RETENTION RIGHTS DETERMINE WHAT BUDGET-MATCHED AGENT-MEMORY COMPARISONS ESTABLISH

Anonymous authors
Paper under double-blind review

ABSTRACT

Valid controller attribution requires equal retention rights, not only equal memory and compute budgets. A controller that can pin rare constraints or replay-validate promotions acts over a larger feasible action set than one limited to eviction and ranking. We introduce capability alignment, a paired intervention that imposes a common retention-right profile before attribution, and full-versus-aligned estimands that quantify the resulting attribution shift. RetainScope, a validated consolidation controller for mutable tool traces, makes the protocol executable. In five-seed, identity-paired audits using Llama-3.1-70B-Instruct and a 4.0 GB cap, the RetainScope–MemRouter gap on LongToolBench v2.1 contracts from $+1.3$ to $+0.2$ pp (51.2 vs. 51.0), an 84.6% shrinkage with direct 95% CI [0.3, 1.9] and $p = 0.012$. On LOCOMO v2, $+2.8$ pp contracts to $+0.3$ pp (47.6 ± 1.9 vs. 47.3 ± 1.8), an 89.3% shrinkage with direct 95% CI [1.2, 3.8] and $p = 0.006$. Both aligned residual intervals overlap zero, while paging/self-edit and frequency-based comparators yield the same attribution decision and gaps contract in 64 of 72 profile-grid pairs. RetainScope also supplies a strong systems operating point: 51.9% LongToolBench success, 52% fewer regressions than Store-all, 2.6 GB median end-to-end memory, and 1.63 s p95 retrieval latency. These results establish retention-right parity as the comparison control for selector attribution and full-system rows as complementary evidence for deployment choice.

1 INTRODUCTION

Persistent memory lets an agent carry constraints, tool recipes, and verified outcomes across sessions. That continuity can raise completion rates and reduce repeated exploration, yet it also makes evaluation unusually sensitive to what each controller is permitted to retain. Consider two rows with the same 4.0 GB cap: both may rank memories, but one may additionally pin a rare safety constraint and promote a schema after tool replay. Their bytes match; their feasible decisions do not. A performance difference between those rows combines selection quality with access to different retention actions.

We make the admissible action set and the retention-right profile that generates it explicit objects of agent-memory evaluation. For method j, $\mathcal{A}_j(m, h_t)$ contains the legal actions for item m given history h_t: retaining, compressing, pruning, pinning, promoting, and validating with a specified signal family. *Budget matching* equalizes resource constraints after actions are chosen. *Capability alignment* additionally imposes a common profile for the rights that determine whether an item can enter the protected or promoted set before utility ranking. This distinction separates three useful claims: an *operating-point* comparison ranks complete systems, a *selector-specific* comparison attributes a residual under capability alignment, and *operating-point corroboration* tests whether a complete configuration transports to another workload.

The resulting audit estimates an attribution shift, not merely sensitivity to another hyperparameter. We measure the full-stack gap g_{full}, the capability-aligned gap g_{align}, and their difference $\Delta_{\text{attr}} = g_{\text{full}} - g_{\text{align}}$. A large positive Δ_{attr} means the full comparison assigned substantial credit to unequal retention rights. Because full and aligned cells share task identities, sessions, and seeds, paired bootstrap and hierarchical permutation tests operate directly on the gap shrinkage. Ordinary ablations leave one method's feasible actions fixed; capability alignment changes the intervention so that selector attribution is defined over the same action set.

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.

RetainScope makes this audit executable while supplying a competitive fixed-budget design. Each sleep cycle scores future utility, validates typed promotions, protects verifier-approved pins, and prunes under footprint and compute constraints. The online loop, prompts, Llama-3.1-70B-Instruct backbone, BGE retrieval stack, task instances, 4.0 GB soft cap, and 25 GPU + 10 CPU minute daily ceiling remain fixed. MemRouter is the strongest learned shared-harness selector, LFU+pin tests a strong frequency heuristic, and a capability-aligned MemGPT reimplementation contributes an architecturally distinct paging and self-edit comparator. Matching pin budget, validator family, and promotion rights leaves selection logic as the principal difference among aligned rows.

The evidence changes the causal reading on two long-horizon benchmarks. LongToolBench's full $+1.3$ pp RetainScope advantage contracts to $+0.2$ pp after alignment, yielding an 84.6% attribution shrinkage. LOCOMO repeats the pattern: $+2.8$ pp becomes $+0.3$ pp, with a direct shrinkage interval of [1.2, 3.8]. Both aligned RetainScope–MemRouter residual intervals overlap zero while every direct shrinkage interval in the primary table excludes zero; full-to-aligned comparisons with MemGPT-reimpl and LFU+pin reach the same attribution decision. Across a rights grid spanning four pin budgets, three validator settings, and matched or unmatched promotion access, gaps contract in 64 of 72 profile pairs. Complementarily, the complete RetainScope operating point reaches 51.9% LongToolBench success while reducing regressions, runtime memory, and latency tails.

The paper contributes:

- a formal intervention over method-specific admissible retention sets, together with full-gap, common-profile aligned-gap, attribution-shrinkage, and collapse-ratio estimands for deciding what a controller comparison establishes;
- identity-paired capability audits on LongToolBench v2.1 and LOCOMO v2, including direct uncertainty on gap shrinkage and aligned rows from learned routing, paging/self-edit, and frequency-based comparator families; and
- a systems evidence package for RetainScope covering accuracy–memory–latency trade-offs, rights-grid robustness, calibrated utility, ablations, cross-model transfer, and security-oriented failure accounting.

Together, these results turn retention rights from an implicit implementation detail into an inspectable experimental control and preserve the practical value of full-stack rows while making selector attribution precise.

