92.0% across five settings of μ and the memory and task standard deviations. A high flip rate is therefore consistent with memory value being a scalar plus a task-difficulty term.

The variance decomposition. We pre-registered a decomposition $\Delta U(m, t, s) = \mu + a_m + b_t + (ab)_{mt} + \varepsilon_s$ into component variance shares against F_{exec} , with a method-of-moments fallback fixed in advance. We recorded the structural precondition before executing: components are identified only relative to a residual scale. It fails here because the reader is deterministic within a session, so we quote no share (Section S7). Standard errors are clustered by task throughout.

Similarity and cost. Semantic distance is the cosine of the pinned embeddings of memory and task, banded by terciles fixed before outcomes were examined, since cosine scales are not comparable across corpora. The seven cost families are **storage** (memory tokens held), **retrieval** (latency, tokens returned), **exposure** (tokens placed in the reader’s context), **generation** (latency, input, output and reasoning tokens), **extraction** (formation cost), **API and compute**, and **selectivity** (how much of what was retrieved was worth exposing). Exposure is recorded separately from retrieval because the gap between them is truncation loss.

S4 Additional LongMemEval-V2 results

S4.1 The cell-level sparsity map

Figure S1 draws every one of the 1,157 measured cells of the screened design. It is the cell-level view behind Figure S5b and Table 2 of the main paper. ΔU takes exactly three values here, because the reward is binary and both replicates agree on every cell, so the marks are discrete rather than shaded. Unmeasured slots are left blank: rows are ragged (at most 9, 9 and 15 cells per memory in the three blocks), and padding them to a rectangle would spread the same marks over more area. Per drawn square the BEARING rate is 36.0% against 9.4% for RANDOM; per padded area the two would look the same, which is why the figure does not pad.

S4.2 Embedding similarity in detail

This section holds the detail behind main paper, Section 5.3. Cosine similarity between memory and task, under the pinned embedding model, is nearly uninformative about ΔU : Spearman $\rho = +0.087$ over all 1,157 cells and $\rho = +0.006$ on unscreened cells alone. The pooled series rises slightly at high similarity partly because BEARING cells concentrate there, and within BEARING, $\rho = +0.164$.

This result is narrower than it looks, for two reasons. First, 96.6% of the similarities are computed from a truncated prefix of the memory, and bearing content sits mostly near the end of a segment: the first N tokens retain it in 0.7% of cells against 28.9% for the last N (main paper, Section 5.4). The embedding therefore mostly never saw the content that matters. Second, Spearman ρ over an outcome that is exactly zero in 84.8% of cells is depressed by ties and is not what a retriever needs, which is to rank the bearing memory first. We therefore claim only that this embedding configuration does not track utility, not that embedding similarity cannot.

S4.3 The reader can use the memory

We planted a per-cell unique code inside the target memory, in the real $\sim 27k$ -token context, and asked the reader for it. The base reader recovers it in 22 of 22 cells and in 0 of 17 when nothing is planted. Two larger models also recover 22 of 22 and solve no cell the base



Figure S1: Utility is sparse over future tasks, and the sparsity is structured. One square per *measured* cell — all 1,157 of them, 60 memories $\times$ their own future tasks, one representation and one frozen reader. ΔU takes exactly three values here, so the marks are discrete rather than shaded. Only 176 cells move at all: 84.8% of measured cells are an exact zero, which is the quantity a single global memory score has to summarize. The three blocks are the same 60 memories against bearing, matched-control and unscreened tasks, so the comparison is paired within memory. Rows are ordered by BEARING count; that arranges the rows but cannot create the block contrast, which is 36.0% against 9.4% with disjoint Wilson intervals. A row’s squares are that memory’s own tasks in its own order, so a *column is not a task*: the blocks span 56, 100 and 147 distinct tasks against widths of 9, 9 and 15. Unrun cells are blank rather than padded — the rows are ragged, and padding them would spread the same marks over more area and collapse the contrast to about 1 $\times$.

reader missed. The reader can locate and use memory content at full context length, so low accuracy on unscreened cells is not a capability ceiling. In the two earlier unscreened arms, accuracy also rises monotonically with how discriminatively the memory matches the task: 0.1429, 0.4545, 0.5000.

Earlier unscreened arms. The same contrast appears in two earlier, unscreened prompt arms, where bearing was identified afterwards from content alone. On the 22 and 22 bearing cells of the two arms, mean ΔU is $+0.0909$ and $+0.1818$ (2 and 4 net helped cells), against -0.0066 and -0.0153 elsewhere; pooled, the difference is $+0.1473$ (permutation $p = 0.0029$ over 20,000 shuffles). These are small samples, and we read them only as agreeing in sign with the screened design.

