StateCross: A Benchmark for Memory and State Tracking in Long-Horizon Agents

Anonymous authors
Paper under double-blind review

Abstract

Long-horizon tasks require agents to follow instructions that may change across sessions. An instruction may be withdrawn, take effect later, or apply only to one part of a task. Agents must therefore remember information and decide which requirements govern their next action. Existing memory evaluations often focus on recall or answer consistency. These measures do not by themselves show whether retained instructions lead to correct actions. Evaluations centered on memory can also overlook workspace files, which preserve instructions and earlier results after the conversation is cleared. We introduce StateCross, a benchmark for evaluating whether retained information supports actions that follow current requirements. Controlled coding tasks combine changing requirements with context resets while workspace files and explicit memory persist. Agents select from predefined programs, whose executed effects are checked against the requirements in force. We compare agents with and without explicit memory using decision accuracy, requirement-specific errors, and consistency across consecutive actions. The framework also distinguishes evidence gaps at storage, retrieval, and model input, supporting separate analysis of information availability and action correctness.

1 Introduction

Agents working on long tasks need to use instructions from earlier conversations. Users may later change those instructions, delay a change, or make an exception for one part of the task. The agent must decide which instructions to follow at each step and keep following them when work resumes in a later session. We study whether the information an agent retains can support these decisions as requirements change.

Consider an assistant arranging a weekly meeting, normally held at 10 a.m. in Room A. The user asks to move the meeting to 11 a.m. starting next week, then adds that this week's meeting should use Room B. The assistant should arrange this week's meeting at 10 a.m. in Room B and next week's at 11 a.m. in Room A. Remembering both messages is not enough if it changes the time too soon or keeps using Room B in later weeks. It must apply each change to the meetings for which it was intended.

Existing benchmarks test memory in several ways. LongMemEval and MemoryAgentBench check whether agents can use information from long conversations and handle updates (Wu et al., 2025; Hu et al., 2026). StateMemBench checks whether answers reflect current rather than outdated information. MemoryArena tests tasks whose later sessions depend on earlier work (Fan et al., 2026; He et al., 2026). These evaluations address recall, updates, and task performance. To assess continued adherence, we also need to check each executed action against the requirements in force at that step.

Focusing on memory alone can also overlook information preserved in project files, which we call the workspace. The meeting assistant, for example, could read a saved schedule instead of recovering the meeting time from memory. Clearing the conversation would not remove that schedule. A correct action therefore need not depend on an explicit memory system. To evaluate what memory adds, we compare agents that retain the same kinds of project files with and without explicit memory.

We introduce StateCross to study these questions through controlled coding tasks (Figure 1). Requirements change over several work sessions. Between sessions, we clear the recent conversation but keep workspace files and records saved by an explicit memory system. The agent chooses among

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.

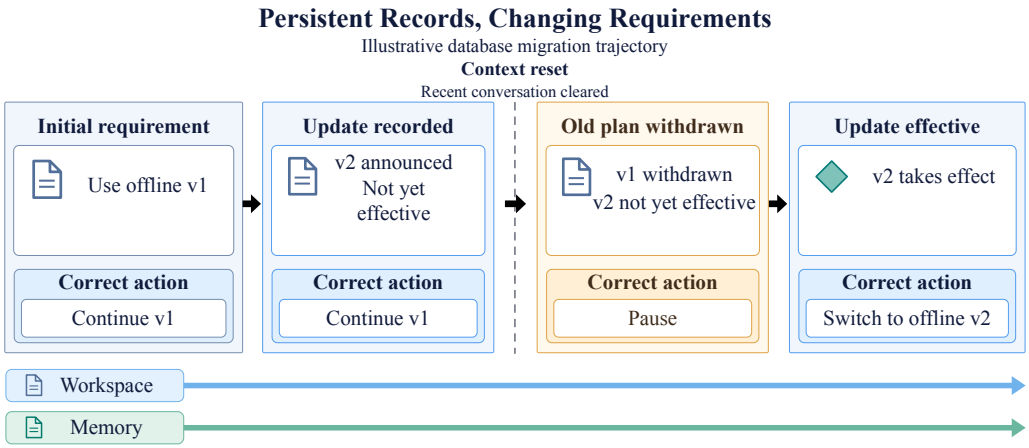

Figure 1: Changing requirements in a StateCross coding task. Announcing v2 does not make it effective. The agent should continue v1 while it remains valid, pause after its withdrawal while v2 is pending, and switch when v2 takes effect. Workspace files and memory persist across context resets. The diagram illustrates expected actions, not observed model performance; scoped permissions are described in Section 3.2.

predefined programs, and we run the selected program to check whether it follows the current requirements. Alongside the workspace-only comparison, the framework defines checks for whether required information was stored, retrieved, and included in the model input. We also check whether the agent continues to follow requirements that it followed earlier. Our contributions are as follows.

- We introduce StateCross, a benchmark for following changing requirements across sessions. Its tasks specify when an instruction takes effect, when it is withdrawn, and which operations it applies to, and check the effects of the agent's chosen program.
- We provide measures of decision accuracy and continued adherence, together with a failure taxonomy and evidence tracking. These distinguish missing information from wrong actions when the required information reaches the model.
- We compare memory systems with agents that retain project files but have no explicit memory system. Separate information-condition comparisons and exploratory diagnostics examine how available records relate to action correctness within the tested tasks.

Section 2 reviews related work. Section 3 presents the benchmark, and Section 4 describes the experiments and metrics. Section 5 reports the results. Sections 6 and 7 discuss their implications and limits, and Section 8 concludes the paper.

2 Related Work

Related work spans memory benchmarks, memory use in agent tasks, memory systems, and failure diagnosis. We review these areas in that order to connect what is evaluated with how memory is used and inspected.

Memory and state benchmarks. LoCoMo evaluates memory over long multi-session conversations through question answering, event summarization, and dialogue generation (Maharana et al., 2024). LongMemEval includes knowledge updates and temporal reasoning (Wu et al., 2025). MemoryAgentBench tests incremental learning and conflicting-fact consolidation (Hu et al., 2026), while MEMTRACK uses evolving records from multiple platforms (Deshpande et al., 2025). StateMemBench scores answers against current and superseded states, including scoped overrides and their expiry (Fan et al., 2026). These benchmarks evaluate recalled information and updated answers.

Memory in agent tasks. MemoryArena evaluates interdependent sessions through environment interaction and task constraints (He et al., 2026). Mem2ActBench scores tool selection and parameter grounding from stored information without executing the generated calls (Shen et al., 2026). AMA-Bench evaluates questions over recorded agent trajectories and reports separate downstream execution experiments (Zhao et al., 2026). Executable software evaluation also has a foundation in SWE-bench, which tests generated repository changes (Jimenez et al., 2024). Table 1 compares the primary protocols of the memory benchmarks.

Memory systems. Generative Agents combines experience records, reflection, and planning, while Reflexion retains feedback for later attempts (Park et al., 2023; Shinn et al., 2023). MemoryBank updates long-term conversational memory, and MemGPT manages information across context tiers (Zhong et al., 2024; Packer et al., 2023). Mem0, A-MEM, and MemOS offer different approaches to extracting, organizing, and maintaining records (Chhikara et al., 2025; Xu et al., 2025; Li et al., 2025). GraphRAG and LightRAG use graph structure for retrieval (Edge et al., 2024; Guo et al., 2025); StateMem explicitly tracks supersession (Fan et al., 2026). We compare selected systems under a shared continuation policy.

Information use and diagnosis. Lost in the Middle shows that models' use of supplied information depends on its position in the context (Liu et al., 2024). MemTrace traces memory operations and attributes errors through execution graphs (Deng et al., 2026). These studies motivate inspecting information at intermediate stages alongside final scores. A coverage gap alone does not establish the cause of an action error.