2 Capability-Aligned Retention Audits

A tool-augmented agent carries a typed memory store $\mathcal{M}_t$ across sleep cycle t. Each item m records content, provenance, timestamp, verifier state, type (`Epi`, `Sem`, or `Proc`), size, recency, and contradiction and retrieval statistics. Method j selects an action $\pi_j(m)$ from the history-dependent admissible set

$$\mathcal{A}_j(m, h_t) \subseteq \{\text{keep}, \text{compress}, \text{prune}, \text{pin}, \text{promote}, \text{validate}_v : v \in \mathcal{V}\}.$$

The set is method-specific: pin access changes whether an item may become unprunable, validator access changes whether a promotion is legal, and promotion rights change which memory tier can receive the item. Let $\rho_j = (b_j^{\text{pin}}, \mathcal{V}_j, \mathcal{T}_j)$ denote method j's retention-right profile: its pin allowance, validator signal family, and admissible promotion destinations. For any imposed profile ρ, $Y_j(\rho)$ denotes the endpoint produced when controller j retains its selector and ranking logic while operating with the rights in ρ. This intervention exposes capability changes that occur before candidate utility determines the retained set.

Budget parity and capability alignment. Methods j and k are budget matched when they share footprint budget B_f and sleep-compute budget B_s. They are capability aligned at a common profile $\rho^\star$ when the intervention gives both controllers the same pin allowance, validator family, and promotion destinations, making the action-set projection onto these rights identical for every paired item and history. The audits align $\rho^\star$ while holding the online loop, backbone, retrieval stack, prompts, task identities, and budgets fixed. A parity certificate records the common profile, exogenous stack, resource ceilings, pairing unit, and seeds (Appendix Table 9). Selector-specific attribution is valid

when both budget parity and capability alignment hold; full-stack comparisons remain the appropriate object for choosing a deployed configuration.

Gap-attribution estimand. For endpoint score Y, define the native-profile and common-profile contrasts

$$\begin{aligned} g_{\text{full}} &= \mathbb{E}[Y_j(\rho_j) - Y_k(\rho_k)], \\ g_{\text{align}}(\rho^\star) &= \mathbb{E}[Y_j(\rho^\star) - Y_k(\rho^\star)], \\ \Delta_{\text{attr}}(\rho^\star) &= g_{\text{full}} - g_{\text{align}}(\rho^\star), \qquad C_{\text{collapse}} = \Delta_{\text{attr}}(\rho^\star)/g_{\text{full}}. \end{aligned}$$

$\Delta_{\text{attr}}(\rho^\star)$ is a paired difference-in-differences over identical seed–task endpoints; C_{collapse} reports the fraction of the full gap assigned to unequal retention rights at the audited common profile. A selector comparison is interpretable when action-set parity holds on the audited axes and the direct interval for Δ_{attr} is estimated from paired cells.

3 RetainScope: Validated Fixed-Budget Consolidation

Constrained controller policy. The audit requires a controller that exposes pin, validator, and promotion rights as explicit actions. RetainScope ranks candidates by predicted future utility while enforcing footprint, compute, pin, and action-admissibility constraints. Let $c(m)$ be footprint, $s(\pi(m))$ offline cost, and $\mathcal{P}_t$ the pinned set. The feasible retained sets are

$$\begin{aligned} \mathcal{F}_t = \Big\{ \mathcal{S} \subseteq \mathcal{M}_t : \quad & \sum_{m \in \mathcal{S}} c(m) \le B_f, \quad \sum_{m \in \mathcal{M}_t} s(\pi(m)) \le B_s, \\ & \mathcal{P}_t \subseteq \mathcal{S}, \quad \pi(m) \in \mathcal{A}_{\text{RetainScope}}(m, h_t) \text{ for each } m \in \mathcal{M}_t \Big\}. \end{aligned} \tag{1}$$

3.1 Overview and Minimal Core

RetainScope integrates score, validate, pin, and prune into one control policy. The online agent writes episodic traces; the sleep stage scores all retained items with calibrated $\hat{U}$, preloads approved pins, selects under footprint, compute, and diversity constraints, validates typed promotions for consistency, staleness, and tool behavior, and prunes the lowest-utility non-pinned items. Distillation and rehearsal run after validation as independently rollback-safe extensions. This ordering makes retention rights operational: pinning determines protected-set membership, and validator access determines promotion eligibility before the ranking policy allocates capacity.

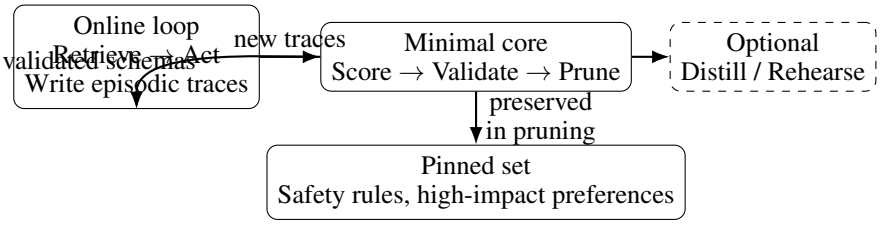

Figure 1: Core operations with optional extensions. Scored selection, validated promotion, and pinned pruning form the three-operator core shown by the solid path; optional distillation and rehearsal appear in the dashed box and are reported separately.

We use three memory types. Raw traces and tool logs constitute episodic items. Semantic items take the form of structured schemas, including constraints or tool recipes, with pointers to provenance. Distilling validated schemas yields reusable high-level templates, which constitute procedural items. The three-operator core reaches 51.4% session-20 success without parameter updates, within 0.5 pp of the complete controller (Table 8); optional adaptation improves the operating point while remaining separable from the retention-right audit.

3.2 Utility Model and Selection

The score operator estimates future usefulness under the logged policy class:

$$U(m) \approx \Pr(r(m) = 1 \mid m) \cdot \mathbb{E}[\Delta(m) \mid r(m) = 1, m],$$

where $r(m)$ records whether reuse occurs within the horizon, while $\Delta(m)$ denotes the marginal increase in success attributable to retaining m for retrieval. Independent LightGBM predictors model reuse and conditional benefit; isotonic regression calibrates their product on held-out replay labels. The features describe recency and access frequency, item size and redundancy, verifier status, indicators of contradiction / staleness, and provenance derived from the producing tool or verifier path. Replayable traces estimate marginal benefit by rerunning frozen prompts and tool I/O with an item available versus withheld. Otherwise, the benefit term is missing and the score uses reuse probability scaled by empirical benefit among comparable memory types. Held-out sessions from LongToolBench, SWE-bench Lite, and LOCOMO retain good utility-signal ranking and calibration (Appendix Table 12).