S4.4 Length control and budget sweep

The length control was frozen before execution. Its extractive arm keeps the final N tokens of the same source segment, with N set

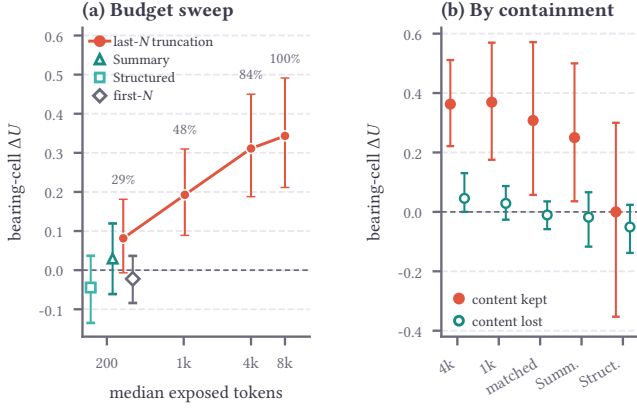


Figure S2: Length-control run on LongMemEval-V2: ΔU on the 135 bearing cells, task-clustered 95% intervals. (a) Raw truncated to its last 4,096 and 1,024 tokens and to each cell’s Summary length (line), against Summary, Structured and the first- N window at that length; percentages give the share of cells whose text still contains the bearing content. (b) The same arms split by whether their text kept the bearing content (filled) or lost it (hollow).

Table S1: Length-control run on LongMemEval-V2: bearing-cell ΔU by arm (135 cells, task-clustered 95% intervals), share of cells whose text still contains the bearing content, and ΔU split by whether it does. Exploratory beyond the three pre-registered contrasts.

arm	median tokens	kept	ΔU	ΔU kept	ΔU lost
Raw (full)	8,344	100%	+0.34 [+0.21, +0.49]	+0.34	–
last 4,096	4,096	84%	+0.31 [+0.19, +0.45]	+0.36	+0.05
last 1,024	1,024	48%	+0.19 [+0.09, +0.31]	+0.37	+0.03
last N (matched)	282	29%	+0.08 [-0.01, +0.18]	+0.31	-0.01
Summary	282	18%	+0.03 [-0.06, +0.12]	+0.25	-0.02
Structured	216	13%	-0.04 [-0.13, +0.04]	+0.00	-0.05
first N (matched)	282	1%	-0.02 [-0.08, +0.04]	–	-0.02

per cell to that cell’s own Summary token count, so it matches Summary’s budget without any model choosing content; a second arm keeps the first N tokens instead, and last- N versus first- N was registered as the decisive contrast. All three pre-registered contrasts are computed on the 135 bearing cells (two replicate rows each, 270 rows) and clustered over 55 tasks: extractive versus Summary (+0.0519 [-0.0678, +0.1727]), Raw versus matched extractive (+0.2612 [+0.1533, +0.3864]), and last- N versus first- N (+0.1037 [+0.0091, +0.2098]).

The same run truncated Raw to its last 4,096, 1,024 and 20,000 tokens; the last equals Raw, since no segment is longer. Table S1 gives each arm’s bearing-cell ΔU , its retention of the bearing content, and ΔU split by retention. `make_lenc1.py` computes every entry from the row file, whose checksum it verifies against the run manifest, and first checks that it reproduces the three pre-registered contrasts. The run is not strictly one session: 14 responses came from a cache filled minutes earlier, and 6 rows with a null reward are excluded, never zeroed.

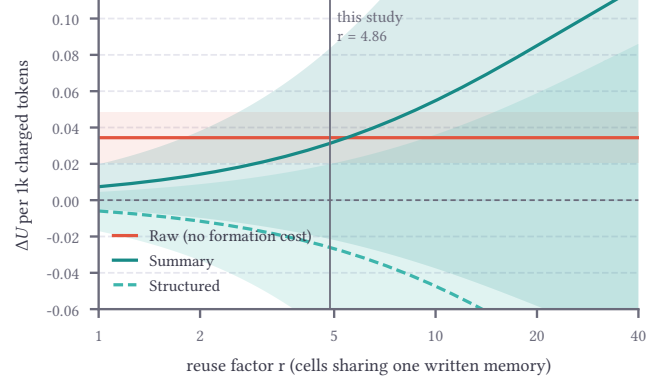


Figure S3: Exposure efficiency (ΔU per thousand charged tokens) against the reuse factor, on the 277 representation cells of LongMemEval-V2. Lines use each representation’s paired BEARING–control effect; bands use its 95% interval over memories. Raw has no writer call and does not depend on r . The vertical line marks the reuse factor of this study.

Is the kept subset easier? Retention is a property of the text, fixed before any outcome, but the cells where a compressed text keeps the content could still be easier ones. They are not for Summary or the matched truncation: on the same cells, Raw gains +0.38 where Summary kept the content and +0.34 where it lost it (+0.33 and +0.35 for the matched truncation). For Structured the kept cells are *harder* (+0.18 against +0.37), which accounts for its low kept estimate.

