WHAT AGENT MEMORY IS WORTH: A ZERO-COST RANDOMISED EXPERIMENT INSIDE TEST-TIME SCALING

Anonymous authors
Paper under double-blind review

ABSTRACT

Agent memory frameworks distil items from past trajectories and retrieve them for new tasks, but whether any individual item helps is unmeasured: retrieval is not random, so a memory's presence is confounded with the difficulty of the tasks that retrieved it. The confound can be removed for free. Memory-aware test-time scaling already runs N rollouts per task and hands each of them the same memories; we randomise which subset each rollout gets, turning a budget already spent into an embedded randomised experiment whose blocked difference of means estimates each item's marginal effect. We ran 2,450 rollouts on four WebArena sites under three conditions, 2,235 of which enter paired comparisons. The randomisation executed as designed, with all 2,891 memory–task cells realising the intended two-of-five assignment, but the comparison between conditions failed: conditions ran in sequence against sites that were restarted but not reset, and the carried-over state biased their contrasts in both directions. On the 72 tasks that write state, later conditions found the work already done, and about half of their rollouts succeeded within two steps, against 7% for the baseline. On read-only tasks that query changing state, such as the latest order, later conditions read the altered state correctly and were graded wrong against fixed references, so their success fell from 0.56 to about 0.41. Pooled together, the two biases produced the token saving, the near-chance judge (AUC 0.538) and the sign reversal we reported at registration, and none of the three is a property of the method. Per-memory contrasts are taken within a task and condition and are not confounded by this carry-over, but 76–96% of blocks show no variation in outcome, no item survives multiplicity correction, and resolving a two-point effect would take 250 to 1,700 blocks per memory, against a median of three or four. We document the failure and its evidence in the logs, register with this submission a confirmatory campaign under a protocol that avoids it, and will release the harness and the per-rollout records.

1 INTRODUCTION

Agents that keep a memory bank across tasks are now a standard way to make language-model agents improve with experience. ReasoningBank distils reasoning strategies from trajectories an LLM judge labels as successful or failed, and its memory-aware test-time scaling (MaTTS) runs several rollouts per task to produce richer experience (Ouyang et al., 2026); workflow memories (Wang et al., 2025), trajectory exemplars (Zheng et al., 2024) and extracted insights (Zhao et al., 2024) follow the same pattern of admitting items and retrieving them later. These systems are evaluated at the level of the whole bank: success with memory against success without. Which items carry that difference, and whether any single item helps at all, is not measured.

Nor can it be read off the logs. Retrieval is driven by task similarity, so the tasks on which a memory is present are not a random sample of tasks, and a memory retrieved mainly on easy tasks looks useful whether or not it is. Recent work questions aggregate gains: a token-matched agent without memory matches several memory systems on WebArena (Hajimiri et al., 2026), agents follow retrieved experience even when it is wrong (Xiong et al., 2026), and memory evaluations carry hidden confounds (Wang, 2026).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

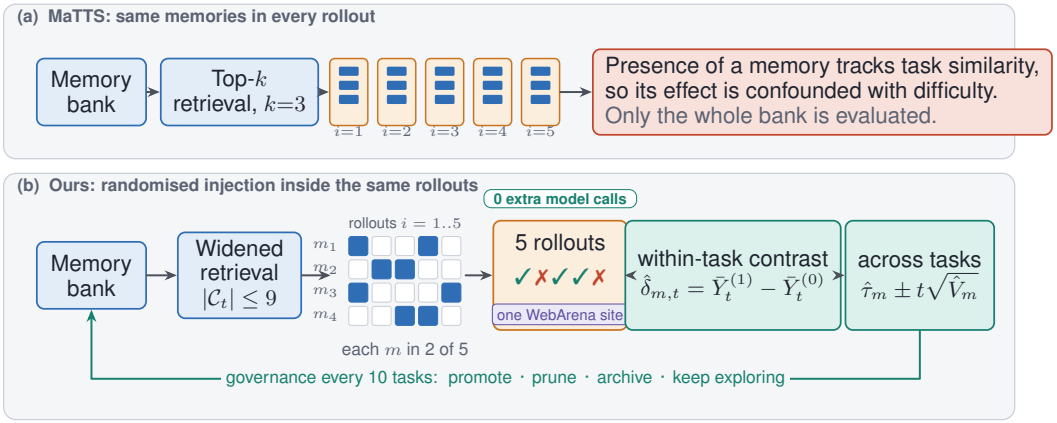

Figure 1: **Overview.** (a) Memory-aware test-time scaling gives the same retrieved memories to all rollouts of a task, so a memory's presence is confounded with the tasks that retrieve it. (b) We inject each retrieved memory into a random two of the five rollouts. The within-task contrast estimates its marginal effect, averaged over tasks and used for governance, without adding any model call.

We observe that test-time scaling already pays for a randomised experiment and then discards it. MaTTS gives the same memories to all N rollouts of a task. If instead each retrieved memory is injected into a random subset of the rollouts, the within-task contrast between rollouts that received it and those that did not is an unbiased estimate of its marginal effect, blocked on the task and free of the retrieval confound, at no additional model calls (Figure 1). Estimating the contribution of context items from random subsets is established (Ilyas et al., 2022; Chang & Jia, 2023; Cohen-Wang et al., 2024). Our variant draws the subsets from rollouts that test-time scaling runs anyway, and its outcome is task success in an interactive environment.