StateCross evaluates how information retained in workspace and memory supports continued adherence. We check executed actions against requirements with explicit effective times and scopes, and trace required evidence to distinguish coverage gaps from behavioral errors.

Table 1: Capabilities explicitly evaluated by each benchmark's primary protocol: ✓ = included; × = not an explicit evaluation target. History use includes recall and history-dependent actions; state updates include changing facts, plans, or environment state.

Benchmark	History use	State updates	Timed rules	Scoped rules	Execution checks	Adherence drift
LongMemEval	✓	✓	×	×	×	×
MemoryAgentBench	✓	✓	×	×	×	×
StateMemBench	✓	✓	✓	✓	×	×
MEMTRACK	✓	✓	×	×	×	×
MemoryArena	✓	✓	×	✓	✓	×
Mem2ActBench	✓	✓	×	×	×	×
AMA-Bench	✓	✓	×	×	×	×
StateCross (ours)	✓	✓	✓	✓	✓	✓

Timed rules have effectiveness or expiry conditions; timestamps alone do not qualify. Scoped rules distinguish shared requirements from local constraints or exceptions. Execution checks score effects of the evaluated agent's actions. Adherence drift counts violations after earlier-session adherence. AMA-Bench's supplementary execution tests are outside its primary QA protocol.

3 Benchmark Design

3.1 Information sources and failure taxonomy

We distinguish three sources of information. The *workspace* contains project files and outputs that remain available between sessions. An explicit *memory* system saves records from earlier work, while *history* is the recent conversation that we clear between sessions. Required information may be written directly in a file, inferred from file contents, or missing from the workspace. Saved records may also be out of date or apply only to another operation. Figure 2 separates the condition of these sources from the agent's behavior. We track three stages of memory access. *Storage* means the information was saved, *retrieval* means the memory system returned it, and *exposure* means it was included in the model input. We call a wrong action with all required evidence exposed a utilization failure. These labels distinguish evidence availability from the correctness of the resulting action.



Figure 2: Conceptual failure taxonomy after a context reset. The diagnostic dimensions can overlap. The calibration assigns a separate label for the earliest missing evidence stage.

3.2 Task structure

Each task is a coding project spread over 16 work sessions. A complete run through these sessions is a *trajectory*. Project files and memory records persist between sessions, while the recent conversation is cleared. We refer to this clearing step as a *context reset*.

The task generator produces instructions, project files, and requests to continue the work. Instructions can change during the task. Each requirement has an *effective time*, when it starts to apply, and a *scope*, the operations it applies to. A new requirement may replace or withdraw an earlier one. The agent receives the project information but not the evaluator's correct-action labels.

For each decision, the evaluator processes requirement changes in order and selects those that apply to the current operation. Let $E_{\leq s}$ denote events effective through session s, and q_d the scope of decision d:

$$C_s = \text{Replay}(E_{\leq s}), \qquad S_d = \text{Applicable}(C_{s(d)}, q_d). \tag{1}$$

Here C_s contains the requirements after processing the changes, and S_d contains those governing decision d. Replay applies additions, replacements, scope changes, and withdrawals in session order, processing prerequisite changes first within a session. The evaluator uses S_d for scoring; it is not automatically shown to the agent. An announced change does not apply before its effective time. If the old plan has been withdrawn and the new one is not yet valid, the agent should pause.

The coding tasks test whether an agent can follow a task-wide rule while applying a local exception. We use a default offline rule for this purpose. An online service is allowed only when an instruction explicitly authorizes it for the current operation. Other operations must remain offline. Offline execution is a task-design choice, not a requirement of long-horizon memory evaluation. The exception is analogous to changing the meeting room for this week's meeting only. We label it local-cloud, where local refers to the permission's limited scope and cloud to the online service. The four candidate programs correspond to the following choices and conditions.

1. **Continue the old plan (old-valid).** Use version v1 while it remains valid.
2. **Switch to the new plan (new-global).** Use version v2 once it takes effect, while following the task-wide offline requirement.
3. **Use the permitted online service (local-cloud).** Apply the exception only to the operation it authorizes.
4. **Pause.** Wait when v1 has been withdrawn but v2 is not yet valid.

Uniform random choice achieves 25% accuracy. We execute the selected program against the project files and check its effects against S_d. Starting v2 too early or using an online service outside the permitted operation fails the check. The evaluated capability is selecting a program that follows the current requirements.

3.3 Behavioral checks and evidence tracing

Continued adherence. Each trajectory contains 256 decisions over 16 sessions and 15 context resets. We check individual choices and groups of four consecutive task steps called cycles throughout the run. These measures describe action-level adherence and local consistency. Cross-session drift counts a violation only if the agent followed the same requirement in an earlier session. We track a requirement and its revisions as a *requirement lineage* so that unrelated instructions are not mixed. Table 2 defines the measures.

Available information. Workspace-only is the condition without an explicit memory system. Tasks vary in file information, content, and requirement timing. A *fixed-decision* experiment compares inputs at the same saved task step, keeping the workspace, candidate programs, and checker unchanged. Evidence tracking labels the first memory stage missing required information. Replacement tests change a record in the input and check whether the action changes. Appendices A and F.3 describe the fixed-decision protocol and archived diagnostic summaries.

Construction checks. Regeneration reproduces 12 archived instances; 48 dependency checks and 100 predefined fault cases return their expected outcomes (Appendix C). These checks validate construction and scoring behavior. Model performance under directly supplied current state is reported separately in Section 5.2.

4 Evaluation Setup

4.1 Experimental design

The main experiment uses nine instances of one coding-task template, spanning five narrative scenarios, with 16 sessions and 256 decisions per instance. DeepSeek V4 Pro chooses actions under seven conditions: workspace-only and six memory systems. Each system runs once per task, yielding 63 complete trajectories and 16,128 decisions. The main table uses the same nine tasks and API route for every system. LightRAG lacks complete coverage of these nine tasks; its completed six-task results are reported separately. A shared reranker selects up to five returned records for the model input. Appendix B documents the adapters and configurations; Appendix D.4 reports the available ten-task extension.

A separate six-task experiment shows or hides workspace-file contents. An archived five-task GPT-5.6 Sol calibration compares information inputs at fixed decisions. Its diagnostic summaries are analyzed separately from the main trajectories (Table 7).

4.2 Metrics

Table 2 summarizes the measures. Appendix A.3 gives their formulas, and the cited tables specify what is counted in each rate. We give each task equal weight. For the main comparison, we estimate uncertainty by resampling whole tasks 10,000 times while keeping each system paired with workspace-only on the same task (Appendix A.4).

5 Results

We first examine whether agents keep following changing requirements. We then compare the information supplied to them and track where required evidence is missing. Section 5.1 uses the complete nine-task comparison. Section 5.2 reports system gains, the separate file-visibility experiment, and GPT fixed-decision comparisons. Section 5.3 examines the archived GPT diagnostic summaries.

Table 2: Evaluation measures, grouped by outcome and diagnostic stage.