Position. Target position is identical across all eight arms within each cell (asserted). Between strata its distribution differs mildly ($\chi^2 p = 0.09$); the position-standardized Raw BEARING–control contrast is +0.39 against +0.39 unadjusted, and positive at every position.

S4.5 Formation-cost sensitivity

Formation cost per cell is the writer’s tokens divided by the reuse factor r , the number of cells that share one written memory. Our design fixes $r = 4.86$, so we recompute exposure efficiency over r from the same utilities and token counts (Figure S3). One Summary costs about 7,956 writer tokens and one Structured record about 8,141. On point estimates Summary overtakes Raw only for $r \geq 5.5$, but its utility interval spans zero at every r , so the crossing point is not estimated with any precision. Under the sparsity we measure, a memory that bears on one task may be shown for few others, which pushes r down and formation cost up. The relevant factor in deployment is exposures per write, not cells per memory. Input and output tokens are priced identically and prompt caching is ignored; since output tokens usually cost more, both choices understate the writer’s cost.

S5 Second reader and writer, $\times$, reader factorial

Design. The second-reader study was pre-registered before any design-cell call was issued, with no transfer, the same ordering at a different magnitude, and a different representation preference all committed as valid outcomes. The memory file, prompt, filler pool,

every rendered request, and each cell’s target position were hash-asserted against the first run at call time, and the text re-verified byte-identical on 1,694 rows; in the session-matched run all 570 of 570 joinable rows are hash-identical.

Session-matched result. Both readers ran in one session. Over 275 matched Raw cells on 123 tasks (550 replicate rows; these three counts are different units), mean ΔU is +0.2182 under the first reader and +0.3164 under the second, a paired change of +0.0982 [+0.0247, +0.1730]. The two readers’ own 95% intervals overlap; plotting those would invite a “no difference” reading that the paired design rules out. The within-memory BEARING-control contrast is +0.3462 [+0.2210, +0.4761] under the first reader and +0.6002 [+0.4787, +0.7144] under the second (59 memories). The no-memory arm moves by -0.0688 , so part of the gain is a weaker reader rather than a better memory.

Cross-session first execution. A first execution with the two readers a day apart returned +0.1268 [+0.0496, +0.2050], inside the session-matched interval. Re-run together, that difference splits into -0.0327 attributable to the session and -0.0364 to the reader, which halves the apparent reader share. We quote only the session-matched estimate.

Cell-level agreement, session-matched run. These are the numbers behind Figure S5f. Over the 275 matched Raw cells, the readers’ three-way labels (helped, unchanged, hurt) agree on 225 of 275 cells, at Cohen’s $\kappa = 0.58$; 170 cells are unchanged under both and can only agree. Of the 105 cells that move under either reader, both mark 55 as helped, and signs agree on 55 of the 58 cells that move under both. The readers agree on no harmed cell (the first reader’s 6 and the second’s 2 do not overlap), but so few cells are harmed that this cannot show whether harm labels reproduce.

Cell-level agreement, earlier cross-session run. The first execution, with the two readers a day apart, covers a slightly different matched set (276 cells) and is the only source for Summary and Structured. There the readers’ per-cell ΔU correlate at $r = +0.520$ over all cells, of which 167 are zero under both; over the 109 cells non-zero under either reader, $r = +0.213$ (+0.187 for Summary, -0.070 for Structured). Signs agree on 49 of 50 Raw cells where both move, and labels on 216 of 276. Summary +0.0036 [-0.0484 , +0.0612] and Structured +0.0112 [-0.0536 , +0.0833] span zero with only 36 and 35 cells non-zero under either reader. Because this run spans sessions, its numbers mix reader and session differences, and they should not be pooled with the session-matched ones. Costs are not pooled across readers: the second reports no reasoning-token counts.

Writer, $\times$, reader factorial. All four writer-reader arms ran in one session. Every one of the eight contrasts (writer, reader, same-family and interaction, on Summary with $n=274$ and Structured with $n=268$) spans zero. The intervals are informative rather than underpowered: they exclude writer and same-family effects larger than 0.0380, against the +0.0982 reader effect the same apparatus detects. Raw has no writer and is not in the factorial, and two checkpoints form a 2×2 design, not a population of models.