We implemented this design and ran it on four WebArena sites (Zhou et al., 2024) against a reproduction of ReasoningBank with MaTTS. The randomisation executed as specified. The comparison between conditions, which was not randomised, failed because conditions ran one after another against site containers that were restarted but whose databases were not reset. The carried-over state inflated later conditions on tasks that write, and deflated them on read-only tasks whose answer depends on state that other tasks change. A pooled analysis turned these two opposite biases into three plausible headlines, which we had reported when registering this paper, and none of them survives a split by task type. Because every rollout, assignment and answer is logged, the failure can be shown in the records themselves.

Contributions. (1) A design that estimates the marginal effect of individual memory items by randomising injection across the rollouts test-time scaling already runs, with its estimand, variance, power and decision rule stated for the assignment actually used (Section 3), and verified in execution on all 2,891 memory–task cells. (2) A documented case, with log-level evidence, of shared-state carry-over that biases condition comparisons on a stateful agent benchmark in both directions, and of how it manufactured a token saving, a near-chance judge and a sign reversal (Section 5.2). (3) Evidence that the measured discrimination of an LLM judge depends on the validity of the ground truth it is scored against (Section 5.3). (4) A per-memory analysis that leaves every effect unresolved, together with the sample sizes a campaign would need to resolve them (Section 5.4). (5) A confirmatory campaign whose protocol, hypotheses and analysis are fixed with this submission, before any of its data exist (Section 7).

Scope. All claims concern strategy-level memory under parallel test-time scaling, on four single-session WebArena sites, with one backbone. We make no claim to improve on ReasoningBank, and this campaign cannot support any comparison between conditions.

2 Related Work

Agent memory. ReasoningBank admits strategies distilled from both successful and failed trajectories, labelled by an LLM judge, and pairs the bank with MaTTS, which runs k=5 rollouts per task in its main experiments (Ouyang et al., 2026). Agent Workflow Memory induces reusable workflows (Wang et al., 2025), Synapse retrieves whole trajectories as exemplars (Zheng et al., 2024), and ExpeL extracts insights across tasks (Zhao et al., 2024); Reflexion keeps verbal feedback within a task (Shinn et al., 2023). MemGuard persists verifier signals as governance metadata and reports gains of up to 7.9 points over ReasoningBank on WebArena (Wang et al., 2026). All of these evaluate the bank as a whole.

Is memory worth it, and which part? Hajimiri et al. (2026) show that a token-matched vanilla agent matches or surpasses several memory modules on WebArena; Xiong et al. (2026) study how agents follow retrieved experience and how poor admission propagates errors; Wang (2026) argue for controlled baselines against hidden confounds. Srivastava (2026) select memories by causal utility estimated through deterministic interventions, and Cho & Park (2026) separate retrieval of a skill from its actual use.

Attribution by random ablation. Datamodels predict a model's output from which training examples are present in random subsets (Ilyas et al., 2022); Chang & Jia (2023) score in-context examples by accuracy when combined with random others; ContextCite ablates random subsets of context sources and fits a sparse linear surrogate to attribute a generation to them (Cohen-Wang et al., 2024). Our estimator applies the same principle to memory items, with two differences: the random subsets are the rollouts test-time scaling already runs, so no calls are added, and the outcome is task success in a stateful environment, which is where the carry-over we report arises.

Judges, evaluation and design. ReasoningBank reports 72.7% judge accuracy against ground truth on WebArena-Shopping with Gemini-2.5-flash (Ouyang et al., 2026); AgentRewardBench reports precision, recall and F1 for twelve judges on expert-annotated trajectories, with a best overall precision of 69.8% (70.2% on WebArena) (Lù et al., 2025). Both are threshold-dependent measures on other populations; we report AUC (Hanley & McNeil, 1982), which is not directly comparable to them. WebArena Verified removes LLM judges from the benchmark's own evaluation (El hattami et al., 2025). CATTS allocates test-time compute per step for web agents without memory (Lee et al., 2026). Our estimator is Neyman's difference of means within blocks (Splawa-Neyman, 1990; Imbens & Rubin, 2015); with several memories randomised at once, the estimand is an average marginal component effect (Hainmueller et al., 2014); within-task interference follows Hudgens & Halloran (2008). Multi-session benchmarks (He et al., 2026) and repository-level tasks (Jimenez et al., 2024) are outside our scope.

3 Method: randomised injection inside test-time scaling

Setting. Tasks $t = 1, \ldots, T$ arrive in sequence. For each task the agent retrieves a candidate set $\mathcal{C}_t$ from the bank and runs $N = 5$ rollouts. A ReasoningBank-style baseline injects the same top-k memories ($k = 3$) into all rollouts. Our conditions widen retrieval to $|\mathcal{C}_t| \leq 3k = 9$ candidates and randomise which of them each rollout receives.

Assignment. Let $Z_{tim} \in \{0, 1\}$ indicate that memory $m \in \mathcal{C}_t$ is injected into rollout i. For each memory independently, the treated rollouts are a uniform draw of two of the five:

$$\Pr(\mathbf{Z}_t = \mathbf{z}) = \prod_{m \in \mathcal{C}_t} \binom{5}{2}^{-1} \mathbf{1}\Big[\textstyle\sum_i z_{im} = 2\Big]. \tag{1}$$

The draws come from a stream seeded by a hash of the global seed and the task identifier, so every assignment is reproducible. (The design is configured with $\pi = 0.5$ and rounds $N\pi = 2.5$ to two treated rollouts.) Within a memory, $\mathrm{corr}(Z_{im}, Z_{i'm}) = -1/4$; across memories the columns are independent, so a rollout receives $\mathrm{Bin}(|\mathcal{C}_t|, 0.4)$ memories. This is neither an independent Bernoulli design nor a fixed number of memories per rollout, and the variance below is derived for it. Figure 2 shows one block.