Measure	Definition
Task performance	
Decision accuracy; paired gain	Correct-action rate; difference from workspace-only on the same tasks (Eq. 3).
Balanced accuracy B; sensitivity B_3	Equal mean of four condition accuracies; B_3 omits old-valid (Eq. 4; wording sensitivity in Section 7).
Checker score	Mean executable-check outcome, allowing partial credit.
Cycle accuracy; longest correct run	Fully correct four-action cycle rate; longest consecutive correct-action run (Table 15).
Behavioral failures	
Action-error rates	Overlapping decision tags: future adoption, stale selection, constraint violation, scope overreach, and plan deviation.
Cross-session drift	Violations of a requirement followed in an earlier session; rates over eligible decisions or calibration counts (Eq. 5).
Memory diagnostics	
Write coverage; selectivity	Fraction of information eligible for writing that new records capture; fraction of new records containing such information (Tables 22, 23).
Storage precision, recall, F1	Fraction of stored records containing current information; fraction of current information represented; their harmonic mean (Eq. 6).
Stale retention; update response; physical retirement	Outdated information still stored; updates represented by replacement or removal; old records removed (Table 23).
Retrieval; exposure coverage	Fraction of stored required items retrieved; fraction of retrieved required items shown to the model (Eq. 7).
Evidence-stage class	First incomplete stage or complete evidence per memory-reliant decision, independent of action correctness (Eq. 2).
Replacement effect; flip rates	Changes after record replacement that survive repeat checks; action changes under the primary and neutral-control replacements (Appendix F.3).
Added-record sensitivity	Action-change rate after appending irrelevant records (Table 25).
Source shares; memory volume	Sources of input records; amounts stored, returned, and shown to the model; frequency of empty memory (Tables 14, 13).

5.1 RQ1. Do actions remain compliant as requirements change?

Decision accuracy. Accuracy ranges from 35.42% to 47.40% across the evaluated configurations (Table 3). Workspace-only reaches 38.41%, while always choosing the new offline plan matches 45.83% of the correct-action labels but gives only 25% balanced accuracy (Appendix Table 12). We therefore report condition-specific accuracy and local consistency alongside overall accuracy.

Table 3: Complete nine-task comparison, excluding deferred LightRAG. Rates are percentages; gains and exploratory 95% paired-task bootstrap intervals are percentage points over workspace-only. Each system has one trajectory on each of the same nine tasks. The old-valid condition is marked in Table 4.

System	Acc.↑	Gain [95% CI]	Bal.↑	Constraint↓	Scope↓	Cycle↑
Workspace-only	38.41	n/a	37.06	35.42	17.40	28.12
Flat-BGE	42.36	$+3.95$ $[+0.95, +6.81]$	40.35	26.82	15.80	34.55
BM25	36.98	-1.43 $[-5.90, +2.43]$	35.66	30.38	18.14	26.91
GraphRAG	45.70	$+7.29$ $[+2.69, +10.20]$	41.12	20.49	13.63	37.15
Mem0	35.42	-2.99 $[-9.24, +2.60]$	36.51	31.64	20.10	27.08
A-MEM	39.28	$+0.87$ $[-2.73, +4.60]$	37.46	29.86	15.15	30.38
MemOS	47.40	$+8.98$ $[+5.03, +13.02]$	44.33	18.14	14.32	38.54

Requirement conditions. The largest differences occur when the new global plan applies (Table 4). MemOS and GraphRAG reach 61.55% and 60.01%, compared with 43.14% for workspace-only. All systems perform much better on local-cloud decisions than on waiting for a pending plan. Old-valid has 17 correct choices among 4,144 decisions across the seven systems. We report B_3 over the other three conditions alongside the full four-condition measure; Section 7 discusses the old-valid wording.

Table 4: Action accuracy (%) by governing requirement, averaging the nine tasks equally. Counts per system are 592 old-valid, 224 pause, 1,056 new-global, and 432 local-cloud decisions. B averages four conditions; B_3 excludes old-valid. †: wording-sensitive condition (Section 7).

System	Old valid†	Pause	New global	Local cloud	B	B_3
Workspace-only	0.80	12.85	43.14	91.44	37.06	49.14
Flat-BGE	0.68	16.32	50.20	94.21	40.35	53.58
BM25	0.23	13.54	40.68	88.19	35.66	47.47
GraphRAG	0.31	12.50	60.01	91.67	41.12	54.73
Mem0	0.37	16.67	38.74	90.28	36.51	48.56
A-MEM	0.41	13.89	45.95	89.58	37.46	49.81
MemOS	0.14	24.65	61.55	90.97	44.33	59.06

Local consistency. Fully correct four-action cycles reach 38.54% for MemOS and 37.15% for GraphRAG, compared with 28.12% for workspace-only. These gains indicate better adherence across short sequences of actions.

5.2 RQ2. How do information conditions change action selection?

Explicit memory. Mean gains range from -2.99 to $+8.98$ percentage points, averaging $+2.78$ across the six memory systems. MemOS has the highest mean accuracy and improves on workspace-only in eight of nine tasks; GraphRAG also improves in eight. GraphRAG exceeds MemOS on five tasks, and removing task 3010 reverses their mean ordering. MemOS gains 18.40 points on new-global decisions but loses 0.46 on local-cloud decisions. Its exploratory Holm-adjusted p is .047 across the six reported comparisons; it is .055 if the original seven-comparison family is retained conservatively. Task-level effects and sensitivity results appear in Appendix A.4. Cross-model comparisons remain separate (Appendix E).

Workspace-file visibility. A separate paired run on the six tasks compares visible file contents with those contents hidden from the policy (Appendix D.5). Accuracy is 50.91% versus 50.00%, a difference of $+0.91$ points with an exploratory 95% task-bootstrap interval of $[-3.84, +5.66]$. The execution workspace, current-session information, and candidate actions remain available in both conditions. The paired difference uses fresh runs of both conditions.

History and current state. The five-task GPT calibration compares inputs at the same saved decisions (Table 5). Workspace-only selects correctly in 41/90 decisions, full history in 78/90, and a direct statement of the current requirements in 88/90. Thus, stating the current requirements helps more than supplying the full history in this experiment. Appendix Table 21 reports the corresponding input diagnostic across policy models.

Table 5: Information-condition ablation with GPT-5.6 Sol on five calibration tasks and 90 fixed decisions per condition. Full history and oracle state supply the complete history and correct current state. Exact counts correct program choices; Score is the mean checker score. Violation counts decisions flagged for constraint violation, plan deviation, stale-state selection, or scope overreach, once per decision; other incorrect choices are excluded. Drift additionally requires prior adherence on the same requirement lineage.

Condition	Score	Exact	Violation	Drift
Workspace-only	.6333	41/90	34	29
Flat retrieval	.6333	41/90	34	27
Mem0	.8222	67/90	18	11
A-MEM	.6083	38/90	37	30
MemOS	.7472	57/90	25	19
Full history	.9000	78/90	12	0
Oracle state	.9833	88/90	2	0

Workspace-only has 29 violations after earlier-session adherence in the GPT calibration, versus zero under full history and explicit current state. Those conditions still have 12 and two pointwise viola-

tions, respectively. Drift identifies violations of requirements the agent followed in an earlier session. Its eligibility depends on earlier adherence, so these counts are read alongside action accuracy.

5.3 RQ3. What do evidence diagnostics add to action accuracy?

Evidence coverage. Retained summaries from the GPT calibration describe 54 decisions per system that require information from memory. Appendix B documents the archived summaries used for this exploratory analysis. A required-state occurrence pairs one needed piece of information with one decision; repeated needs are counted again. Across four systems, 488 occurrences reduce to 467 stored, 254 retrieved, and 135 included in model input (Table 6). Retrieval accounts for 213/353 (60.3%) of missing occurrences, exposure for 33.7%, and storage for 5.9%. These reconstructed counts concern missing information regardless of action correctness.

Table 6: Required-state occurrences reconstructed from archived GPT calibration summaries, across 54 memory-reliant decisions per system. A state is counted again whenever another decision requires it. MemOS storage counts rely on inferred matches between memory records and required information. Full lifecycle measures appear in Appendix Table 22.