Table S2: Per-cell representation contrasts: ΔU under one representation minus ΔU under Raw on the same memory and task, with task-clustered 95% intervals. Only LongMemEval-V2 separates from zero. Every SciCode and SWE-bench interval includes zero (SWE-bench’s Summary–Raw only at its upper bound), and doubling the SciCode cell count separated no contrast. MDE is the minimum detectable contrast at 80% power and two-sided $\alpha = 0.05$, computed from each interval’s width. Raw’s own with-versus-without ΔU is -0.01 [-0.05 , +0.02] on SciCode and -0.05 on SWE-bench, so on these benchmarks no memory effect exists for a representation to preserve, and the contrasts cannot test the ordering.

contrast	benchmark	<i>n</i>	mean	95% CI	MDE
Summary–Raw	LongMemEval-V2	277	-0.1408	[-0.2211, -0.0601]	0.12
	SciCode	122	+0.0246	[-0.0342, +0.0991]	0.10
	SWE-bench	40	-0.1000	[-0.2250, +0.0000]	0.16
Structured–Raw	LongMemEval-V2	277	-0.1625	[-0.2454, -0.0798]	0.12
	SciCode	122	+0.0328	[-0.0342, +0.1200]	0.11
	SWE-bench	40	-0.0500	[-0.1750, +0.0750]	0.18

S6 Cross-benchmark contrasts

The one quantity all three benchmarks define identically is the per-cell representation contrast: a representation’s ΔU minus Raw’s on the same memory and task. It separates from zero on LongMemEval-V2 alone (Figure S5e): Summary–Raw is -0.1408 [-0.2211 , -0.0601] and Structured–Raw is -0.1625 [-0.2454 , -0.0798] on 277 cells. On SciCode ($n=122$) and SWE-bench Verified ($n=40$) every interval includes zero, but there the memories themselves have no effect even in raw form: Raw’s own ΔU is -0.01 [-0.05 , +0.02] on SciCode, where 122 of 127 cells are unchanged, and -0.05 [-0.17 , +0.07] on SWE-bench. With no Raw effect to preserve or lose, a representation contrast there is uninformative about representation; these benchmarks bound where memory matters for this reader, not how it should be written (Section S6).

Table S2 gives the values plotted in Figure S5e. We compare contrasts rather than per-representation ΔU because each benchmark carries its own nondeterminism. The no-memory arm, whose input is identical across representations, still disagrees with itself on $7/285=0.025$ of LongMemEval-V2 cells and $5/135=0.037$ of SciCode cells, and a paired contrast removes that term. SWE-bench has no such floor because its no-memory arm is deterministic by construction. SciCode’s equivalent judge control disagrees on 5 of 135 cells within one session, so determinism cannot be assumed of a new benchmark.

S7 Sign reversal and the decomposition

Why flip rates cannot be compared across strata. A memory can show a flip only if at least two of its cells move with opposite signs, so the chance of a flip grows with the number of cells per memory. In the screened design a memory has 2.3 BEARING cells and 14.5 RANDOM cells on average. The naive contrast (0 of 35 BEARING memories against 17 of 60 RANDOM, Fisher one-sided $p = 1.5 \times 10^{-4}$) therefore mostly reflects that difference. To test this, we took each of the 35 memories with at least two BEARING cells and drew, without replacement, as many of its RANDOM cells as it has

Table S3: Screened run on LongMemEval-V2: transitions between the no-memory and with-memory outcome (binary reward). Accuracy columns give the share of cells answered correctly without and with the memory. Hurt is conditioned on cells correct without the memory, help on cells wrong without it; Wilson 95% intervals in brackets.

stratum	n	acc. w/o	acc. with	hurt correct	help wrong
BEARING	139	11.5	46.0	12.5 [3.5, 36.0]	40.7 [32.4, 49.5]
control	146	14.4	17.1	28.6 [13.8, 50.0]	8.0 [4.4, 14.1]
RANDOM	872	20.5	26.9	14.5 [10.1, 20.4]	11.8 [9.6, 14.5]
opportunities: hurt 16 / 21 / 179 cells; help 123 / 125 / 693 cells					

BEARING cells, then recounted flips (20,000 resamples). The random stratum averages 0.43 flips and shows none in 63% of resamples (Figure S4b), so 0 of 35 is what chance alone would most often produce. A Wilson bound on the BEARING flip rate would also treat 35 memories from 21 episodes as independent. Matching each memory’s random subsample on the no-memory outcome as well (so both strata get the same opportunities to help and to hurt) leaves all 35 memories eligible; the random stratum then shows a mean of 0.33 flips and none in 70% of resamples. We therefore do not compare flip rates across strata.

Why the sign of moving cells is not a comparison either. A tempting alternative compares the share of negative cells among cells that move: 2 of 52 for BEARING against 26 of 108 for RANDOM. That comparison is confounded by baseline accuracy. A memory can only hurt where the reader was correct without it, and BEARING tasks have the lowest no-memory accuracy, so they offer the fewest chances to hurt.