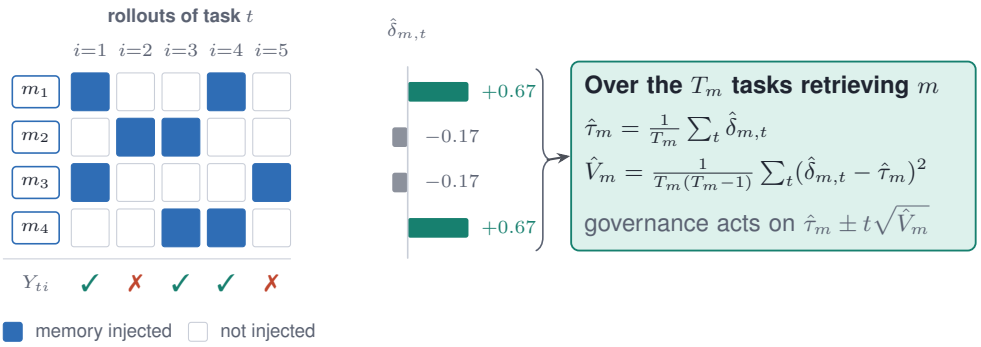

Figure 2: One task block. Each retrieved memory (rows) is injected into two of the five rollouts (columns), drawn independently per memory. With the illustrative outcomes Y_{ti} shown, each row gives the contrast $\hat{\delta}_{m,t}$ between treated and untreated rollouts; contrasts are then averaged over the tasks on which the memory was retrieved.

Estimand. For outcome Y, the per-task contrast $\hat{\delta}_{m,t} = \bar{Y}_t(Z_{\cdot m}{=}1) - \bar{Y}_t(Z_{\cdot m}{=}0)$ has expectation over (1) $\tau_{m,t} = \frac{1}{5}\sum_i \big(\mathbb{E}[Y_{ti} \mid Z_{tim}{=}1] - \mathbb{E}[Y_{ti} \mid Z_{tim}{=}0]\big)$, and the target is $\tau_m = T_m^{-1}\sum_t \tau_{m,t}$ over the T_m tasks that retrieved m. With no interference between rollouts, τ_m is the average marginal component effect of m with every other candidate included with probability $2/5$ (Hainmueller et al., 2014); it depends on the co-assignment law and on the bank, and does not transfer to a different retrieval policy. With interference between rollouts of the same task, $\hat{\delta}_{m,t}$ is still unbiased for $\tau_{m,t}$, but $\tau_{m,t}$ then mixes the direct effect of m with spillover from siblings (Hudgens & Halloran, 2008): a control rollout has two treated siblings, a treated one has one. We assume no additivity across memories; the estimand is marginal. The contrast is also taken in whatever state the environment is in when the block runs; if that state makes the task impossible or trivially satisfied, every rollout in the block has the same outcome and the block contributes nothing.

Estimator and variance. Blocks are weighted equally: $\hat{\tau}_m = T_m^{-1}\sum_t \hat{\delta}_{m,t}$ with $\hat{V}_m = \sum_t(\hat{\delta}_{m,t} - \hat{\tau}_m)^2/\{T_m(T_m-1)\}$ and a t interval on $T_m - 1$ degrees of freedom. $\hat{V}_m$ is unbiased for the variance over a superpopulation of tasks and conservative for the realised tasks (Imbens & Rubin, 2015). No within-block variance is estimated: one two-versus-three draw per block does not identify it. The effective sample size is the number of tasks, not rollouts. When the same memory appears in two conditions on the same task, the ten rollouts form one block with four treated; condition is orthogonal to assignment within such blocks.

Reading magnitudes. For $\hat{\tau} \sim \mathcal{N}(\mu, \sigma^2)$, $|\mu| \le \mathbb{E}|\hat{\tau}| \le |\mu| + \sigma\sqrt{2/\pi}$, so the mean of $|\hat{\tau}_m|$ over memories is positive even when every μ is zero. We compare it with its distribution under the sharp null, obtained by re-drawing the two-of-five assignment within each task and condition.

Power. To detect an effect δ at level α with Bonferroni over the K' memories that have at least two blocks, and power 0.8, a memory needs

$$T_m \;\ge\; \big(z_{1-\alpha/(2K')} + z_{0.8}\big)^2\,\hat{\sigma}^2/\delta^2 \tag{2}$$

blocks, where $\hat{\sigma}^2$ is the pooled between-block variance of $\hat{\delta}_{m,t}$. There is no factor of two: the unit is already a difference.

Governance. The bank is governed every ten tasks by five rules on each memory's bounds at family-wise level 0.10 (Bonferroni over the memories in the bank): promote if the lower bound exceeds $\tau^+ = 0.02$; prune if the upper bound is below zero; archive an established null when $T_m \ge 15$ and the interval, of width at most 0.06, contains zero; downgrade on a declining history; otherwise keep exploring. Repeated looks are not corrected for.

The outcome used inside the loop. Governance and the choice of the rollout to learn from use a composite $Y = 0.25V + 0.35A + 0.40J$ of execution validity V (termination, invalid actions,

Table 1: The three conditions. All use the same backbone, step cap and task order; they were run one after another, not interleaved.