Selection clusters items by embedding similarity, applies per-cluster quotas, and reserves 10% of capacity for rare high-impact memories. It begins with the pinned set and accepts candidates in decreasing $\hat{U}(m)$ order only while the updated set remains in $\mathcal{F}_t$; it does not claim to solve a global utility maximization exactly. Appendix A.12 gives the resulting feasibility invariant.

3.3 Validation, Pinning, and Optional Adaptation

Selection is followed by compression of episodic traces into schemas that carry explicit provenance. Schemas have types whose use depends on the benchmark: every benchmark family includes `Constraint`, traces with executable tool interactions use `ToolRecipe`, LOCOMO is dominated by relation-style semantic schemas, and only SWE-style traces include `BugFixTemplate`. Candidate filtering then uses lightweight checks for consistency, detection of contradictions, checks for staleness relative to newer evidence, and replay of tool-level verifiers where benchmark logs permit deterministic replay.

Pinned constraints protect rare, high-impact rules. Pins are never pruned; promotion to the pinned set follows the same validator pathway as ordinary semantic promotion. Newer, higher-confidence contradictory evidence demotes a pin to ordinary memory, where the current evidence can replace it under the normal budget policy.

Two optional modules can also run after validation in the full recipe. Distillation may convert validated schemas into small LoRA adapters or procedural templates; short rehearsal examples may be constructed from failure clusters. Canary gating and safe rollback apply to both modules: a regression on a canary causes rejection of the adapter update, leaving the main consolidated store unchanged. The ablations in Sec. 5.4 find smaller gains from these modules than from utility-guided selection and validation.

4 Experimental Setup

Research questions. RQ1 asks how controller attribution changes after capability alignment. RQ2 measures the accuracy–stability–resource operating point. RQ3 tests attribution across benchmarks, comparator families, and a grid of retention rights. RQ4 isolates operating-point mechanisms, resource tradeoffs, and retention enforcement.

Benchmarks and splits. LongToolBench v2.1 contains 200 tool-use identities observed over 20 sessions with shifted arguments, hidden state, tool responses, and constraints, yielding 1,000 seed–identity endpoints at session 20. Memories remain within identity, and scorer training, isotonic calibration, and headline evaluation use disjoint identity splits. LOCOMO v2 (Maharana et al., 2024) supplies a second 20-session aligned audit with 120 held-out query trajectories and 600 seed–identity endpoints. Both audits use the same five seeds and identity pairing. Public τ-bench v0.4 (Yao et al., 2025) supplies an operating-point check: release-ordered Retail (200 tasks) and Airline (150 tasks) are partitioned into 20 contiguous batches, with memory persisting within domain and seed. SWE-bench Lite (Jimenez et al., 2024) tests procedural reuse in the appendix.

Systems and comparators. All aligned rows share Llama-3.1-70B-Instruct (temperature 0.2, top-p 0.9), bge-large-en-v1.5 embeddings, bge-reranker-v2-m3, FAISS HNSW ($M = 32$, $ef = 128$), prompts, tools, tasks, redaction, a 4.0 GB end-to-end soft cap, and a 25 GPU + 10 CPU minute daily consolidation ceiling. MemRouter is the strongest learned selector evaluated in this shared harness; LFU+pin tests frequency-based selection; and MemGPT-reimpl preserves hierarchical paging and self-edit logic while adopting the common backbone and retrieval stack. The capability contract changes the rights interface only: MemRouter's learned routing, MemGPT's paging/self-edit policy, and LFU's frequency rule remain method-native while pin budget, NLI validation, promotion access, cap, tasks, and seeds are aligned. Store-all, TTL eviction, periodic summarization, and offline consolidation contextualize RQ2. Appendix Table 9 records exact parity.

Metrics and inference. The primary endpoint is session-20 success. A regression is a session-level success drop of at least 1.0 pp on the same task set; we also report end-to-end runtime memory and per-request latency. Success is mean±standard deviation across seeds; heavy-tailed regression and memory metrics are median [Q25, Q75]. Endpoint contrasts use paired seed–task bootstrap intervals. The direct shrinkage test resamples identities within seed and applies a hierarchical sign-flip permutation to Δ_{attr}; its null is equal full and aligned gaps. We report five seed-level effects and all 1,000 LongToolBench and 600 LOCOMO endpoint tuples in Appendix Table 4. Latency percentiles pool 80,000 requests per method across seeds. Experiments run on 4×A100-80GB SXM GPUs with CUDA 12.4, driver 550.90.07, PyTorch 2.3.1, and vLLM 0.5.2, using seeds $\{13, 37, 61, 71, 97\}$.

5 Results

RQ1 and Table 1 carry selector attribution; RQ2 selects among complete operating points; RQ3 tests the intervention band; and RQ4 explains operating-point mechanisms and enforcement.

5.1 RQ1: How Much of the Full Gap Survives Capability Alignment?

Capability alignment assigns most of the full comparator gap to unequal retention rights (Table 1). Against MemRouter, LongToolBench changes from $51.9 - 50.6 = +1.3$ pp in the full rows to $51.2 - 51.0 = +0.2$ pp in aligned rows, hence $\Delta_{\text{attr}} = +1.1$ pp and $C_{\text{collapse}} = 84.6\%$. The direct shrinkage interval, [0.3, 1.9], excludes zero ($p = 0.012$), whereas the aligned residual interval [-0.7, 1.1] includes zero ($p = 0.642$). MemGPT-reimpl contracts from a $+1.8$ pp full gap to $+0.4$ pp after alignment; LFU+pin contracts from $+1.7$ to $+0.5$ pp. Their direct shrinkage intervals also exclude zero.

LOCOMO reproduces the attribution shift across all three comparators. Its RetainScope–MemRouter gap contracts from $48.7 - 45.9 = +2.8$ pp to $+0.3$ pp (47.6 vs. 47.3). The resulting $+2.5$ pp shrinkage has 95% CI [1.2, 3.8] and $p = 0.006$; the aligned residual interval [-0.8, 1.4] includes zero ($p = 0.581$). The paging/self-edit comparator contracts from $+3.6$ to $+0.5$ pp, and the frequency comparator from $+3.1$ to $+0.7$ pp. The shrinkage pattern therefore replicates across routing, paging/self-edit, and frequency selectors under the shared capability contract. By construction, every aligned row receives the same 128-pin, NLI-only, promotion-enabled profile; RetainScope's native replay access contributes to g_{full} and is excluded from g_{align}.