Transitions conditioned on the no-memory outcome. Reward is binary in the screened run: each stratum’s mean equals (helped – hurt)/ n exactly. With Table 2’s both-arms-fail rate this fixes the full transition table (Table S3). Conditioned on opportunity, the strata do not differ detectably in harm, and they differ sharply in help (Figure S4a). In the RANDOM stratum help and harm rates are close, the signature of a content-independent response to inserting an item, which we treat as the perturbation floor (main paper, Section 3.4). The table’s Wilson intervals treat cells as independent. Resampling memories instead, the harm rates have intervals of [0.0, 31.2] (BEARING), [10.3, 50.0] (control) and [9.3, 19.9] (RANDOM), which overlap, and the help rates [31.4, 50.4], [3.7, 12.4] and [9.9, 13.8], where BEARING’s is disjoint from the others. Both analyses here are exploratory and are produced by `make_reanalysis.py`, which first checks that it reproduces every flip count and stratum mean in the paper.

The additive-null comparison. Fitting $\mu + a_m + b_t$ and simulating from it, the excess of the observed reversal rate over the fitted additive model is negative in all three strata. Two independent implementations agree on the sign and disagree on the magnitude, so we quote only the sign.

Why the decomposition is not identifiable. The within-session execution floor is exactly zero: replicates issued 0–16 seconds apart

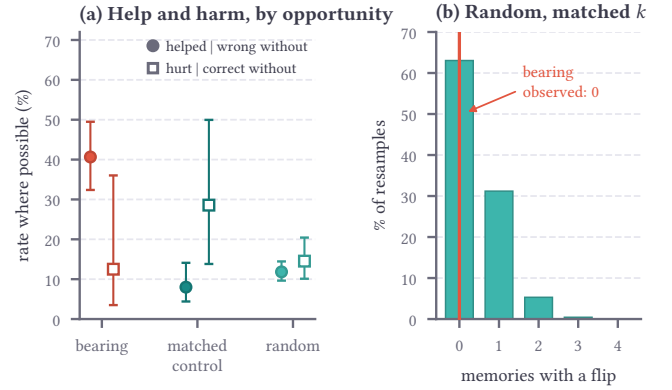


Figure S4: Sign reversal on LongMemEval-V2, screened design (1,157 cells, 60 memories). (a) Help rate among cells the reader got wrong without the memory (filled) and harm rate among cells it got right (hollow), with Wilson 95% intervals; counts are in Table S3. Harm does not differ detectably across strata; help does. (b) Flips among the 35 memories with at least two BEARING cells when each memory’s RANDOM cells are subsampled to its own BEARING-cell count (20,000 resamples); the line marks the observed BEARING count.

agree on 1,157 of 1,157 cells. With no residual scale, the fitted residual collapses and the task and interaction shares are not separately identified, so we report none. Clean per-cell utilities and a variance decomposition are in tension on a reader without execution noise.

S8 Data, code and prompt availability

Public dataset. The Conditional Memory Utility Dataset is available on Hugging Face. The frozen v0.6-length-control snapshot contains 16,816 records: 16,535 from LongMemEval-V2, 154 from SciCode, and 127 from SWE-bench Verified. A row describes a memory–task pair under a representation, reader, condition and replicate, with the outcome under each arm, available output text and cost metadata. Missing or unscorable measurements remain explicitly marked; they must not be converted to zero rewards.

Version and reproducibility. The data file is `data/cells.jsonl`. The repository also provides a manifest with file checksums, schema and validation code, and documentation of the included conditions and upstream sources. Readers can download the data directly or pin the v0.6-length-control revision with the Hugging Face client. The data card contains a runnable download example and explains the validation warnings. This snapshot was built on 2 October 2026 and does not contain every later manuscript analysis; in particular, its condition inventory does not include the fixed-total-context experiment. The public Hub package contains schema and validation code, not the complete experiment, analysis and prompt bundle. The length-control identity arm is an A/A control and must not be pooled with Raw as an independent condition. Per-pair utility labels describe the recorded evaluation conditions, rather than unconditional training targets.

Withheld and missing text. The 154 SciCode records contain Raw memories; their reader outputs were not retained. SciCode Summary and Structured records are absent because their memory text was not persisted. The 127 SWE-bench records provide identifiers and derived measurements, with restricted corpus and output text withheld under the release policy. These are distinct from records missing because of a persistence gap. The data card documents the sentinels and per-source reuse terms; access to the dataset does not grant a new license to withheld upstream content.

S9 Extended limitations

Leakage rates are lower bounds. Our leakage detector reuses the benchmark’s phrase matcher, whose normalizer strips punctuation without inserting a space. A gold phrase next to a quote or an equals sign is glued to its neighbor and the match fails. Such leaks are missed, and punctuation is common in accessibility trees, so every leakage rate we report is a lower bound. A control test records the defect, so an upstream fix would be detected. The 156 LLM-judged questions cannot be audited without an inference call and are recorded as undecidable rather than clean. 69 multiple-choice questions with gold answers of one to five characters cannot be audited structurally and are never folded into a pooled rate.