Condition	Injection	Bank	Role
B3	same top-3 memories in all 5 rollouts	built online (ReasoningBank-style induction)	reproduction of ReasoningBank + MaTTS
M-online	up to 9 candidates, each in 2 of 5 rollouts	built online; governance every 10 tasks	full loop
M-frozen	up to 9 of 12 fixed memories, each in 2 of 5	12 memories selected from M-online's bank on exposure; no induction	more blocks per memory

and with weight 0.5 an efficiency share for finishing in few steps), agreement A of the rollout's answer with its siblings' answers, and an LLM judge score J. Dropping J renormalises the others to $0.417V + 0.583A$. Because A compares a rollout with its siblings, Y contains a within-task interference channel by construction. For this reason Section 5 reports every result on the ground-truth task success returned by the WebArena evaluator.

4 Experimental setup

Environment and tasks. We use four WebArena sites (Zhou et al., 2024) (reddit, gitlab, shopping, shopping_admin) served from WebArena Verified images (El hattami et al., 2025), following ReasoningBank's four-site protocol. A frozen subset of 200 tasks is drawn with a fixed seed and a checksum; its 172 non-map tasks were run. After attrition, 149 tasks are complete in all three conditions (22/37/49/41 by site), giving $149 \times 5 \times 3 = 2{,}235$ paired rollouts out of 2,450 scored and 2,580 launched. Losses differ by condition (B3 8.1%, M-online 4.1%, M-frozen 2.9%); a crash removes the whole task from the paired set. The agent, inducer and judge are `deepseek-v4-flash`; episodes are capped at 15 steps.

Conditions. Table 1. M-frozen necessarily runs after M-online, whose bank it samples. Final bank sizes are 50–93 memories per site for B3 and 51–92 for M-online; near-duplicate merging never fired.

Judge. The judge is `deepseek-v4-flash` at temperature 0, shown the last 24k characters of the rollout log (Appendix C). Two defects must be stated. First, in the offline rescoring that produced our final scores, the judge received the task identifier instead of the task statement. On the 1,634 randomised-condition rollouts that were also scored in the loop with the statement, the two scores are identical in 84.5% of cases ($r = 0.92$), and their AUCs are 0.472 and 0.473; the defect is listed among the threats in Section 6. Second, the step meant to strip injected memories before judging had nothing to remove: none of the 2,580 logs contains the injection markers, because the injected block is not written to the log, while the agent's own restatements of memories remain. The judge decided admission in no condition: B3 ran without it, M-frozen induces nothing, and in M-online it entered only through Y when choosing which rollout to learn from.

Ground truth. Success is the WebArena evaluator's verdict. On 16 of the 149 tasks this verdict comes from a fuzzy-match step that calls an LLM; since the original model was not reachable, it was routed to the same DeepSeek model, identically for all conditions. We report these 16 tasks separately where it matters.

Execution protocol. The five rollouts of a task ran concurrently against one instance of the site; tasks ran in sequence; conditions ran in the fixed order B3, M-online, M-frozen, each over the whole task list before the next began. Before each condition the site container was restarted, which does not reset its database. Condition was not randomised, so every contrast between conditions compares runs made at different times on an environment that earlier conditions had modified.

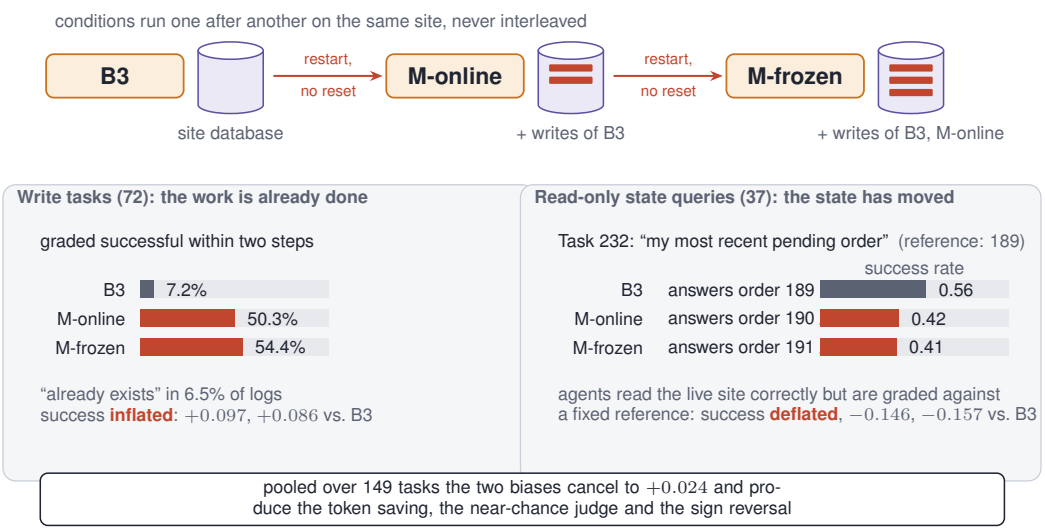

Figure 3: **Carry-over between conditions.** Conditions ran in sequence on sites that were restarted but not reset. Later conditions find write tasks already done (left) and read state that earlier conditions changed (right, task 232). The two biases have opposite signs and cancel in the pooled contrast.

Task classes. WebArena grades 72 of our tasks by inspecting site state (`program_html`; 19/21/14/18 by site): we call them *write* tasks. The other 77 are graded on the answer or the final URL. Following an adversarial review of an earlier draft, we further split these 77 by a keyword rule on the task statement that flags queries about changing state ("latest", "most recent", "count", "status", "top", and similar; the rule and the 37 tasks it selects are listed in Appendix B). This split was defined after we had seen per-task results; it is exploratory, and we use it only to show the direction of the bias.