System	Required	Stored	Retrieved	Exposed
Flat retrieval	122	120	40	13
Mem0	122	120	117	75
A-MEM	122	120	40	13
MemOS	122	107	57	34
Total	488	467	254	135

System-specific gaps. The retained stage summaries classify each decision by the first stage missing any required information. Flat retrieval and A-MEM each have 38 retrieval gaps among 52 decisions with all required information stored. Mem0 has 22 exposure gaps among 49 decisions with all required information retrieved. For MemOS, inferred record matches indicate 14 storage gaps and 33 retrieval gaps among the remaining 40 decisions. These aggregates locate the evidence gaps at different stages for each system.

Final record selection. In the retained GPT calibration scorecards, shared reranking raises MemOS accuracy from 45/90 to 57/90 relative to native selection. Mem0 and A-MEM instead decline by 8.89 and 6.67 points (Appendix Table 24). These comparisons suggest sensitivity to the records presented to the policy. The intervention changes record selection and ordering together. Appendices D.3 and F.3 report source attribution and historical replacement controls.

6 Discussion

Workspace as a baseline. Workspace-only provides a reference for memory gains while retaining the task request, candidate programs, and current-session information. The paired visibility experiment measures the contribution of file contents within this input, with a mean difference of $+0.91$ points across six tasks.

System differences. MemOS combines the highest new-global and pause accuracy with the lowest constraint-error rate. The system comparison evaluates the complete writing, retrieval, and selection pipeline. GraphRAG has the lowest scope-error rate, while 98.13% of its exposed source records describe the agent's own earlier execution. Flat-BGE outperforms BM25 with the same record interface and shared selection; Mem0 exposes no memory in 29.17% of decisions, and A-MEM yields a small mean gain despite nonempty retrieval in most sessions. Together, these measurements characterize both the supplied records and the resulting actions.

Memory integration. Our adapters standardize the policy-facing interface, map graph retrieval back to source records, and apply a common final reranker with a five-record limit. These changes

make record selection explicit and auditable. The archived fixed-decision scorecards report a 13.33-point MemOS gain from common selection over native selection, with declines for Mem0 and A-MEM. This pattern highlights the role of the final record-selection step in configuring memory systems.

Locating evidence gaps. Separate storage, retrieval, and exposure checks identify stages to inspect even when action scores are similar. The main traces verify which records were presented, while the archived calibration illustrates required-state coverage from retained summaries. Workspace information may compensate for missing memory records, and complete evidence may still accompany a wrong action. Reporting coverage alongside behavior distinguishes missing information from errors made with complete evidence.

7 Limitations

The main comparison uses nine instances of one template, spanning five narrative scenarios, with one run per system and task. Its intervals describe task variation, not repeated-run stability or generalization to new task families. The old-valid candidate is described as withdrawn even while it remains valid. This confounds temporal tracking with wording and may affect later workspaces; excluding that condition from a summary cannot repair the trajectories. Neutral candidate wording and complete trajectory reruns are needed to resolve this limitation. LightRAG coverage is incomplete, and the ten-task extension includes mixed API routes. System interfaces and token or compute budgets differ, and record-selection interventions also change ordering. The historical memory inventories, retrieval traces, and paired diagnostic inputs were not recovered. The archived summaries therefore provide exploratory diagnostics; system-level associations do not isolate causal memory mechanisms.

8 Conclusion

StateCross evaluates actions under changing requirements across sessions while distinguishing information in workspace files and explicit memory. The reported comparisons describe decision accuracy and local consistency within the archived task interface. Source-record traces and exploratory coverage summaries motivate examining available evidence alongside behavior. Together, these measures connect changing task requirements, retained information, and the actions an agent takes.

Ethics Statement

The benchmark uses constructed coding tasks and trusted predefined candidate programs in isolated execution environments. The experiment studies continuation under changes in task requirements and records the resulting local software artifacts.

Reproducibility Statement

The artifact contains generators, checkers, result identifiers, source hashes, and analysis scripts. The updated analysis records hashes for 69 completed non-LightRAG trajectories and independently reconstructs all 17,664 action labels, including the 16,128 labels in the balanced nine-task panel. Deterministic construction checks and the independent 12-trajectory file-visibility comparison remain separate. Archived calibration scorecards support the supplementary diagnostic analysis (Appendix B).

AI Use Statement

Generative AI tools assisted literature discovery, the development of the evaluation framework, experiment design, implementation and code review, result analysis, manuscript drafting, and English–Chinese translation. Generative AI tools also assisted the illustrations in Figures 1 and 2. Numerical tables were checked against saved experiment records, and automated checks verified source hashes and reconstructed action labels. The authors are responsible for the final text, methods, measurements, and claims.

REFERENCES

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready AI agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.

Xinle Deng, Ruobin Zhong, Hujin Peng, Xiaoben Lu, Yanzhe Wu, Guang Li, Buqiang Xu, Yunzhi Yao, Jizhan Fang, Haoliang Cao, Junjie Guo, Yuan Yuan, Ziqing Ma, Yuanqiang Yu, Rui Hu, Baohua Dong, Hangcheng Zhu, and Ningyu Zhang. MemTrace: Tracing and attributing errors in large language model memory systems. *arXiv preprint arXiv:2605.28732*, 2026.

Darshan Deshpande, Varun Gangal, Hersh Mehta, Anand Kannappan, Rebecca Qian, and Peng Wang. MEMTRACK: Evaluating long-term memory and state tracking in multi-platform dynamic agent environments. *arXiv preprint arXiv:2510.01353*, 2025. Presented at the NeurIPS 2025 Workshop on Scaling Environments for Agents.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From local to global: A graph RAG approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*, 2024.

Xinyi Fan, Miri Liu, Ruozhen Yang, Siru Ouyang, and Jiawei Han. Can agent memory systems track evolving state? *arXiv preprint arXiv:2608.19652*, 2026.

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. LightRAG: Simple and fast retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 10746–10761. Association for Computational Linguistics, 2025.

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. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Yuanzhe Hu, Yu Wang, and Julian McAuley. Evaluating memory in LLM agents via incremental multi-turn interactions. In *International Conference on Learning Representations*, 2026.

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*, 2024.

Zhiyu Li, Chenyang Xi, Chunyu Li, Ding Chen, Boyu Chen, Shichao Song, Simin Niu, Hanyu Wang, Jiawei Yang, Chen Tang, Qingchen Yu, Jihao Zhao, Yezhaohui Wang, Peng Liu, Zehao Lin, Pengyuan Wang, Jiahao Huo, Tianyi Chen, Kai Chen, Kehang Li, Zhen Tao, Huayi Lai, Hao Wu, Bo Tang, Zhengren Wang, Zhaoxin Fan, Ningyu Zhang, Linfeng Zhang, Junchi Yan, Mingchuan Yang, Tong Xu, Wei Xu, Huajun Chen, Haofen Wang, Hongkang Yang, Wentao Zhang, Zhi-Qin John Xu, Siheng Chen, and Feiyu Xiong. MemOS: A memory OS for AI system. *arXiv preprint arXiv:2507.03724*, 2025.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of LLM agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13851–13870. Association for Computational Linguistics, 2024.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pp. 1–22. Association for Computing Machinery, 2023.

Yiting Shen, Kun Li, Wei Zhou, and Songlin Hu. Mem2ActBench: A benchmark for evaluating long-term memory utilization in task-oriented autonomous agents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8173–8190. Association for Computational Linguistics, 2026.

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. Curran Associates, Inc., 2023.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. LongMemEval: Benchmarking chat assistants on long-term interactive memory. In *International Conference on Learning Representations*, 2025.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-MEM: Agentic memory for LLM agents. In *Advances in Neural Information Processing Systems*, volume 38, pp. 17577–17604. Curran Associates, Inc., 2025.