The judge floor measures decoding. F_{judge} is zero wherever we measured it: over 3,486 explicit re-scorings of 60 units in the earlier official arm, and in every stratum of the screened run. Both estimators carry a positive control (a planted disagreement makes them fire). A zero floor here reflects a deterministic scorer and greedy decoding, not an unambiguous rubric. The SciCode control in Section S6, which disagrees on 5 of 135 cells, is a different scorer on a different benchmark. We have no cross-session floor for any arm and presume it to be non-zero.

Semantic type. We intended semantic memory type as a second axis, but in our representation pilot the two labeling passes agreed on 3 of 8 segments at Cohen’s $\kappa = 0.000$: the structured writer’s own pass called every segment episodic, a degenerate marginal, so we report no semantic-type breakdown. Separately, exactly one deterministically scorable *errors-gotchas* question exists in the benchmark and accounts for 865 of 6,920 web cells, so any per-stratum result there is a single-question result.

Writer failures and faithfulness. The structured writer failed on 3 of 60 segments, all by exhausting the token budget mid-value rather than by malformed output, and on 0 of the 57 writes that finished. An 8-segment pilot had put the failure rate at 0 of 8 and could not have seen this, since it never reached the cap. Our faithfulness measures are a screen, not a guarantee: a low score is strong evidence of a problem, a high score is weak evidence of its absence, and they do not detect recombination, omission, unsupported inference or negation.

Unrecorded quantities. Two quantities were not recorded during the runs and cannot be recovered: how many gold-free picks contain the answer, and how the 12 position-probe cells were chosen, which is why we treat that probe as a pilot.

Stated exclusions. 17 of the 295 deterministically scored questions have no memory unit that carries their gold answer under the

segmentation answerability check, so their cells can only yield $\Delta U \leq 0$. The bearing screen, a looser containment test, fires on no unit for 15 of them (Section S11). 23 of the 1,889 units (1.2%) are their entire trajectory.

S10 Pre-registration record

Each pre-registered analysis was fixed in a pre-commitment document in the code repository before the data it governs existed. The documents are released with the code. To preserve anonymity we omit their commit identifiers here; the release links each one to a commit and its timestamp, so its timing can be checked on de-anonymization. Table S4 lists every pre-registered analysis, its anchor, and every deviation the run ledgers record. Pre-commitments were not edited after the fact; where a result does not match one, the mismatch is reported.

Table S4: Pre-registered analyses, their anchors, and recorded deviations.

analysis	anchor	recorded deviation
screened design and its reduction criterion reversal rate (headline)	criterion fixed in code before the numbers existed pre-registration, section 5	none recorded
variance decomposition	pre-registration, section 6	comparison with an additive null added later, as a diagnostic not identifiable here (zero execution floor); in earlier arms the mixed model did not converge and the registered fallback was used
gold-free screens	gold-free screen pre-commitment	none recorded
length control	frozen at a recorded commit, before execution	3 of 27 descriptive figures do not re-derive: bearing quartile means by Raw length; none appears in the paper
second reader	written before any design-cell call	cross-session estimate superseded by a session-matched run
session-matched reader run	falsifiers committed before data	none recorded
writer $\times$ reader factorial	factorial pre-commitment	one prose sign error ($-\frac{1}{2}$ for $+\frac{1}{2}$ interaction); replication check fails on sign only
SciCode scale-up	scale-up pre-commitment	none recorded

The two deviations that bear on results. The additive-null comparison of reversal rates (main paper, Section 5.6) was *not* pre-registered; the registered headline is the reversal rate itself, and the comparison is a diagnostic added afterwards. In the writer $\times$ reader factorial, the registered criterion asked the re-measured reader effect to reproduce the earlier one in sign and significance. Both estimates are near zero and span zero (Summary $+0.004$ then -0.004 , Structured $+0.011$ then -0.007), so the significance verdict reproduces and the sign does not; the two intervals contain each other’s point estimates. The factorial’s pre-commitment also states that the same-family effect equals $-\frac{1}{2}$ of the interaction; the correct relation, which the analysis code checks numerically, is $+\frac{1}{2}$. Every quantity was computed as registered.

S11 Follow-up runs

Three follow-up runs test the main results under conditions the main design held fixed; all are exploratory and used the same serving endpoint and reader as the main runs. A generator script in the released code computes every number below from the row files.