Intervals. Contrasts between conditions are paired by task, with t intervals that are 95% simultaneous over the two comparisons with B3 (Bonferroni; each 97.5%), per outcome and subset, without adjustment across outcomes or subsets. AUC intervals are from a task-clustered bootstrap (2,000 resamples). Randomisation p-values use 4,000 re-draws with a fixed seed (Monte Carlo standard error below 0.008). Token counts are agent input tokens per rollout, averaged within task; judge and inducer calls are excluded.

5 Results

5.1 The design executed as specified

All 2,891 memory–task cells in the randomised conditions have exactly two treated rollouts out of five, and all 329 task blocks replay from their seeds. The frequency with which two given memories are co-injected in a rollout is 0.1598, against $0.4^2 = 0.16$ under (1). Rollouts received 3.37 (M-online) and 3.60 (M-frozen) memories on average, against 3 in B3. The 2,033 contrast blocks include 558 pooled ten-rollout blocks for the 12 frozen-bank memories that also appear in M-online.

5.2 Carry-over biased condition contrasts in both directions

Write tasks: later conditions find the work done. The share of rollouts graded successful within two steps is 26 of 360 for B3 (7.2%), 181 of 360 for M-online (50.3%) and 196 of 360 for M-frozen (54.4%) on write tasks, against 56, 48 and 47 of 385 on read-only tasks. The phrase "already exists" appears in 6.5% of write-task rollout logs and in none of the read-only ones. On shopping task 469, all ten randomised-condition rollouts were graded successful after a single step; B3's took two or three.

Table 2: Contrasts against B3, paired by task, 95% simultaneous over the two conditions. Success is ground truth; tokens are agent input tokens per rollout. The last two columns split the 77 read-only tasks by the exploratory rule of Appendix B. Bottom rows: success rate by condition.

Outcome	Condition	Pooled (149)	Write (72)	Read, state query (37)	Read, other (40)
Success	M-online	$+0.024\,[-0.033, +0.082]$	$+0.097\,[+0.017, +0.177]$	$-0.146\,[-0.289, -0.002]$	$+0.050\,[-0.022, +0.122]$
	M-frozen	$+0.017\,[-0.038, +0.073]$	$+0.086\,[+0.010, +0.162]$	$-0.157\,[-0.294, -0.020]$	$+0.055\,[-0.016, +0.126]$
Tokens	M-online	$-7{,}875\,[-15{,}395, -355]$	$-19{,}220\,[-30{,}611, -7{,}829]$	$+4{,}741\,[-13{,}935, +23{,}417]$	$+877\,[-6{,}001, +7{,}754]$
	M-frozen	$-1{,}698\,[-10{,}267, +6{,}870]$	$-16{,}467\,[-28{,}297, -4{,}636]$	$+22{,}733\,[+128, +45{,}337]$	$+2{,}287\,[-5{,}289, +9{,}862]$
Success rate	B3 / M-online / M-frozen	0.581 / 0.605 / 0.599	0.608 / 0.706 / 0.694	0.562 / 0.416 / 0.405	0.550 / 0.600 / 0.605

Table 3: Judge AUC against ground-truth success, with task-clustered 95% bootstrap intervals. Base rate is the share of successful rollouts. Rows marked † use the exploratory split.

Tasks	Tasks	Rollouts	Base rate	AUC	B3 / M-online / M-frozen
All	149	2,207	0.603	0.538 [0.475, 0.601]	0.637 / 0.476 / 0.504
Read-only, deterministic grading	61	906	0.532	0.689 [0.613, 0.766]	0.669 / 0.681 / 0.715
† state queries	30	445	–	0.746 [0.647, 0.839]	0.774 / 0.737 / 0.738
† other	31	461	–	0.627 [0.509, 0.742]	0.564 / 0.621 / 0.695
Read-only, LLM-graded	16	238	0.525	0.817 [0.711, 0.911]	0.891 / 0.724 / 0.832
Write	72	1,063	0.680	0.359 [0.273, 0.457]	0.544 / 0.268 / 0.279

Read-only tasks: later conditions read changed state. Task 232 asks for "the order number of my most recent pending order"; the reference answer is 189. The agent's answers, from the logs, were order 189 (placed 5/2/23) in B3, order 190 in M-online and order 191 in M-frozen, both placed during our campaign by write tasks. The later agents read the site correctly and were graded wrong. Tasks 46 ("most recent open issues"), 78 ("count of approved reviews"), 96 ("my latest order") and 232 go from five successes of five in B3 to none of ten in the later conditions. On the 37 read-only tasks the exploratory rule flags as queries about changing state, success falls from 0.562 to 0.416 and 0.405 (Table 2).

Consequences. The pooled success contrasts (Figure 3) ($+0.024$, $+0.017$) are the sum of a positive bias on write tasks and a negative one on state queries. M-online's pooled saving of 7,875 input tokens per rollout comes entirely from write tasks, where it tracks a drop of three steps per rollout ($-3.03\ [-4.14, -1.91]$), against $-0.01\ [-0.69, +0.67]$ steps on read-only tasks. The sign reversal of Y when the judge is dropped ($-0.040\ [-0.064, -0.016]$ against $+0.040\ [+0.010, +0.070]$ for M-frozen) is -0.073 against $+0.079$ on write tasks and -0.009 against $+0.003$ on read-only tasks. On the 40 remaining read-only tasks, neither randomised condition differs from B3 in success or tokens. This subset supports no conclusion about the method, and neither does any other contrast between conditions in this campaign: the subset was chosen after the fact and may still contain state-dependent tasks, and conditions were never randomised.