5.2 RQ2: Which Full Controller Offers the Best Operating Point?

RetainScope leads the seven LongToolBench configurations in session-20 success while using the least runtime memory and producing the fewest regressions (Table 2). Relative to Store-all, it gains $+3.8$ pp, reduces median end-to-end memory $4.92\times$, and cuts regression events by 52%. Relative to MemRouter, it gains $+1.3$ pp (95% CI $[0.3, 2.3]$, $p = 0.018$) while lowering memory from 3.5 to 2.6 GB. The 2.6 GB value includes resident controller state and temporary replay workspaces; Figure 5's 2006 MB value measures the persisted store after sleep.

Session curves in Figure 3 show that RetainScope's advantage emerges after memory has accumulated and persists through session 20 on both long-horizon workloads. The complete controller also transports beyond the aligned audits: on the public τ-bench adaptation it exceeds MemRouter by $+2.8$ pp on Retail and $+2.6$ pp on Airline while using less memory and lower p95/p99 latency (Appendix Table 11). Qwen2.5-72B preserves 84% of the Llama-based full-controller advantage on

Table 1: Primary selector-attribution evidence for RQ1 over 1,000 LongToolBench and 600 LOCOMO seed–identity endpoints. Full rows preserve method-native profiles; aligned rows share a 4.0 GB cap, 128 pins, NLI-only validation, promotion destinations, tasks, and seeds. Identical pin and validator labels across regimes do not imply profile parity because full rows retain native promotion interfaces. The lower panel directly tests gap shrinkage.

Bench.	Regime	Method	Pins	Validator	s20 (%)	RS gap (pp)
LongToolBench v2.1						
LT	Full	*RetainScope*	128	Replay+NLI	51.9 ± 1.7	—
LT	Full	MemRouter	128	NLI-only	50.6 ± 1.7	+1.3
LT	Full	MemGPT-reimpl	0	Self-edit	50.1 ± 1.9	+1.8
LT	Full	LFU+pin	128	None	50.2 ± 1.7	+1.7
LT	Aligned	*RetainScope*	128	NLI-only	51.2 ± 1.8	—
LT	Aligned	MemRouter	128	NLI-only	51.0 ± 1.8	+0.2
LT	Aligned	MemGPT-reimpl	128	NLI-only	50.8 ± 1.9	+0.4
LT	Aligned	LFU+pin	128	NLI-only	50.7 ± 1.8	+0.5
LOCOMO v2						
LO	Full	*RetainScope*	128	Replay+NLI	48.7 ± 1.8	—
LO	Full	MemRouter	128	NLI-only	45.9 ± 1.7	+2.8
LO	Full	MemGPT-reimpl	0	Self-edit	45.1 ± 1.9	+3.6
LO	Full	LFU+pin	128	None	45.6 ± 1.6	+3.1
LO	Aligned	*RetainScope*	128	NLI-only	47.6 ± 1.9	—
LO	Aligned	MemRouter	128	NLI-only	47.3 ± 1.8	+0.3
LO	Aligned	MemGPT-reimpl	128	NLI-only	47.1 ± 1.9	+0.5
LO	Aligned	LFU+pin	128	NLI-only	46.9 ± 1.9	+0.7

Bench.	Comparator	g_{full}	g_{align}	Δ_{attr}	Collapse	Shrinkage 95% CI; p
LT	MemRouter	+1.3	+0.2	+1.1	84.6%	[0.3, 1.9]; 0.012
LT	MemGPT-reimpl	+1.8	+0.4	+1.4	77.8%	[0.5, 2.3]; 0.008
LT	LFU+pin	+1.7	+0.5	+1.2	70.6%	[0.2, 2.2]; 0.021
LO	MemRouter	+2.8	+0.3	+2.5	89.3%	[1.2, 3.8]; 0.006
LO	MemGPT-reimpl	+3.6	+0.5	+3.1	86.1%	[1.6, 4.6]; 0.003
LO	LFU+pin	+3.1	+0.7	+2.4	77.4%	[1.0, 3.8]; 0.009

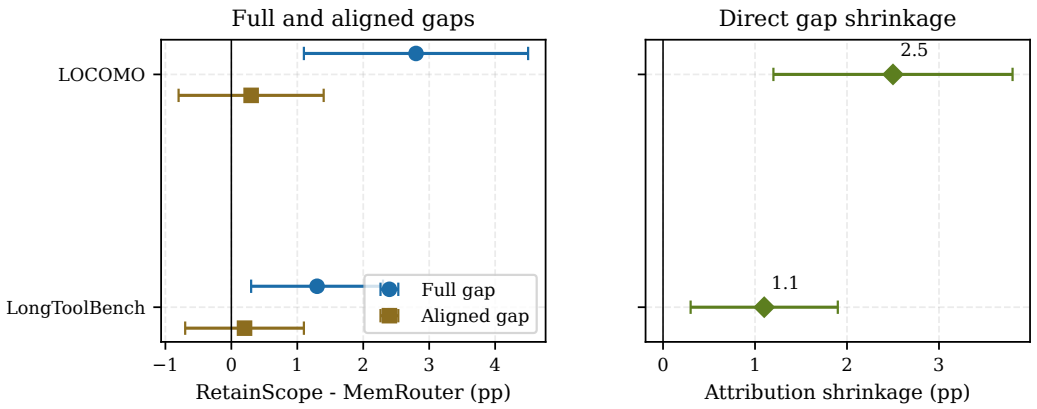

Figure 2: Capability alignment contracts the RetainScope–MemRouter success gap on both benchmarks. Points and horizontal bars show paired estimates and 95% intervals; the right panel reports the directly estimated shrinkage.

Table 2: LongToolBench v2.1 full-controller operating points at session 20 (five seeds). Managed methods share the 4.0 GB soft cap and compute ceiling; Store-all is the uncapped reference. Runtime memory covers the resident process; Figure 5 separately reports the 2006 MB persisted store at session 20.