Yujie Zhao, Boqin Yuan, Junbo Huang, Haocheng Yuan, Zhongming Yu, Haozhou Xu, Lanxiang Hu, Abhilash Shankarampeta, Zimeng Huang, Wentao Ni, Yuandong Tian, and Jishen Zhao. AMA-Bench: Evaluating long-horizon memory for agentic applications. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. MemoryBank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19724–19731, 2024.

Appendix A defines the experiment sets, metrics, and statistical procedures. Appendices B and C describe system configuration and construction checks. Appendix D gives supplementary main-experiment results, followed by model comparisons in Appendix E and calibration diagnostics in Appendix F.

A Additional Evaluation Details

A.1 Experiment sets

A fixed decision comprises a checkpoint, its required state, a workspace snapshot, candidate actions, and an executable checker. Information conditions are compared at that same decision. Continuous trajectories instead retain each system's earlier actions and outputs, which can change later workspaces and requests. Table 7 separates these experiment sets; Table 8 describes the main tasks.

Table 7: Experiment sets used in the reported comparisons. Scale counts scored decisions; trajectory experiments retain each system's earlier actions, while fixed decisions hold the checkpoint, workspace, candidates, and checker constant. Each set is analyzed separately.

Experiment set	Policy model	Tasks	Scale	Question
Main matrix (§5)	DeepSeek V4 Pro	9	63 trajectories; 16,128 decisions	Accuracy and compliance vs. workspace-only
Information calibration (§5.2–5.3)	GPT-5.6 Sol	5	90 fixed decisions per condition	Information conditions and evidence coverage
Model diagnostic (App. E)	DeepSeek, GPT, Opus	1	150 fixed decisions	Responses to matched inputs
Cross-model trajectories (App. E)	GPT-5.6 Sol	9	45 trajectories; 11,520 decisions	Transfer to a second policy model
Common-subset comparison (App. E)	Four policy models	3	256 decisions per system–task cell	Within-model gains across models
Ten-task extension (App. D.4)	DeepSeek V4 Pro	10	60 trajectories; 15,360 decisions	Six complete systems; overlaps the main panel
File visibility (App. D.5)	DeepSeek V4 Pro	6	12 trajectories; 3,072 decisions	Effect of visible file contents

Table 8: The nine main-experiment instances, all from one template. Each has 16 sessions, 15 context resets and 256 decisions. The four rightmost columns count decisions under each governing condition. Scenario and workspace-evidence class come from archived metadata.

Task	Scenario	Workspace evidence	Old valid	Pause	New global	Local cloud
3004	schema migration	derivable	80	32	96	48
3005	benchmark running	absent	80	32	96	48
3006	firmware validation	explicit	80	32	96	48
3007	experiment pipeline	derivable	64	16	128	48
3008	release building	absent	64	16	128	48
3009	schema migration	explicit	64	16	128	48
3010	benchmark running	derivable	64	16	128	48
3012	experiment pipeline	explicit	48	32	128	48
3013	release building	derivable	48	32	128	48

A.2 Evidence-stage assignment

Each fixed decision that needs memory receives one stage label for the earliest missing information. We check storage first, then retrieval, then exposure to the model input. Let $I_{\text{sto}}, I_{\text{ret}}, I_{\text{exp}} \in \{0, 1\}$

indicate that every required state is stored, retrieved, and exposed at that decision:

$$\text{stage}(d) = \begin{cases} \text{unattributable}, & \text{storage evidence unavailable}, \\ \text{storage gap}, & \text{otherwise if } I_{\text{sto}} = 0, \\ \text{retrieval gap}, & \text{otherwise if } I_{\text{ret}} = 0, \\ \text{exposure gap}, & \text{otherwise if } I_{\text{exp}} = 0, \\ \text{complete}, & \text{otherwise.} \end{cases} \tag{2}$$

Decisions without a memory channel, and decisions whose required states do not need memory, are excluded before labeling; unattributable decisions do not enter the storage-gap denominator. Complete decisions subdivide by the checked action and, when probed, by the replacement outcome.

A.3 Metric definitions

Decision accuracy and paired gain. Let y_{mtd} indicate whether system m selects the correct action at decision d of task t. With T tasks and n_t decisions per task,

$$A_m = \frac{100}{T} \sum_{t=1}^{T} \frac{1}{n_t} \sum_{d=1}^{n_t} y_{mtd}, \qquad G_m = \frac{1}{T} \sum_{t=1}^{T} (A_{mt} - A_{\text{workspace},t}). \tag{3}$$

Here A_{mt} is task-level accuracy in percent, so G_m is in percentage points. Fixed-decision checker scores can award partial credit and are reported separately from exact action accuracy.

Balanced accuracy. Each task has four requirement conditions: (1) old version still valid, (2) pause between versions, (3) new version now valid, (4) local permission active. We compute accuracy for each condition separately, then average them equally:

$$B = \frac{A_{\text{old}} + A_{\text{pause}} + A_{\text{new}} + A_{\text{local}}}{4}. \tag{4}$$

This gives each requirement condition equal weight despite different decision counts. A policy correct throughout exactly one condition scores 25%. The old-valid wording is discussed in Section 7. We additionally report the three-condition sensitivity measure $B_3 = (A_{\text{pause}} + A_{\text{new}} + A_{\text{local}})/3$. Both summaries use the same observed trajectories.

Cross-session drift. A violation counts as drift only after adherence to the same requirement lineage in an earlier session. For example, violating an offline requirement after following it before a context reset qualifies. A violation without earlier adherence remains an action error. For task e,

$$D_e = \frac{\text{Drift decisions}}{\text{Eligible decisions}} \times 100\%. \tag{5}$$

Eligible decisions have prior adherence to a currently relevant requirement lineage. Eligibility can differ between systems, so drift must be interpreted together with action accuracy and its denominator. Table 5 reports fixed-decision drift counts rather than comparable rates over all 90 decisions.

Evaluation scope. The reported trajectory measures assess individual actions and local consistency. The stored `terminal_checker_passed` field records correctness at the last decision.

Action-error rates. Five tags mark how a wrong action is wrong. Future-state adoption acts before a requirement becomes effective. Stale-state selection retains a superseded version. Constraint violation runs cloud execution without an applicable exception to the offline requirement, and scope overreach is that violation while a local permission is active. Plan deviation covers premature adoption, stale selection, and pausing outside the valid repair interval. Tags may overlap, and each is counted once per decision. Every error rate is the share of decisions carrying the tag, averaged within task and then across tasks, in the same way as Eq. 3. Lineage-qualified drift additionally requires earlier adherence on the same requirement lineage.

Cycle accuracy and longest correct run. Each trajectory contains 64 consecutive four-decision cycles of inspection, implementation, validation, and recording. Cycle accuracy is the fraction with all four actions correct. Longest correct run is the maximum number of consecutive correct actions. Both measures average tasks equally and describe local consistency.

A.4 Statistical aggregation

Tasks receive equal weight, with systems paired within policy model and task. The main-comparison intervals use 10,000 bootstrap resamples of nine whole paired tasks (seed 20260909); the independent file-visibility analysis uses 20,000. The statistical unit is a whole task trajectory. Exact two-sided sign-flip tests enumerate all 2^9 task-level sign assignments. Holm adjustment covers six reported memory comparisons. Because LightRAG was deferred after data collection began, we also give the conservative result retaining seven comparisons for the smallest p value: $6 \times .0078125 = .046875$ versus $7 \times .0078125 = .0546875$. These analyses are exploratory.

B Memory Integration and Configuration

At session boundaries, the runner supplies public history and execution records to each memory interface. Flat retrieval and BM25 retain source records; Mem0, A-MEM, and MemOS use their integrated writing procedures. GraphRAG and LightRAG return source identifiers that the adapters map to original records. Their generated answers are not supplied to the policy. Writing procedures and backend representations remain system-specific.