5.3 Judge discrimination depends on the ground truth

Pooled over all tasks, the judge separates successful from failed rollouts at AUC 0.538, and its task-clustered interval [0.475, 0.601] includes 0.5 (Table 3). This figure averages very different regimes. On read-only tasks with deterministic grading, AUC is 0.689 [0.613, 0.766]; the two exploratory subsets differ (0.746 and 0.627) but their intervals overlap widely, and we draw no conclusion from the difference. On the 16 tasks whose ground truth itself comes from a DeepSeek fuzzy match, AUC is 0.817, which measures agreement between two calls to the same model family more than it measures validity. On write tasks it falls below chance in the later conditions (0.268, 0.279): there the ground truth credits rollouts that found the work already done, and the judge does not; on task 469 it scored each of the ten one-step "successes" 0. The judge's apparent preference for B3 follows the same pattern: the paired M-frozen minus B3 difference in mean judge score is -0.155 $[-0.214, -0.095]$ pooled, -0.289 on write tasks and $-0.029\ [-0.079, +0.020]$ on read-only ones. The near-chance pooled AUC thus reflects a ground truth that was corrupted on part of the tasks,

Table 4: Per-memory effects on ground-truth success, all 149 tasks, randomised conditions. K: memories ever retrieved; blocks: memories with at least one (two) blocks; degenerate: share of task–condition blocks in which all rollouts have the same outcome; null: mean $|\hat{\tau}_m|$ under 4,000 re-randomisations; p: one-sided; BH: discoveries at FDR 0.10; need: blocks per memory for 80% power at $\delta = 0.02$ and 0.05 (Eq. 2).

Site	K	Blocks $\geq$1 ($\geq$2)	Median	Degenerate	Mean $\lvert\hat{\tau}\rvert$	Null	p	BH	Need .02	Need .05
reddit	67	34 (27)	3.5	0.955	0.013	0.012	0.39	0	252	41
gitlab	120	58 (49)	4	0.865	0.063	0.062	0.42	0	1,001	161
shopping	149	75 (57)	3	0.765	0.098	0.078	0.010	0	1,656	265
shopping_admin	141	68 (53)	4	0.756	0.042	0.040	0.38	0	1,554	249

and says little about the judge itself. All of these figures come from a judge that never saw the task statement.

5.4 Per-memory effects

Per-memory contrasts are taken within a task and a condition, so the state a block inherits is common to its five rollouts and does not confound the contrast, although it limits what the contrast can show. Between 76% and 96% of task–condition blocks have the same outcome in every rollout (Table 4), and such a block contributes a contrast of zero for every memory in it; a task made impossible or trivial by inherited state is one source. The resulting estimates are attenuated towards zero and should not be read as small effects.

Few memories are estimable: of 67–149 memories retrieved per site, 34–75 have a block, and the median has three or four. On reddit, gitlab and shopping_admin, the mean absolute effect is what re-randomisation produces under the sharp null (ratio 1.02–1.06). Shopping shows more dispersion than the null (ratio 1.25, $p = 0.010$), which is compatible with some items mattering there; no single memory survives a Benjamini–Hochberg correction at 0.10 (Benjamini & Hochberg, 1995) on any site. Resolving a two-point effect at 80% power would take 250 to 1,700 blocks per memory, and a five-point effect 40 to 270. Whether any memory helps is thus open at this scale, and Eq. (2) gives the campaign size needed to settle it.

Governance was equally inert. In M-online it recorded 244 decisions, all moving a candidate to active; no memory was promoted, pruned, archived or downgraded. With intervals this wide, only the rule that keeps exploring could fire.

6 Threats to validity

(1) Unrandomised, sequential conditions on a stateful environment make every condition contrast uninterpretable (Section 5.2). *(2) Within-task concurrency*: the five rollouts of a task share one site, so the per-memory estimand mixes direct and spillover effects, in a direction we cannot identify. *(3) Attenuation* of per-memory contrasts by degenerate blocks. *(4) Agreement inside Y* couples siblings; reported results use ground truth. *(5) Judge input*: the rescoring judge saw the task identifier, not the statement, and memory stripping removed nothing. *(6) Evaluator substitution* on 16 tasks, reported separately. *(7) Attrition* is unequal across conditions (2.9–8.1%) and removes whole tasks. *(8) Baselines*: one backbone, a 15-step cap and no memory-free condition; B3's absolute success (0.581 on the paired tasks) comes from a different backbone, step cap and task subset than ReasoningBank's published results and is not validated against them. *(9) Bank dilution*: merging never fired, banks grew to 50–93 memories, and observations were spread over up to 149 simultaneous tests. *(10) Upper bounds not run*: conditions that put the ground truth in the loop in place of the judge or of retrieval are specified in our harness but not implemented. *(11) Post-hoc subsets*: the split of read-only tasks was defined after seeing results.

7 A PRE-REGISTERED CONFIRMATORY CAMPAIGN

Because the campaign above cannot answer its own questions, we fix, with this submission and before collecting any data, the protocol, hypotheses and analysis of a second campaign. We will report its outcome whichever way it falls, as a revision if it completes in time and otherwise with the release.

Protocol. The same 172 non-map tasks of the frozen subset, the same backbone and step cap. Four conditions: B1 (no memory, $N = 5$), B3, M-online and M-frozen (a fixed bank of $K = 12$ memories). For each task the four conditions run in a random order drawn from a seeded stream, and every rollout runs on a site instance created from the image immediately before it, so no rollout can see another's writes. Tasks 232 and 78 are run on a fresh instance every 20 tasks as sentinels; if either fails to return its reference answer, the campaign stops. The injected block is written to the log between markers, and the judge receives the task statement. The primary outcome is ground-truth success; tasks lost to a crash are dropped from every condition and reported.

