Replaying a production memory distiller: a fidelity control that failed, and what it found

Luiz Antonio Busnello

Version 1.0 · doi:10.5281/zenodo.24017487

Version 1.0 (2026-11-06), doi:10.5281/zenodo.24017487. Not peer reviewed. A data note on one deployment, with measurements taken on 2026-11-06 over sessions whose last event falls between 2026-08-15 and 2026-10-30. Every result with a decimal, a percentage or a thousands separator has an entry in `PROVENANCE.json`, in the code-and-data archive of this record.

Abstract

We pre-registered an offline comparison of two extraction criteria for an agent memory system on its own production sessions: the deployed criterion C0 against a revised criterion C2, which had won on a public benchmark. The registered order ran in full: pre-registration tag, frozen sample, question set, model runs, a blind human sample and the reading, each step committed publicly before the next. The reading stopped at its controls. Three of the four passed. The fourth, a fidelity control that asks whether our replay of C0 reproduces what production stored, failed: on the 30 sessions production sent to its model, answers from production's stored memory were correct 0.248 of the time and answers from our replay 0.438, against a tolerance of 0.05. By the registered rule, the two hypotheses are reported as not interpreted. A diagnosis written after the result found a defect in the replay that plausibly explains the failure, though it was not tested by rerunning the replay. Production's distiller drops the message that opens a scheduled job (the job's prompt) before it distils; our replay did not. That prompt was 82% of the text our replay sent to the model, and 138 of the 255 primary questions cite only text inside it. A follow-up census under production's input rule found that the distiller sent few eligible sessions to its model (19 of 223 outside the sample). Most of the eligible sessions it never logged ended in one week while it was reading an emptied session address, and later runs did not reach them before they left its lookback; most of the logged sessions it did not send were two-message sessions whose text is almost all on the user side, which its skip rule passes over. The replay defect is a fact about our instrument; the reach figures describe this deployment in this window.

1. Question and design

Paper 3 found, on LongMemEval, that replacing C0 with C2 raised answer accuracy by +24.8 to +27.3 pp across two judge families. That benchmark is personal chat; this system's memory is built from agent work sessions. The question here was whether C2 extracts a more useful memory than C0 from the same production sessions, asked offline, on sessions frozen before any model call.

The pre-registration (`PREREGISTRATION.md`, tag `paper-d-prereg-v1.0`) fixes the frame, the draw, the question writer and its validation, the arms, the reader, two judges from different model families, a blind human sample, the controls and the reading. Its scripts refuse to run a step unless the previous step's commitment is on the public main branch, written once, and byte-equal to the private copy. The four commitments are `FREEZE-COMMITMENT.json`, `QUESTIONS-COMMITMENT.json`, `HUMAN-COMMITMENT.json` and the reading itself (`READING-2026-11-06.json`).

The arms that matter here are four. C0 and C2 distil the session's complete prose under each criterion. P serves the memory production stored for the session. V replays C0 on the part of the session production could have seen when it ran (the turns before its processing time), and is empty when production's log says the session had fewer than 3 messages, which production skips without calling the model. P against V is the fidelity control: if our replay is a faithful copy of production's C0, the two should answer the same questions about equally well.

2. What ran

The draw took 80 sessions from a frame of 418. A writer model produced 255 primary questions (decisions, preferences, lessons, context) and 221 secondary ones (identifiers and passages) from windows of each session, each with a verbatim citation; 67 primary and 16 secondary facts were dropped by the registered validation. The model runs cost US$ 0.73. Some distillation calls returned no text. They cost the C2 arm six blocks and the C2-cut arm three (a block with nothing to cut is the same call in both arms), and C0 and V none; they are counted, as registered, not redone.

The human grader marked 60 rows drawn at random from the C0 and C2 answers, without seeing the arm or the judge's verdict. Agreement with judge 1 was 0.850 and Cohen's kappa 0.692. In the 9 disagreements the human accepted an answer the judge rejected; the judge never accepted one the human rejected.

3. The controls

control	measured	passes when	value	result
ceiling	arm A (the evidence itself), primary, judge 1 (judge 2: 0.600, reported, not a gate)	≥ 0.70	0.749	pass
positive	C2 with every evidence region cut out	≤ 0.10	0.024	pass
counterfactual	answers that repeat the original number after it was altered	$\leq 10\%$, and fewer than with the altered number	1 of 60 against 16 of 60	pass
fidelity	P against V, 105 primary questions in the 30 sessions production sent to its model	differ by ≤ 0.05	0.248 against 0.438	**fail** (0.19)

Over all 80 sessions, P and V were 0.102 and 0.180; the registered figure is the one on the 30 sessions production sent, because elsewhere both arms are empty by the same rule.

4. Diagnosis of the fidelity-control failure

The replay calls the same model with the same prompt, the same generation settings, the same 2,000-character blocks, the same noise filter and the same importance threshold as the deployed code, and on the 30 sessions it produced 517 memory units where production stored 244. The 244 equal the count production logged for those sessions, so no stored unit was lost between ingest and the freeze; production's semantic deduplication, which runs before that count, removed between 0% and 15% of candidates in each of the eight weekly runs in its log (`DISTILL-LOG-2026-11-06.json`). Neither explains a factor of two. The difference is in the input. Production's message parser skips any message whose text begins with `[cron:`, which is how a scheduled job's prompt arrives in the transcript. Our replay built its input from the transcript's cleaned turns, which keep that prompt. Rebuilding, line by line, the text production's parser would have sent for the same 30 sessions (`d-fidelity-diagnosis.py`) gives 66,290 characters in 48 blocks; the replay sent 367,003 characters in 76 blocks. Of the replay's text, 301,441 characters (82%) were scheduled-job prompts, present in 28 of the 30 sessions. The replay's other turns hold 59,759 characters of message text, against 66,290 characters in production's blocks. The two are not the same measure (block characters include the role tags and line breaks), and we did not rebuild the replay's blocks without the prompts and rerun it, so how much of the 0.19 difference the prompts account for is not measured.