Table 9: Memory interfaces and versions specified by the released system profiles and lockfiles. A-MEM is identified by its source-commit prefix. Benchmark adapter revisions are recorded per result cell.

System	Pinned version	Backend / adapter interface
Flat-BGE	Repository adapter	Source records; dense retrieval
BM25	Repository adapter	Source records; Okapi BM25
Mem0	mem0ai 2.0.12	Native extraction; Qdrant
A-MEM	ceffb860f071	Notes and links; Chroma
MemOS	memos 2.0.23	Tree mode; Neo4j
GraphRAG	graphrag 3.1.2	Local search; original sources
LightRAG	lightrag-hku 1.5.7	Mixed search; original sources

The shared profiles use BAAI/bge-m3 embeddings (1,024 dimensions; revision 5617a9f61b02), BAAI/bge-reranker-v2-m3 (revision 953dc6f6f85a), and DeepSeek V4 Pro for memory writing. BM25 uses $k_1 = 1.5$ and $b = 0.75$. In the main experiment, the shared reranker ranks at most 20 returned candidates and exposes up to five records, breaking ties by native rank. Fewer candidates and empty retrievals are retained as observed. Optional Mem0 and LightRAG reranking is disabled before shared selection.

Saved traces identify candidate and exposed record IDs for all 13,824 memory decisions in the nine-task panel. The original API route is shared by every main-panel cell. Provider route, configuration hash, and code revision are retained per result; the ten-task extension includes separately identified official-API reruns. The policy client does not explicitly set temperature or seed. Package pins identify the released configurations.

Calibration records. The archived calibration provides aggregate scorecards. Lifecycle counts are reconstructed from saved ratios; exact and inferred provenance identify the reported matching procedure. Section 7 describes the availability of the underlying diagnostic records.

C Construction Validation

Regeneration reproduces all 12 archived instances examined. Forty-eight dependency cases test valid continuation, missing predecessors, changed artifact contents, and duplicate effects; all return the expected outcomes. One hundred predefined fault cases also reproduce their expected drift classifications. These checks cover task construction and scoring behavior.

In the schema-migration example, a hosted analysis service is permitted for an isolated profiling operation, while migration execution must remain offline. The four candidate programs represent

the old plan, replacement plan, permitted online operation, and pause. Executable checks verify the applicable version and constraints. The archived old-plan wording is discussed in Section 7.

An independent action-ledger audit reconstructs all 16,128 correctness labels in the updated main comparison without disagreement. The archived six-task source audit is also available. These checks establish consistency with archived task events and scoring rules.

D Main Experiment Details

D.1 Task-level results and reference policies

Table 10: Per-task action accuracy (%). All seven systems have 256 decisions on each of the same nine task instances; cells are one trajectory each. Workspace denotes workspace-only.

Task	Workspace	Flat-BGE	BM25	GraphRAG	Mem0	A-MEM	MemOS
3004	29.30	35.55	25.39	38.67	41.80	39.06	35.55
3005	36.33	34.38	20.70	44.53	37.11	29.30	41.80
3006	33.20	39.06	34.77	41.80	33.98	39.84	46.09
3007	43.36	41.41	42.97	50.39	27.73	38.28	48.05
3008	46.48	48.05	39.45	58.20	44.53	48.83	53.12
3009	37.89	45.70	43.36	47.66	37.11	35.55	55.47
3010	37.50	36.72	43.75	27.73	35.16	37.50	48.83
3012	41.41	49.61	43.36	50.00	20.70	37.89	59.38
3013	40.23	50.78	39.06	52.34	40.62	47.27	38.28

Table 8 describes the nine instances. MemOS and GraphRAG each improve on workspace-only in eight of nine tasks. Removing task 3010 gives 47.95% for GraphRAG and 47.22% for MemOS, reversing their mean ordering; every other leave-one-task-out subset keeps MemOS first. The per-task table makes this variation visible.

Workspace-only averages 37.50%, 37.60%, 41.41% over explicit, derivable, and absent workspace-information groups (three, four, and two tasks). The paired file-visibility experiment evaluates the file-content channel directly.

Table 11: Nine-task action-error rates (%). Tags can overlap. Each rate averages decisions within a task, then the nine tasks equally. Definitions follow Appendix A.3.

System	Future	Stale	Constraint	Scope	Plan
Workspace-only	47.09	0.17	35.42	17.40	49.05
Flat-BGE	44.97	0.22	26.82	15.80	46.96
BM25	46.92	0.30	30.38	18.14	48.87
GraphRAG	43.45	0.22	20.49	13.63	45.36
Mem0	45.79	0.26	31.64	20.10	48.05
A-MEM	48.05	0.35	29.86	15.15	50.00
MemOS	40.19	0.22	18.14	14.32	42.53

Table 12: Offline constant-action references on 2,304 unique checkpoints across the nine tasks. Balanced accuracy weights the four conditions equally. These are checkpoint-label references, not additional online trajectories.

Constant choice	Accuracy (%)	Balanced (%)
Always old plan	25.69	25.00
Always pause	9.72	25.00
Always new offline	45.83	25.00
Always local cloud	18.75	25.00

D.2 Memory volume

Table 13 counts stored records before retrieval, returned candidates, and records exposed to the policy. Means include empty retrievals; Total counts repeated exposures. Empty denotes decisions with no exposed memory, while workspace information remains available. Native object counts reflect system-specific storage units. W/T/L counts task wins, ties, and losses against workspace-only.

Table 13: Memory volume and task-level comparisons over nine tasks. Means include all 2,304 decisions per system. Total counts repeated record exposures; W/T/L compares task accuracy with workspace-only. Native storage objects are system-specific.

System	Live	Retrieved	Exposed	Total	Empty (%)	W/T/L
Workspace-only	n/a	0.00	0.00	0	100.00	n/a
Flat-BGE	251.42	18.75	4.69	10,800	6.25	6/0/3
BM25	251.42	18.75	4.69	10,800	6.25	4/0/5
GraphRAG	251.42	17.29	4.66	10,744	6.25	8/0/1
Mem0	4.24	4.24	3.20	7,376	29.17	4/0/5
A-MEM	251.42	18.75	4.69	10,800	6.25	4/1/4
MemOS	205.35	18.70	4.69	10,800	6.25	8/0/1

D.3 Source selection and task variation

Table 14: Nine-task source selection, including repeated exposures. Own/Public count earlier agent execution/public-history records. Mapped is the exposed-ID fraction linked to exact source content. Candidate/Exposed give own-record shares; Age is the mean session gap. N/A denotes unavailable attribution, not absence of own-record use.

System	Own	Public	Total	Mapped (%)	Cand. (%)	Exposed (%)	Age
Flat-BGE	867	9,933	10,800	100	6.23	8.03	5.26
BM25	833	9,967	10,800	100	17.29	7.71	4.64
GraphRAG	10,543	201	10,744	100	98.10	98.13	6.18
Mem0	N/A	N/A	7,376	0	N/A	N/A	N/A
A-MEM	N/A	N/A	10,800	0	N/A	N/A	N/A
MemOS	N/A	N/A	10,800	0	N/A	N/A	N/A

GraphRAG's own earlier execution records account for 98.13% of exposures, compared with 8.03% for Flat-BGE and 7.71% for BM25. Its own-record share is already 98.10% before common reranking. These shares describe the records supplied to the policy. Table 14 marks unmapped sources as N/A.

On task 3010, GraphRAG reaches 27.73% accuracy, below workspace-only at 37.50% and MemOS at 48.83%. Each system reaches this task through its own preceding actions and workspace updates.

Table 15: Local consistency averaged over nine tasks. Longest run counts consecutive correct actions (maximum 256); cycle accuracy counts four-action cycles with every action correct. Earlier cycles need not be correct.