Hypotheses and analysis. Three condition-level hypotheses, paired by task, with Bonferroni over the three: *H1*, M-online is not worse than B3 in success by more than 0.07 (one-sided, non-inferiority); *H2*, M-online and B3 differ in agent input tokens per rollout (two-sided); *H3*, B3 and B1 differ in success (two-sided), that is, whether the memory bank helps at all. One item-level hypothesis, *H4*: in M-frozen, the mean $|\hat{\tau}_m|$ over the 12 memories exceeds its re-randomisation null (4,000 re-draws), with individual memories tested by Benjamini–Hochberg at 0.10.

Power, stated in advance. Using the between-task standard deviations of this campaign as planning values (0.30 for the success difference, 40,500 tokens for the token difference), 172 tasks give power 0.75 for H1 when the true difference is zero, and a minimum detectable token difference of about 10,000 at 80% power for H2; the 7,875-token difference of our pooled analysis would not be reliably detected. For H4, if each task retrieves 9 of the 12 memories, each memory gets about 129 blocks, and Eq. (2) with $K' = 12$ gives a minimum detectable effect of 0.05 to 0.06 for between-block standard deviations of 0.15 to 0.19. The campaign requires $4 \times 172 \times 5 = 3{,}440$ rollouts.

8 CONCLUSION

Randomising which memories each test-time-scaling rollout receives costs no extra calls and executes as designed, and it turns the question "which memories help" into one with intervals. In this campaign the question stays open: individual effects are unresolved at the scale of a full campaign, and every difference between conditions was produced by state carried over between them, in both directions. The protocol changes follow from this: a fresh site instance for every rollout, conditions interleaved in random order, sentinel tasks whose answers depend on mutable state checked at the start of each condition, the injected block written to the log, the judge given the task statement, a memory-free condition, and a campaign sized by Eq. (2) around a few memories rather than a hundred.

AI USE STATEMENT

Generative AI tools were used in four ways in this work: to aid and polish the writing; for retrieval and discovery of related work; for research ideation and execution, including writing and debugging parts of the experimental harness and the analysis scripts and designing the analyses; and improve the writing of certain parts of the paper. Every reference was checked against its published record, and every number in the paper was recomputed from the logged per-rollout records with deterministic scripts that will be released. AI assistance also introduced errors, including the judge-input defect of Section 4; these were found by re-running analyses from the raw records and by an adversarial review of an earlier draft. The authors take responsibility for the full content of this work.

Ethics statement

All agent actions were executed against self-hosted WebArena containers, with no real users or external services. The carry-over we report is a reproducibility hazard for any benchmark that shares state across conditions, and a judge whose errors correlate with a condition can bias what a self-improving memory admits.

Reproducibility statement

Section 3 and Appendix A give the design, estimator, variance and power formulas; Section 4 gives the protocol, including its defects; Appendix B gives the exploratory task rule. The frozen task subset (200 tasks, fixed seed, checksum), the harness with its test suite, the per-rollout records and logs of all 2,450 scored rollouts, and the scripts that regenerate every table will be released after the review period; an anonymised copy can be made available to reviewers through OpenReview.

References

Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B*, 57(1):289–300, 1995.

Ting-Yun Chang and Robin Jia. Data curation alone can stabilize in-context learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8123–8144, 2023.

Seonghyeon Cho and Chanjun Park. Skill following: Evaluating actual skill use in retrieval-enabled LLM agents, 2026. arXiv:2609.00549.

Benjamin Cohen-Wang, Harshay Shah, Kristian Georgiev, and Aleksander Madry. ContextCite: Attributing model generation to context. In *Advances in Neural Information Processing Systems*, 2024. arXiv:2409.00729.

Amine El hattami, Megh Thakkar, Nicolas Chapados, and Christopher Pal. WebArena verified: Reliable evaluation for web agents. In *Workshop on Scaling Environments for Agents, NeurIPS*, 2025.

Jens Hainmueller, Daniel J. Hopkins, and Teppei Yamamoto. Causal inference in conjoint analysis: Understanding multidimensional choices via stated preference experiments. *Political Analysis*, 22(1):1–30, 2014.

Sina Hajimiri, Masih Aminbeidokhti, Jose Dolz, Ismail Ben Ayed, Issam H. Laradji, Spandana Gella, and Nicolas Gontier. Are online skill and memory modules always worth their tokens? A budget-constrained study of web agents, 2026. arXiv:2606.15017.

J. A. Hanley and B. J. McNeil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143(1):29–36, 1982.

Zexue He, Yu Wang, Churan Zhi, Yuanzhe Hu, Tzu-Ping Chen, Lang Yin, Ze Chen, Tong Arthur Wu, Siru Ouyang, Zihan Wang, Jiaxin Pei, Julian McAuley, Yejin Choi, and Alex Pentland. MemoryArena: Benchmarking agent memory in interdependent multi-session agentic tasks, 2026. arXiv:2602.16313.

Michael G. Hudgens and M. Elizabeth Halloran. Toward causal inference with interference. *Journal of the American Statistical Association*, 103(482):832–842, 2008.

Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry. Datamodels: Predicting predictions from training data, 2022. arXiv:2202.00622.

Guido W. Imbens and Donald B. Rubin. *Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction*. Cambridge University Press, 2015.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world Github issues? In *International Conference on Learning Representations (ICLR)*, 2024.