We had checked the replay against production before the run, but by the wrong quantity. Our recount of messages matched the count production logged in 78 of the 80 sessions, because production counts the job's prompt as a message. It drops the prompt later, after the count is written. A match on counts did not establish a match on content.

5. What the failure exposed

The same prompt shaped the questions. 68 of the 80 sessions contain a scheduled-job prompt, and 138 of the 255 primary questions (54%) cite only text inside it; 116 cite none of it and one cites both. A job's prompt is the job's standing instruction, and production's parser drops it before distilling. For more than half of the primary set, the study measured whether a memory keeps text that production's distiller never reads.

A census under production's input rule (`FRAME-CENSUS-V11-2026-11-06.json`), outside the 80 sessions already used, asked how many sessions a corrected study could draw on. Of 558 sessions eligible under the original rule, 335 fall below the 3,000-character minimum once production's parser rule is applied (it drops scheduled-job prompts, heartbeat messages and the history header), leaving 223. Production's distiller had logged 80 of those 223 and sent 19 to its model (8.5%).

Splitting reach by week shows where the gap comes from (`DISTILLER-REACH-2026-11-06.json`). That split counts every eligible session in the window except the 20 spent on an earlier study, so it holds 245 sessions, 22 more than the census, which also set aside the 80 drawn and 5 smoke sessions; the 22 are those of the 85 that still pass the rule. Until 2026-09-08 the distiller read a session address that a migration had emptied; the fix ran first on 2026-09-13. The distiller runs in two places (`DISTILLER-SCHEDULE-2026-11-06.json`): a weekly run on Sundays that looks back 30 days and takes at most 50 sessions, and a daily step of an agent's end-of-day job that looks back one day. A third job's prompt names the distiller only to forbid running it, and the log, from 2026-09-13 to 2026-11-06, has no rows outside the hours of those two runs. Only the weekly run can reach a session more than a day old, and every one of its eight logged runs started with a full list of 50. Of the 145 eligible sessions it never logged, 118 ended in the single week of 2026-08-23, while the address was empty. The weekly runs that followed were full on every run and did not reach them before they left the 30-day lookback; the log is consistent with that account but does not show which sessions each run chose. For sessions ending from 2026-09-20 on, it logged 65 of 71 and missed 6.

Logging a session is not sending it to the model. Of those 65, 18 were sent and 47 skipped for having fewer than 3 messages; 40 of the 47 had exactly 2. Of those 40, 39 come from one agent, and in their recorded text the user turn is long (median 7,215 characters) and the reply short (median 72), so 99.2% of their characters are on the user side. That fits scripted task invocations, whose text is mostly instruction; we did not classify their content, so this note does not say what they would have yielded if distilled.

6. The registered contrasts, not interpreted

The pre-registration requires every number to be reported when a control fails. Under judge 1, C2 − C0 on the primary set was −0.043 [−0.113, +0.028] over 255 questions in 80 sessions; under judge 2, −0.047 [−0.110, +0.015]. On the secondary set, 221 questions in 79 sessions, it was +0.140 [+0.077, +0.206] under judge 1 and +0.149 [+0.088, +0.213] under judge 2. These are 95% cluster-bootstrap intervals over sessions. They are reported, as the pre-registration requires, and not interpreted: they compare the criteria on input that production does not send, on questions that are mostly about that input, and the control that would have certified the replay failed.

7. What this note claims, and what it does not

It claims three things. The registered study is an instrument failure, and a defect that plausibly accounts for it is located: a replay input that kept the scheduled-job prompts production drops. How much of the failure it accounts for was not measured. More than half of the primary questions were about that prompt. And on this deployment the production distiller logged few of the eligible sessions, most of the missing ones from one week while its session address was empty; for sessions ending since 2026-09-20 it logged 65 of 71, and most of the logged ones it did not send were two-message sessions with almost all their text on the user side.

It does not claim that C2 is better or worse than C0 on production sessions; this study cannot say. It does not claim that the low reach is a standing property of the deployment. It does not claim that the sessions the rule skips hold nothing worth keeping, nor that the 0.19 fidelity gap would close without the prompts. A weekly cap that fills on every run could leave a new backlog behind a future outage, under either criterion. Since 2026-11-06, after these measurements, the distiller takes the oldest pending sessions first and logs how many it left for the next run; that is a change to the deployment, outside this note's data.

8. Two things that transfer

When a replay is checked against the system it copies, the check has to compare what the system read, not how much of it. Here the counter ran before the filter, so matching the counter reproduced exactly the part of the code that did not matter. Rebuilding the input text from the deployed code and comparing characters and blocks would have caught the defect before any paid call; the whole run, including that check, would have cost the same.

A question set inherits whatever its source text over-represents. Locating each question's citations by the kind of turn they fall in took one pass over data we already had, and would have shown, before the run, that the set was mostly about the scheduled-job prompts.

9. Data availability

The sessions, the questions and the human sheet are private and are not released. The scripts, the four commitments, the reading, the diagnosis, the census and the reach split are in the public repository under `paper-d-c2-production/`; each carries only counts, rates and hashes, so a reader can check that the files are the committed ones but cannot rerun the analysis without the private data. Every result with a decimal, a percentage or a thousands separator has an entry in `PROVENANCE.json` naming its artifact, a locator string that must occur in it, and the commit that holds it; counts below that and design constants are in the artifacts named in each section.