Family	Method	Success (%)	Regressions / 100 ep.	Runtime memory (GB)
Reference	Store-all	48.1 ± 1.9	15.3 [11.4, 19.7]	12.8 [11.2, 15.1]
Heuristic	TTL eviction	46.2 ± 2.3	17.8 [13.1, 23.2]	5.7 [4.8, 6.9]
Summary	Periodic summarization	49.3 ± 1.6	12.1 [8.8, 16.3]	4.6 [3.9, 5.5]
Consolidation	Offline consolidation	49.7 ± 1.7	11.4 [8.3, 15.0]	3.9 [3.2, 4.7]
Heuristic	LFU+pin	50.2 ± 1.7	9.6 [7.1, 13.2]	3.2 [2.7, 4.0]
Learned	MemRouter	50.6 ± 1.7	9.1 [6.8, 12.4]	3.5 [2.9, 4.3]
Validated	*RetainScope*	51.9 ± 1.7	7.4 [5.1, 10.2]	2.6 [2.1, 3.2]

average, and cross-benchmark utility transfer preserves 71–87% after zero-shot transfer or 50-episode recalibration (Appendix Tables 10 and 20).

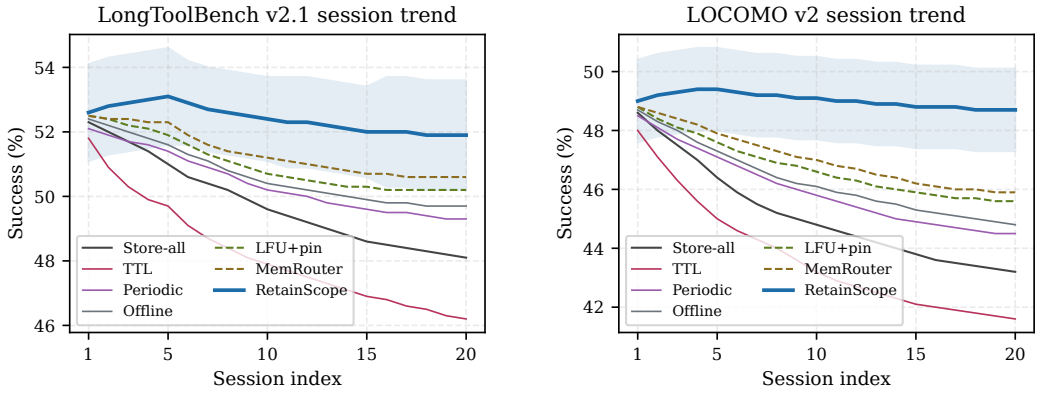

Figure 3: Full-controller session-wise success on LongToolBench v2.1 (left) and LOCOMO v2 (right). Bands show 95% bootstrap intervals for RetainScope over five seeds.

5.3 RQ3: Is Attribution Stable Across Retention Rights and Workloads?

Capability alignment remains decisive over a $4 \times 3 \times 2$ grid of pin budgets, validator access, and promotion parity for each benchmark and audit comparator. Across 72 matched-versus-unmatched setting pairs, it reduces the success gap in 64, with six local reversals and two ties preserving a non-monotone surface; median shrinkage is 66.7%, and RetainScope's regression-event advantage ranges from 0.4 to 3.1 per 100 episodes. Table 6 reports every success-gap cell, Table 7 adds direct uncertainty and raw action counts, and Figure 4 visualizes the surface. The attribution decision at 128 pins with NLI therefore extends across the surrounding capability band.

5.4 RQ4: Which Mechanisms Deliver Stability and Security?

RQ4 explains the RQ2 operating point through mechanism and enforcement evidence. Utility selection and validation are the largest contributors: removing them costs 2.5 and 1.8 pp respectively and increases regressions by 3.4–3.9 events per 100 episodes (Table 3). Pins, replay, and diversity quotas contribute complementary stability, while the core without distillation and rehearsal remains within 0.5 pp. Under 10% adversarial writes, RetainScope quarantines 81.7% (490/600) at initial validation; 39 additional writes are caught by replay and 71 are promoted. Appendix Table 16 links each protected object and adversary to its trust boundary, failure rate, utility impact, and overhead.

Table 3: Component ablations and security stress tests on LongToolBench. Regressions are median [Q25, Q75] per 100 episodes. The adversarial-write denominator covers all attempted malicious writes across five seeds.

A. Component ablations (LongToolBench s20)

Variant	s20 (%)	Regr. / 100 ep.
RetainScope	51.9 ± 1.7	7.4 [5.1, 10.2]
w/o utility selection	49.4 ± 2.1	10.8 [7.9, 14.6]
w/o validation	50.1 ± 1.9	11.3 [8.4, 15.1]
w/o pins	50.3 ± 1.9	9.8 [7.2, 13.4]
w/o verifier replay	50.0 ± 1.8	10.4 [7.6, 14.0]
w/o diversity quotas	50.6 ± 1.8	9.1 [6.7, 12.3]
w/o distillation	51.6 ± 1.6	8.1 [5.8, 11.0]
w/o rehearsal	51.5 ± 1.7	8.0 [5.7, 10.9]
w/o distill + rehearsal	51.4 ± 1.7	8.6 [6.2, 11.7]

B. Safety stress suite (LongToolBench s20)

Condition	s20 (%)	Incident rate
Clean	51.9 ± 1.7	—
10% contradiction injection	50.6 ± 2.1	2.7% contradiction miss
Major tool drift	49.7 ± 2.3	7.1% stale recipe
Paraphrase / derived secret	51.6 ± 1.8	2 / 412 leaks (0.49%)
Adversarial writes, $\rho = 10\%$	49.3 ± 2.3	81.7% (490/600) quarantined

6 Related Work

Persistent-memory systems stage, summarize, or restructure experience through MemoryBank, SCM, LightMem, and MemInsight (Zhong et al., 2023; Wang et al., 2023; Fang et al., 2026; Salama et al., 2025); MemGPT and HiAgent organize hierarchical or working memory (Packer et al., 2023; Hu et al., 2025). Retrieval-augmented and long-context models instead expand accessible evidence or cached state (Lewis et al., 2020; Izacard et al., 2023; Khandelwal et al., 2020; Borgeaud et al., 2022; Liu et al., 2024), while replay-based continual learning manages parameter or buffer updates (Kirkpatrick et al., 2017; Rolnick et al., 2019; Lopez-Paz et al., 2017; Chaudhry et al., 2019). Agent benchmarks expose long dialogue, tool interaction, and recurring software tasks (Qin et al., 2023; Wang et al., 2024; Maharana et al., 2024; Yao et al., 2025; Jimenez et al., 2024; Tan et al., 2025).