Nicholas Lee, Lutfi Eren Erdogan, Chris Joseph John, Surya Krishnapillai, Michael W. Mahoney, Kurt Keutzer, and Amir Gholami. Agentic test-time scaling for WebAgents, 2026. arXiv:2602.12276.

Xing Han Lù, Amirhossein Kazemnejad, Nicholas Meade, Arkil Patel, Dongchan Shin, Alejandra Zambrano, Karolina Stańczak, Peter Shaw, Christopher J. Pal, and Siva Reddy. AgentRewardBench: Evaluating automatic evaluations of web agent trajectories. In *Conference on Language Modeling (COLM)*, 2025. arXiv:2504.08942.

Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long T. Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. ReasoningBank: Scaling agent self-evolving with reasoning memory. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2509.25140.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652, 2023.

Jerzy Splawa-Neyman. On the application of probability theory to agricultural experiments. Essay on principles. Section 9. *Statistical Science*, 5(4):465–472, 1990. Translated and edited by D. M. Dabrowska and T. P. Speed.

Saksham Sahai Srivastava. Causal intervention-based memory selection for long-horizon LLM agents, 2026. arXiv:2605.17641.

Haoyu Wang, Guangyuan Dong, He Liang, Zijing Zhang, Jiachen Luo, Chuang Liu, Chao Xue, and Hao Tang. MemGuard: Persisting verifier signals for LLM-agent memory governance, 2026. arXiv:2608.21867.

Kuan Wang. MemDelta: Controlled baselines and hidden confounds in agent memory evaluation, 2026. arXiv:2606.29914.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *PMLR*, pp. 63897–63911, 2025.

Zidi Xiong, Yuping Lin, Wenya Xie, Pengfei He, Zirui Liu, Jiliang Tang, Himabindu Lakkaraju, and Zhen Xiang. How memory management impacts LLM agents: An empirical study of experience-following behavior. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 623–645, 2026.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024.

Longtao Zheng, Rundong Wang, Xinrun Wang, and Bo An. Synapse: Trajectory-as-exemplar prompting with memory for computer control. In *International Conference on Learning Representations (ICLR)*, 2024.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations (ICLR)*, 2024.

A Derivations

Unbiasedness under interference. Under (1) each rollout is treated with probability $2/5$ and $\hat{\delta}_{m,t} = \frac{1}{2}\sum_i Z_{tim}Y_{ti} - \frac{1}{3}\sum_i(1 - Z_{tim})Y_{ti}$. Taking expectations rollout by rollout, $\mathbb{E}\,\hat{\delta}_{m,t} = \frac{1}{5}\sum_i \big(\mathbb{E}[Y_{ti} \mid Z_{tim}{=}1] - \mathbb{E}[Y_{ti} \mid Z_{tim}{=}0]\big) = \tau_{m,t}$, whatever Y_{ti} depends on; interference changes the meaning of $\tau_{m,t}$, not the unbiasedness.

Variance. With $\hat{\delta}_{m,t} = \tau_{m,t} + \varepsilon_{m,t}$ and independent draws across tasks, $\mathbb{E}\hat{V}_m = T_m^{-2}\sum_t \mathrm{Var}(\varepsilon_{m,t}) + \{T_m(T_m - 1)\}^{-1}\sum_t(\tau_{m,t} - \tau_m)^2$, the second term being the usual conservative heterogeneity term (Imbens & Rubin, 2015).

Folded normal. For $X \sim \mathcal{N}(\mu, \sigma^2)$, $\mathbb{E}|X| = \sigma\sqrt{2/\pi}\, e^{-\mu^2/2\sigma^2} + \mu\{1 - 2\Phi(-\mu/\sigma)\}$, which lies between $|\mu|$ and $|\mu| + \sigma\sqrt{2/\pi}$.

Power. Eq. (2) with $\alpha = 0.05$, K' the memories with at least two blocks, and $\hat{\sigma}^2 = \sum_m (T_m - 1)s_m^2 / \sum_m (T_m - 1)$ pooled over those memories. The between-block standard deviations are 0.08, 0.15, 0.20 and 0.19 by site (reddit, gitlab, shopping, shopping_admin).

B Exploratory rule for state queries

A read-only task is flagged as a query about changing state if its statement matches, case-insensitively and as whole words, one of: *most recent*, *latest*, *last*, *recent*, *newest*, *current*, *status*, *count*, *how many*, *number of*, *total*, *top*, *most*, *least*, *best-selling* (or *best selling*), *pending*, *approved*, *search terms*. It flags 37 of the 77 read-only tasks: 6, 13, 14, 27, 43, 46, 48, 66, 69, 78, 79, 94, 96, 109, 127, 128, 136, 170, 173, 181, 188, 195, 196, 205, 232, 239, 245, 246, 283, 286, 292, 298, 302, 309, 337, 348, 796. The rule was written after an adversarial review had pointed to tasks 232, 78, 46, 96 and 43; it is coarse, and some flagged tasks may not depend on state that our write tasks change, and conversely.

C Judge prompt

```
You are grading whether a web agent accomplished a task. You see only the
task statement and the agent's trajectory. You do not see the correct
answer, and you must not invent one.

TASK
{task}

TRAJECTORY (last part)
{trajectory}

Rate how likely it is that the task was accomplished, on a scale
from 0 to 1:
- 0.0  clearly failed, or never reached anything resembling the goal
- 0.5  genuinely uncertain from the trajectory alone
- 1.0  clearly accomplished

Answer with the number alone. No explanation, no units, no other text.
```

In the offline rescoring, `{task}` was filled with the task identifier rather than the task statement (Section 4). Unparsable replies score 0.5.