System	Longest correct run	Local cycles correct (%)
Workspace-only	26.44	28.12
Flat-BGE	34.00	34.55
BM25	28.78	26.91
GraphRAG	31.89	37.15
Mem0	28.67	27.08
A-MEM	27.89	30.38
MemOS	37.44	38.54

D.4 Available ten-task extension

Tasks 3004–3013 give two instances of each narrative scenario. Six systems have complete coverage, giving 60 trajectories. No accepted GraphRAG result for task 3011 was available at the analysis cutoff, so that system is omitted here. These observations overlap the nine-task panel and are not independent replication. BM25, Flat-BGE, and Mem0 on task 3011 were rerun through the official DeepSeek API after provider failures; the other cells use the original route. Each rerun used one route throughout. Table 16 identifies the route used for each added result.

Table 16: Ten-task sensitivity for the six complete systems. Accuracy is percent; gain is in percentage points over workspace-only. Official marks the three task-3011 reruns; Original denotes the shared main-panel route.

System	Mean accuracy	Gain	Task 3011	Route on 3011
Workspace-only	38.48	+0.00	39.06	Original
Flat-BGE	43.28	+4.80	51.56	Official
BM25	38.79	+0.31	55.08	Official
Mem0	36.48	-1.99	46.09	Official
A-MEM	40.39	+1.91	50.39	Original
MemOS	47.70	+9.22	50.39	Original

Deferred LightRAG. LightRAG has six accepted trajectories on tasks 3004–3008 and 3010, with 38.35% mean accuracy versus 37.70% for workspace-only on that same subset ($+0.65$ points). At the analysis cutoff, its other four trajectories were incomplete and were neither scored as failures nor included in expanded means. Deferral reflects execution time and completion status, rather than a performance threshold. The result is reported on its completed six-task subset.

D.5 Workspace-file visibility control

This independent DeepSeek V4 Pro run compares visible and hidden workspace-file contents on the same six tasks. Both conditions retain the execution workspace, current-session information, candidate actions, and state hashes. The client does not explicitly set temperature or seed. Each task and condition has one fresh 256-decision trajectory, giving 3,072 decisions overall.

Three tasks improve with visible files and three decline. The mean difference is $+0.91$ points, with an exploratory 95% interval of $[-3.84, +5.66]$ from 20,000 paired-task bootstrap resamples. The comparison uses the fresh workspace-only runs in this experiment.

Table 17: Independent workspace-file visibility control. Action accuracy is in percent; differences are visible minus hidden in percentage points. Each cell is one 256-decision trajectory.

Task	Files hidden	Files visible	Difference
3004	41.80	46.09	$+4.30$
3005	44.92	47.66	$+2.73$
3006	44.14	42.19	-1.95
3007	55.08	53.12	-1.95
3008	57.42	49.61	-7.81
3010	56.64	66.80	$+10.16$
Mean	50.00	50.91	$+0.91$

E Additional Model Comparisons

E.1 Nine-task GPT trajectories

Table 18: GPT-5.6 Sol supplementary evaluation on nine tasks, with 2,304 decisions per system. Score is the mean checker score, gain is relative to workspace-only, and brackets give 95% paired-task bootstrap intervals.

System	Score	Gain [95% interval]
Workspace-only	.3390	n/a
Flat retrieval	.3394	+.0004 [−.0794, .0720]
Mem0	.3767	+.0378 [−.0243, .0933]
A-MEM	.3728	+.0339 [−.0326, .0933]
MemOS	.4162	+.0773 [.0022, .1393]

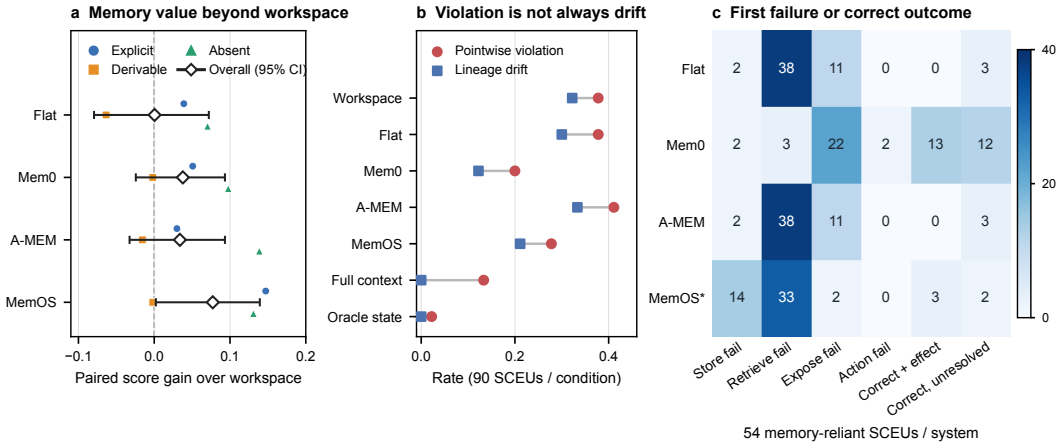

Figure 3: Supplementary GPT-5.6 Sol results, separate from the main DeepSeek comparison in Table 3. (a) Nine-task trajectory gains over workspace-only, grouped by explicit, derivable, and absent workspace information (3, 4, and 2 tasks). Diamonds show means and 95% task-bootstrap intervals. (b) Five-task calibration: violations and lineage-qualified drift on 90 fixed decisions per condition. (c) Earliest missing stage or complete-evidence outcome on 54 memory-reliant decisions per system. SCEU denotes a fixed decision. MemOS* uses inferred provenance; stage labels precede action correctness and come from retained aggregates.

The nine-task GPT comparison includes tasks 3004–3010, 3012, and 3013, with complete five-system pairing. Incomplete task blocks are excluded.

E.2 Matched comparisons across policy models

The three-task comparison uses tasks 3005, 3006, and 3007, one from each workspace-information category. Gains compare each memory system with workspace-only under the same policy and task. Tables 19 and 20 give the mean and task-level gains.

Table 19: Gains in mean checker score (0–1 scale) on the same balanced three-task subset. Each column compares a memory system with workspace-only under that policy model.

Policy model	Flat gain	Mem0 gain	MemOS gain
GPT-5.6 Sol	-.0729	+.0651	+.0638
DeepSeek V4 Pro	+.0156	+.0091	+.0924
Opus 4.8	+.0078	+.0000	+.0091
Qwen3.7 Max	-.0013	-.0586	-.0091

Table 20: Task-level gains in mean checker score over workspace-only, multiplied by 100, for the three supplementary policy reports. Each cell compares 256 decisions within the same model and task. These are the common-subset runs, separate from the nine-task main matrix; GPT aggregate results remain in Table 19.

Policy model	System	3005	3006	3007
DeepSeek V4 Pro	Flat retrieval	+2.34	-0.39	+2.73
DeepSeek V4 Pro	Mem0	+8.98	-2.73	-3.52
DeepSeek V4 Pro	MemOS	+16.41	+1.56	+9.77
Opus 4.8	Flat retrieval	+1.95	+0.00	+0.39
Opus 4.8	Mem0	-0.78	+0.78	+0.00
Opus 4.8	MemOS	+0.39	+2.34	+0.00
Qwen3.7 Max	Flat retrieval	+0.00	+0.39	-0.78
Qwen3.7 Max	Mem0	+0.00	-13.67	-3.91
Qwen3.7 Max	MemOS	-0.39	-11.33	+8.98

E.3 Single-task input diagnostic

This diagnostic uses one task, software-mem0-42, separate from the five-task calibration. Ten input settings at five checkpoints give 50 matched inputs per model and 150 decisions in total. Workspace, history, and memory inputs are matched across models.