These lines of work motivate three intervention levels. Native-stack evaluations answer whole-system choice while preserving each design; shared-harness evaluations compare policies after normalizing backbone, retrieval, prompts, and budgets; capability-aligned evaluations additionally equalize rights that change the feasible retention set before ranking. A hyperparameter or fairness ablation measures sensitivity within one policy's action set. Capability alignment instead imposes the same profile $\rho^\star$ on both controllers and estimates the paired change from the native-profile gap to $g_{\mathrm{align}}(\rho^\star)$. RetainScope's full/aligned design makes this attribution shift measurable across routing, paging/self-edit, and frequency-based controllers.

7 Discussion and Implications

Two decisions from one evaluation. Full-controller rows answer deployment selection: RetainScope is the strongest measured accuracy–memory–stability point, and 4.0 GB lies near the Pareto knee. Doubling the cap to 8.0 GB adds 0.2 pp success but increases p95 latency from 1.63 to 1.94 s and p99 from 3.30 to 3.94 s (Appendix Table 14). Capability-aligned rows answer scientific attribution: once pin, validator, and promotion rights match, the remaining selector differences are small and uncertain, while the shrinkage itself is decisive on both benchmarks. Reporting both rows preserves the system-engineering result and assigns its causes correctly.

Experimental practice. Benchmark authors can attach a retention-right profile and parity certificate to each claim-bearing row alongside memory and compute budgets. Reviewers can request an aligned row whenever methods differ in pin eligibility, validation signals, promotion tiers, or replay access. Controller builders can expose these rights directly, making it possible to compare ranking logic over a common feasible set. The rights grid further shows why a single scalar "memory budget" is insufficient: action parity explains most of the attribution shift across a broad, mildly non-monotone configuration surface.

Security boundary. For mutable tool traces, retention control is also an enforcement boundary. Pins protect rare constraints, validators limit stale or contradictory promotions, replay catches behaviorally invalid recipes, and redaction/canaries measure leakage. The stress tests retain non-zero failure rates, including 2.7% contradiction misses, 7.1% stale recipes, and 2/412 paraphrased leaks. These measured failures localize where stronger validators matter while the benign operating point remains competitive. Across two aligned long-horizon workloads, two learned comparator families, one heuristic family, and a fixed shared stack, the same capability profile connects attribution evidence to measured enforcement behavior.

8 Conclusion

Retention rights determine what budget-matched agent-memory comparisons establish. Capability alignment turns pinning, validation, and promotion access into an explicit common-profile intervention and quantifies their contribution through a direct gap-attribution estimand. Across LongToolBench and LOCOMO, alignment reassigns 84.6% and 89.3% of the full RetainScope–MemRouter gaps, with the same attribution decision under paging/self-edit and frequency comparators and across the rights grid. RetainScope simultaneously delivers a strong practical operating point in success, stability, memory, and latency. Together, the intervention, estimand, and empirical package establish retention-right parity as a design principle for persistent-memory evaluation: complete systems support deployment selection, and capability-aligned controllers support causal attribution.

AI Use Statement

Generative AI tools assisted with manuscript revision and presentation, including restructuring and editing the title, abstract, introduction, formal exposition, results narrative, related-work organization, appendices, and figure captions. The authors reviewed all AI-assisted revisions and take responsibility for the final content, including its text, claims, and reported results.

Ethics Statement

Persistent memory can preserve sensitive text, stale instructions, or adversarial writes. RetainScope applies redaction before write, validator checks before promotion, verifier replay for executable recipes, protected safety pins, canary gating, and rollback. Table 3 and Appendix Tables 16 and 18 report contradiction, staleness, poisoning, false-prune, and leakage outcomes with operational denominators.

Reproducibility Statement

Section 4 specifies workloads, identity-level splits, metrics, seeds, hardware, software versions, budgets, and inference procedures. Section 2 defines the alignment intervention and estimands; Appendix A.4 gives controller and parity details, and Appendix Tables 4, 6, and 21 provide seed-level effects, the full rights grid, and experiment-level compute.

References

Joshua Ainslie, Santiago Ontañón, Chris Alberti, et al. ETC: Encoding Long and Structured Inputs in Transformers. In *EMNLP*, 2020.

Rahaf Aljundi, Lucas Caccia, Eugene Belilovsky, et al. Online Continual Learning with Maximally Interfered Retrieval. In *NeurIPS*, 2019.

Akari Asai, Zeqi Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. In *ICLR*, 2024.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, et al. Improving Language Models by Retrieving from Trillions of Tokens. In *ICML*, 2022.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient Lifelong Learning with A-GEM. In *ICLR*, 2019.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. In *ACL*, 2019.

Jizhan Fang, Xinle Deng, Haoming Xu, Ziyan Jiang, Yuqi Tang, Ziwen Xu, Shumin Deng, Yunzhi Yao, Mengru Wang, Shuofei Qiao, Huajun Chen, and Ningyu Zhang. LightMem: Lightweight and Efficient Memory-Augmented Generation. In *ICLR*, 2026.

Mengkang Hu, Tianxing Chen, Qiguang Chen, Yao Mu, Wenqi Shao, and Ping Luo. HiAgent: Hierarchical Working Memory Management for Solving Long-Horizon Agent Tasks with Large Language Model. In *ACL*, 2025.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot Learning with Retrieval Augmented Language Models. *JMLR*, 2023.

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 *ICLR*, 2024.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through Memorization: Nearest Neighbor Language Models. In *ICLR*, 2020.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, et al. Overcoming Catastrophic Forgetting in Neural Networks. *PNAS*, 2017.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *NeurIPS*, 2020.

Nelson F. Liu, Kevin Lin, John Hewitt, et al. Lost in the Middle: How Language Models Use Long Contexts. *TACL*, 2024.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient Episodic Memory for Continual Learning. In *NeurIPS*, 2017.

Aman Madaan, Niket Tandon, Prakhar Gupta, et al. Self-Refine: Iterative Refinement with Self-Feedback. In *NeurIPS*, 2023.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating Very Long-Term Conversational Memory of LLM Agents. In *ACL*, 2024.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories. In *ACL*, 2023.