Second session, randomized position. All 285 bearing and control cells, three replicates each, the memory’s position redrawn per replicate (1,708 real calls, 0 from cache, 2 failed and excluded). Bearing minus control, paired within memory: +0.50 [+0.37, +0.63] over 60 memories; episode-clustered [+0.34, +0.67] (35 episodes); two-way (memory $\times$ task) +0.40 [+0.26, +0.55]. Across the three positions of a cell, the label changes in 28 of 284 cells (10%), and in 28 of the 81 cells that move at least once (35%); position accounts for 11% of the variance of ΔU . Joined one-to-one to the published run, labels agree in 91% of 156 replicate-cells at the same position ($\kappa = 0.78$) and in 87% of 697 at a new one ($\kappa = 0.65$).

Fixed total context. 277 cells over 57 memories, two replicates. For each cell, T is the base filler plus the Raw memory; every other arm, and the no-memory arm, is padded after the base context with bearing-disjoint filler to T (asserted equal across arms), the memory kept at its K2 position. *Disclosed defect:* the no-memory cache key was (trajectory, task, replicate), so two memories from one trajectory paired with one task could share a no-memory response although their padding differs. The headline numbers exclude all 63 such cells ($n = 77$ bearing cells); including them ($n = 135$) changes no conclusion:

Table S5: Bearing-cell ΔU at a fixed total context (task-clustered 95% intervals).

arm	excluding exposed cells	all cells
Raw	+0.35 [+0.23, +0.49]	+0.37 [+0.25, +0.51]
Summary	+0.04 [-0.04, +0.12]	+0.03 [-0.03, +0.09]
Structured	+0.01 [-0.04, +0.07]	-0.01 [-0.07, +0.03]
matched truncation	+0.04 [-0.01, +0.11]	+0.11 [+0.03, +0.20]
Summary – Raw	-0.31 [-0.44, -0.18]	-0.34 [-0.48, -0.21]
Structured – Raw	-0.34 [-0.47, -0.21]	-0.39 [-0.52, -0.26]

The matched truncation is sensitive to the exclusion and supports no claim. The cache key now includes the memory, so a re-run cannot repeat the defect.

Retrieval. The reference labels are the screened design’s (75 tasks, 266 bearing edges, reproduced and asserted). Each method ranks the task’s whole domain pool by the raw question; dense retrieval embeds 512-token chunks (a unit scores its best chunk) or the first 2,000 tokens.

Of the 295 deterministic questions, 15 fire on no unit, 75 on one to eight, and 205 on more than eight. Without the cap, string containment fires for all 1,889 memories, on 18 to 91 tasks each.

Variance decomposition. In the second-session run each cell has three replicates at redrawn positions, which supplies the residual scale the main run lacked. On the 284 cells with all three replicates we fit the additive model $\mu + a_m + b_t$ to cell means by least squares; its residual mean square (105 degrees of freedom) estimates the interaction variance plus the within-cell variance divided by three,

Table S6: Retrieval of bearing units (task-clustered 95% intervals; chance recall@10 is about 0.01). MRR is the mean reciprocal rank of the first bearing unit; AUROC is the area under the ROC curve over bearing and non-bearing units.

method	recall@1	recall@10	MRR	AUROC
BM25	0.03	0.21 [0.14, 0.28]	0.18	0.86
dense, prefix	0.11	0.35 [0.26, 0.44]	0.36	0.82
dense, chunks	0.14	0.40 [0.31, 0.50]	0.41	0.85

and the within-cell variance estimates the residual. The shares are 32% interaction, 55% additive memory and task effects, and 13% position with decoding; the interaction F test gives $F = 8.5$ ($p < 0.001$). This is a method-of-moments estimate on a sparse crossed design, reported as exploratory.

Power for harm. With 16 bearing cells able to show harm, against the random stratum’s rate of 14.5%, the screened design detects (80% power, two-sided $\alpha = 0.05$) only a bearing harm rate of 44% or more; the null on harm is therefore weak evidence.

Containment inside the random-task stratum. Memories in this stratum are real same-domain pages. On the 285 random cells where the containment detector fires (at any breadth) the memory helps in 15.5% of the 220 cells the reader got wrong without it, against 10.1% of 473 where it is silent; harm is 10.8% and 16.7%. Part of the random stratum’s help is content, so its help rate bounds pure perturbation from above.

The gold-free screen over the full universe. The model-based screen was applied to all 277,594 (task, unit) pairs: an exhaustive lexical stage kept 841 candidates, and the judge confirmed 42 over 23 tasks and 28 memories, under the same cap of eight units per task. A gold-free definition does not make bearing denser.

Sparsity across screen caps. Recomputing the screen over the deterministic tasks at four caps on the number of units a task may fire on: at a cap of 4, 47 tasks and 64 bearing memories; at 8 (the paper’s screen), 75 and 165; at 16, 88 and 241; at 32, 101 and 355. At every cap the median bearing memory bears on exactly one task, and 64–72% bear on exactly one. Only removing the cap entirely makes containment fire for every memory.