All 21 incorrect decisions select v2 at the checkpoint where v1 is still required. That choice passes one of two checks, giving a checker score of .5 despite an incorrect action. Supplying the current state resolves this error for DeepSeek and GPT, but not Opus (Table 21). Changes in record selection also affect some decisions; these comparisons evaluate the complete model input.

Table 21: Fixed-decision ablations across three policy models. Each cell reports mean checker score followed by exact action accuracy (%) in parentheses. There are five decisions per cell, 50 per model, and 150 in total. Oracle state directly supplies the current requirements; n/a denotes conditions without memory selection. These results use the single-task diagnostic set.

Condition	Selection	DeepSeek V4 Pro	GPT-5.6 Sol	Opus 4.8
Workspace-only	n/a	0.90 (80)	0.90 (80)	0.90 (80)
Full history	n/a	0.90 (80)	0.90 (80)	0.90 (80)
Oracle state	n/a	1.00 (100)	1.00 (100)	0.90 (80)
Flat retrieval	Common	0.90 (80)	0.90 (80)	0.90 (80)
Mem0	Native	0.90 (80)	1.00 (100)	0.90 (80)
Mem0	Common	0.90 (80)	1.00 (100)	0.90 (80)
A-MEM	Native	0.90 (80)	0.90 (80)	0.90 (80)
A-MEM	Common	1.00 (100)	0.90 (80)	0.90 (80)
MemOS	Native	1.00 (100)	1.00 (100)	0.90 (80)
MemOS	Common	1.00 (100)	1.00 (100)	0.90 (80)

F Calibration Diagnostics

F.1 Memory coverage and storage quality

These measurements reconstruct the archived five-task GPT calibration summaries, separately from the continuous trajectories. A state is a piece of task information, and an object is a memory record. Table 22 follows required-state occurrences across 54 memory-reliant decisions, counting a state again whenever another decision requires it. Probe counts exposed occurrences tested by replacement; Effect counts tested occurrences with a detected action or score change. Neither count is restricted to initially incorrect actions.

Table 22: Conditional memory-lifecycle measurements. Write and F1 use all 90 calibration decisions. Other columns give successive numerator/denominator counts within 54 memory-reliant decisions. Of 122 required-state occurrences, 120 are stored for Flat, Mem0, and A-MEM; 107 for MemOS. Counts are uniquely reconstructed from full-precision aggregates. Exact/inferred denotes the reported provenance mode.

System	Provenance	Write	F1	Retr. recall	Expose	Probe	Effect
Flat retrieval	exact	1.000	.210	40/120	13/40	9/13	1/9
Mem0	exact	.900	.657	117/120	75/117	51/75	23/51
A-MEM	exact	1.000	.210	40/120	13/40	9/13	2/9
MemOS	inferred	.920	.339	57/107	34/57	15/34	4/15

Table 23 reports the following storage measures. Current states are requirements that still apply; retired states no longer apply.

- Write coverage measures eligible information captured by new records; selectivity measures the share of new records containing eligible information.
- Precision measures the share of stored records containing current information; recall measures the share of current information represented in storage.
- Stale retention measures the share of stored records containing retired information. A record may contain both current and retired information.
- Update response counts retired states whose old representation is removed or whose replacement is stored. Physical retirement counts only removal of the old representation.

Let $\mathcal{O}_{\text{live}}$ denote stored records and $\mathcal{O}_{\text{cur}}$ those containing current information. Let $\mathcal{S}_{\text{cur}}$ denote current states and $\mathcal{S}_{\text{rep}} \subseteq \mathcal{S}_{\text{cur}}$ those represented in storage.

$$P = \frac{|\mathcal{O}_{\text{cur}}|}{|\mathcal{O}_{\text{live}}|}, \qquad R = \frac{|\mathcal{S}_{\text{rep}}|}{|\mathcal{S}_{\text{cur}}|}, \qquad F_1 = \frac{2PR}{P+R}. \tag{6}$$

Retrieval and exposure coverage use successive conditional denominators over required-state occurrences:

$$\text{Ret} = \frac{\#\{\text{stored and retrieved}\}}{\#\{\text{stored}\}}, \qquad \text{Exp} = \frac{\#\{\text{retrieved and exposed}\}}{\#\{\text{retrieved}\}}. \tag{7}$$

Table 23: GPT calibration storage quality and state updates. Ratios pool the checkpoint observations used by the storage scorecard for the 90 fixed decisions. Provenance follows Table 22: exact for Flat retrieval, Mem0, and A-MEM; inventory-based inference for MemOS. Each column is a ratio between 0 and 1.

System	Selectivity	Precision	Recall	Stale	Update	Retired
Flat retrieval	0.099	0.118	0.949	0.024	0.902	0
Mem0	0.377	0.532	0.859	0.157	0.902	0
A-MEM	0.099	0.118	0.949	0.024	0.902	0
MemOS	0.146	0.213	0.834	0.072	0.725	0

F.2 Record selection and added context

Native selection uses the backend's returned ranking; common selection applies the shared reranker. Table 24 reports the retained scorecards for both settings on the same fixed decisions. The selection settings jointly determine record identity and ordering.

Table 24: GPT memory-selection ablation: 90 fixed decisions per setting on the same five tasks. N and C denote native and common selection. Score permits partial credit; Exact is action accuracy (%); Δ is C minus N exact accuracy in percentage points.

System	N score	N exact (%)	C score	C exact (%)	Δ
Mem0	0.8750	83.33	0.8222	74.44	-8.89
A-MEM	0.6472	48.89	0.6083	42.22	-6.67
MemOS	0.6583	50.00	0.7472	63.33	+13.33

The added-record test appends unrelated administrative notes while keeping the original state evidence, workspace, checkpoint, and policy fixed. Five tasks provide four decision types, covering early adoption, stale selection, local permission, and local–global conflict. Each addition level has 20 contrasts per system/selection setting, with 420 contrasts overall. Action and checker-outcome change rates coincide in these records. The intervention tests context composition by adding records and increasing input length together.

Table 25: GPT sensitivity to additional visible memory. Cells give action-change and behavior-change rates (%), which coincide, over 20 paired contrasts each. Column labels give the number of appended records. Native uses the backend ranking; Common applies the shared reranker. Repeated conditions share the same five tasks.

System	Selection	+1	+5	+20
Flat retrieval	Common	0	0	0
Mem0	Native	0	0	0
Mem0	Common	10	10	15
A-MEM	Native	25	20	20
A-MEM	Common	15	20	30
MemOS	Native	20	15	15
MemOS	Common	10	10	10

F.3 Replacement controls

Exact provenance matches required information through backend write/update records; inferred provenance uses changes in stored contents. Replacement probes substitute an attributed record with unrelated administrative text and test for a repeat-stable action or score change. Other records may supply the same information after replacement.

The released procedure uses two neutral filler templates. It preserves record count, position, rank, score, and timestamps, and matches the replaced content's character length by repeating and truncating the filler. Length matching uses characters. The primary contrast replaces the original record with the first filler; the sham contrast compares the two fillers. Nonzero sham changes indicate sensitivity to filler wording. Table 26 reports the archived scorecard results.

Effect in Table 22 is the fraction of probed required-state occurrences with a repeat-stable action or score change. Primary and sham flip rates in Table 26 instead count action changes over intervention contrasts. Their denominators differ, so these rates cannot be compared by subtraction.

Table 26: GPT calibration replacement sensitivity under common selection (%), as reported in the retained scorecard. Primary is record-replacement sensitivity; Sham is the action-flip rate of the negative control. These are distinct from the conditional state-occurrence effect counts in Table 22.

System	Primary (%)	Sham (%)
Flat retrieval	6.67	6.67
Mem0	32.76	3.45
A-MEM	13.33	13.33
MemOS	15.38	10.26