Joon Sung Park, Joseph O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative Agents: Interactive Simulacra of Human Behavior. In *UIST*, 2023.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating Large Language Models to Master 16000+ Real-world APIs. *arXiv:2307.16789*, 2023.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive Transformers for Long-Range Sequence Modelling. In *ICLR*, 2020.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. In-Context Retrieval-Augmented Language Models. *TACL*, 2023.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. iCaRL: Incremental Classifier and Representation Learning. In *CVPR*, 2017.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience Replay for Continual Learning. In *NeurIPS*, 2019.

Rana Salama, Jason Cai, Michelle Yuan, Anna Currey, Monica Sunkara, Yi Zhang, and Yassine Benajiba. MemInsight: Autonomous Memory Augmentation for LLM Agents. In *EMNLP*, 2025.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, et al. Toolformer: Language Models Can Teach Themselves to Use Tools. In *NeurIPS*, 2023.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language Agents with Verbal Reinforcement Learning. In *NeurIPS*, 2023.

Haoran Tan, Zeyu Zhang, Chen Ma, Xu Chen, Quanyu Dai, and Zhenhua Dong. MemBench: Towards More Comprehensive Evaluation on the Memory of LLM-based Agents. In *Findings of ACL*, 2025.

Bing Wang, Xinnian Liang, Jian Yang, Hui Huang, Shuangzhi Wu, Peihao Wu, Lu Lu, Zejun Ma, and Zhoujun Li. Enhancing Large Language Model with Self-Controlled Memory Framework. *arXiv:2304.13343*, 2023.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An Open-Ended Embodied Agent with Large Language Models. *arXiv:2305.16291*, 2023.

Xingyao Wang, Zihan Wang, Jiateng Liu, Yangyi Chen, Lifan Yuan, Hao Peng, and Heng Ji. MINT: Evaluating LLMs in Multi-turn Interaction with Tools and Language Feedback. In *ICLR*, 2024.

Yuhuai Wu, Markus N. Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing Transformers. *arXiv:2203.08913*, 2022.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as Operating Systems for Persistent Memory. *arXiv:2310.08560*, 2023.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering. In *NeurIPS*, 2024.

Shunyu Yao, Jeffrey Zhao, Dian Yu, et al. ReAct: Synergizing Reasoning and Acting in Language Models. In *ICLR*, 2023.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains. In *ICLR*, 2025.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, et al. Big Bird: Transformers for Longer Sequences. In *NeurIPS*, 2020.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. MemoryBank: Enhancing Large Language Models with Long-Term Memory. *arXiv:2305.10250*, 2023.

A.1 GAP-ATTRIBUTION DETAIL

The paired analysis uses the same identity and seed in every full and aligned cell. LongToolBench contributes 1,000 session-20 seed–identity tuples; LOCOMO contributes 600. Table 4 exposes the five seed-level effects underlying the direct shrinkage tests.

Table 4: Seed-level RetainScope–MemRouter gaps and attribution shrinkage (percentage points). Mean values match Table 1; direct confidence intervals resample identities within seed.

Benchmark	Full-gap effects	Aligned-gap effects	Shrinkage effects
LongToolBench	0.8, 1.6, 1.1, 1.7, 1.3	0.7, -0.2, 0.5, 0.1, -0.1	0.1, 1.8, 0.6, 1.6, 1.4
LOCOMO	2.1, 3.4, 2.6, 3.2, 2.7	0.8, -0.1, 0.4, 0.2, 0.2	1.3, 3.5, 2.2, 3.0, 2.5

Table 5: Capability-removal diagnostics under NLI-only validation. Gaps compare each no-pin RetainScope row with the corresponding aligned MemRouter row; these rows test the pin mechanism separately from the selector-attribution estimand.

Benchmark	Pins	s20 (%)	Gap vs. aligned MemRouter (pp)
LongToolBench	0	49.6 ± 2.0	-1.4
LOCOMO	0	45.8 ± 2.0	-1.5

A.2 RETENTION-RIGHTS GRID

The grid crosses pin budgets $\{0, 64, 128, 256\}$, validator access $\{\text{none}, \text{NLI}, \text{replay+NLI}\}$, and promotion rights $\{\text{unmatched}, \text{aligned}\}$. Backbone, retriever, prompts, tasks, 4.0 GB cap, and sleep-compute ceiling remain fixed. Table 6 reports the RetainScope success gap against each comparator; Figure 4 visualizes the same values. Gap compression is the dominant pattern across the grid, while local reversals and ties preserve measured seed variation rather than imposing monotonicity.

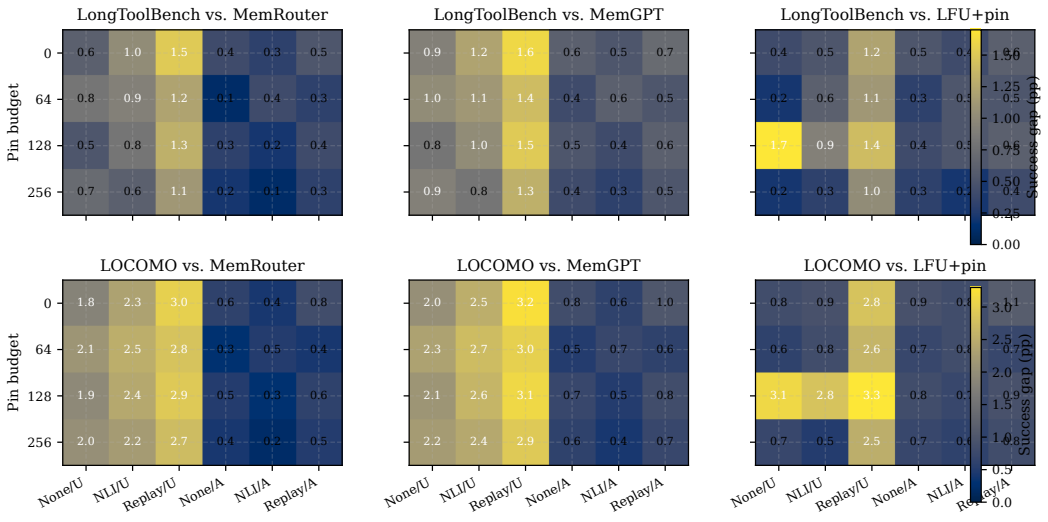

Figure 4: Success-gap response surfaces for two benchmarks and all three audit comparators. Each panel crosses pin budget with validator and promotion access; darker cells indicate larger RetainScope–comparator gaps. The color scale is shared within benchmark.