Which memories were sampled. The 60 memories are every memory that bears on two or more tasks and passes the size cap (35 of the corpus’s 46) plus 25 single-task memories. The headline contrast is +0.36 [+0.12, +0.60] on the single-task memories and +0.32 [+0.19, +0.45] on the multi-task ones; reweighted to the 165 bearing memories of the corpus, of which 72% bear on one task, it is +0.35 [+0.17, +0.53] (memory bootstrap within each group).

S12 Additional figures

Figure S5 collects views referenced from the main paper: bearing breadth, helped cells by stratum, retrieval, the representation contrasts with their token costs, the cross-benchmark contrasts, and the two readers’ agreement. Figure S2 (Section S4.4) details the length-control run.

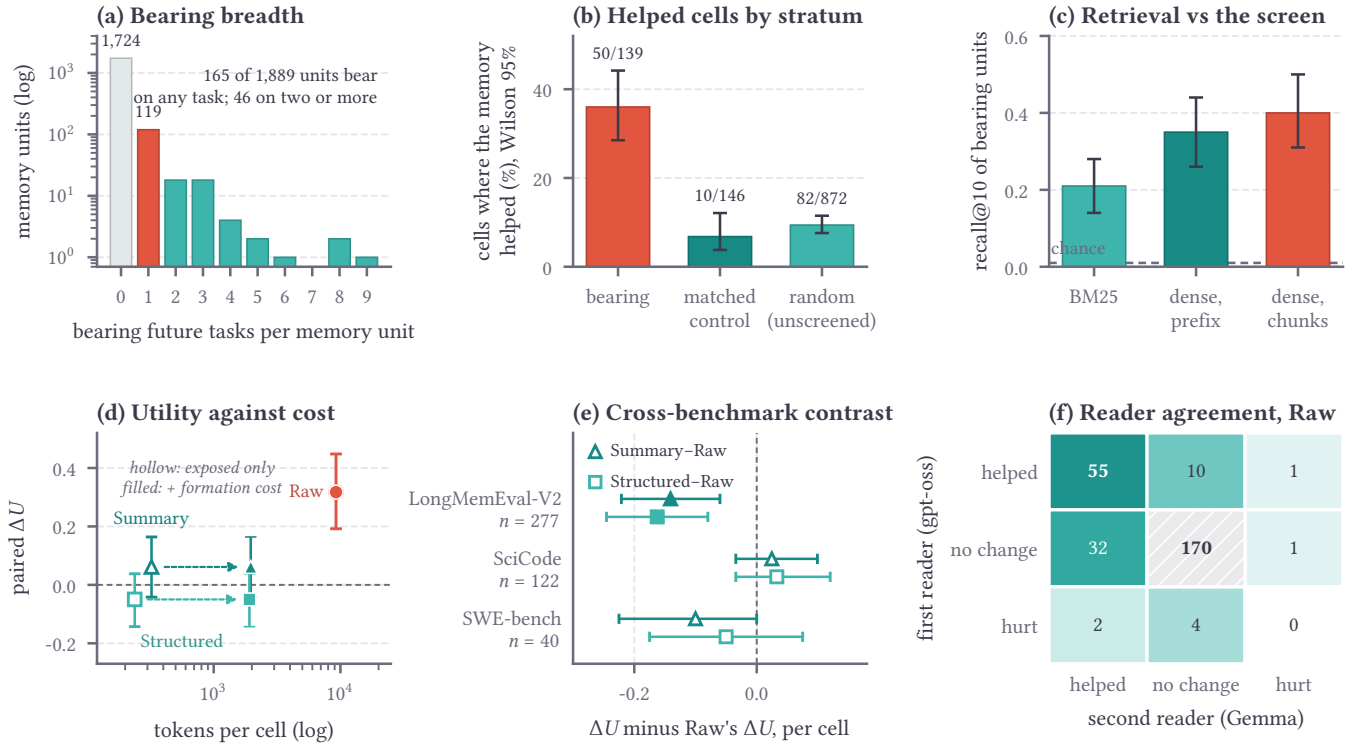


Figure S5: Overview of further results on LongMemEval-V2 (a–d, f) and across benchmarks (e). (a) Future tasks each of the 1,889 memory units bears on, over 75 discriminative tasks (log axis). (b) Share of cells where the memory helped, cell-level Wilson 95% intervals (memory-resampled ones in Section S7), for the same 60 memories in three strata. (c) Share of bearing units a retriever ranks in its top ten of the domain pool (75 tasks, task-clustered 95% intervals); the dashed line is chance. (d) Paired BEARING–control effect per representation (277 cells, 57 memories, intervals over memories); hollow markers count exposed tokens, filled ones add formation cost amortized over 4.86 cells. (e) Per-cell ΔU minus Raw's, task-clustered intervals; filled markers exclude zero. (f) Two readers' labels on 275 Raw cells in one session; the hatched cell (zero under both) cannot disagree.