Table 6: Session-20 success gaps (RetainScope minus comparator, pp) across the retention-rights grid. Column labels combine validator and promotion access: U is unmatched and A is aligned. Every cell averages five seeds.

Pins	None/U	NLI/U	Replay/U	None/A	NLI/A	Replay/A
LongToolBench, comparator: MemRouter						
0	0.6	1.0	1.5	0.4	0.3	0.5
64	0.8	0.9	1.2	0.1	0.4	0.3
128	0.5	0.8	1.3	0.3	0.2	0.4
256	0.7	0.6	1.1	0.2	0.1	0.3
LongToolBench, comparator: MemGPT-reimpl						
0	0.9	1.2	1.6	0.6	0.5	0.7
64	1.0	1.1	1.4	0.4	0.6	0.5
128	0.8	1.0	1.5	0.5	0.4	0.6
256	0.9	0.8	1.3	0.4	0.3	0.5
LongToolBench, comparator: LFU+pin						
0	0.4	0.5	1.2	0.5	0.4	0.6
64	0.2	0.6	1.1	0.3	0.5	0.5
128	1.7	0.9	1.4	0.4	0.5	0.6
256	0.2	0.3	1.0	0.3	0.2	0.4
LOCOMO, comparator: MemRouter						
0	1.8	2.3	3.0	0.6	0.4	0.8
64	2.1	2.5	2.8	0.3	0.5	0.4
128	1.9	2.4	2.9	0.5	0.3	0.6
256	2.0	2.2	2.7	0.4	0.2	0.5
LOCOMO, comparator: MemGPT-reimpl						
0	2.0	2.5	3.2	0.8	0.6	1.0
64	2.3	2.7	3.0	0.5	0.7	0.6
128	2.1	2.6	3.1	0.7	0.5	0.8
256	2.2	2.4	2.9	0.6	0.4	0.7
LOCOMO, comparator: LFU+pin						
0	0.8	0.9	2.8	0.9	0.8	1.1
64	0.6	0.8	2.6	0.7	0.8	0.7
128	3.1	2.8	3.3	0.8	0.7	0.9
256	0.7	0.5	2.5	0.7	0.6	0.8

Table 7: Rights-grid uncertainty and action realization. Shrinkage intervals use hierarchical paired bootstrap; pin counts and accepted promotions aggregate five seeds at the 128-pin setting. Regression entries are comparator-minus-RetainScope events per 100 episodes.

Benchmark / comparator	Unmatched gap	Aligned gap	Shrinkage [95% CI]	Regression range	Pins; promotions
LT / MemRouter	0.9	0.3	0.6 [0.1, 1.1]	0.4–2.2	123 [120,127]; 482/736
LT / MemGPT	1.1	0.5	0.6 [0.1, 1.2]	0.5–2.4	122 [119,126]; 471/728
LT / LFU+pin	0.75	0.45	0.3 [0.0, 0.8]	0.4–2.3	124 [121,128]; 465/719
LO / MemRouter	2.4	0.45	1.9 [1.0, 2.9]	0.8–3.1	121 [117,126]; 438/691
LO / MemGPT	2.6	0.65	1.9 [0.9, 3.0]	0.6–3.0	120 [116,125]; 426/684
LO / LFU+pin	1.7	0.8	0.9 [0.3, 1.6]	0.7–3.0	121 [118,126]; 419/677

Table 8: Mechanism and retention audits on LongToolBench. Panel A perturbs utility supervision and optional parameter updates; Panel B measures pin recall, rare-critical false pruning, and leakage across adversarial-write rates.

A. Mechanism test (LongToolBench s20)

Condition	s20 (%)	Δ vs. RetainScope
RetainScope (reference)	51.9 ± 1.7	0.0
w/o benefit term (reuse prob only)	49.8 ± 2.0	-2.1
w/o replay-derived supervision	49.5 ± 1.9	-2.4
25% utility-label corruption	50.4 ± 1.9	-1.5
50% utility-label corruption	48.1 ± 2.2	-3.8
No parameter updates at all	51.4 ± 1.7	-0.5

B. Retention audit (LongToolBench s20)

ρ (%)	Pin recall	Rare-crit. FP	Leak
0	97.4%	2.6%	0.49%
5	96.9%	3.1%	0.49%
10	96.1%	3.8%	0.73%
20	93.7%	5.9%	1.21%

A.3 Additional Appendix Diagnostics

A.4 Additional Experimental Details

Matched-budget methods share the base LLM, embedding model, index, prompts, tools, task instances, redaction pass, and allocated compute ceiling. The aligned intervention additionally assigns RetainScope, MemRouter, and MemGPT-reimpl the same pin, validator, and promotion rights. Table 9 records the common online stack and the full-versus-aligned capability configurations.

Table 9: Configured parity for operating-point and capability-aligned comparisons. The 4.0 GB cap is controller-managed; the 16.0 GB ceiling is an OOM guard. The common allocated compute ceiling remains available to every managed method; dashes in the sleep-budget rows denote operating points that realize no corresponding sleep compute.

Parity dimension	Store-all	TTL	Summ.	Offline	LFU+pin	MemRouter / RetainScope
Base LLM	Llama-3.1-70B-Instruct (temp 0.2, top_p 0.9)					
Embedder / reranker	bge-large-en-v1.5 / bge-reranker-v2-m3					
Index	FAISS HNSW, $M = 32$, $ef = 128$					
Prompts / tools / task set	identical across methods					
Soft cap (GB)	—	4.0	4.0	4.0	4.0	4.0
Hard ceiling (GB)	16.0	16.0	16.0	16.0	16.0	16.0
Sleep budget (GPU min/day)	—	—	25	25	—	25
Sleep budget (CPU min/day)	—	—	10	10	10	10
Typed items (`Epi`/`Sem`/`Proc`)	Yes	Yes	Yes	Yes	Yes	Yes
Redaction before write	Yes	Yes	Yes	Yes	Yes	Yes
Pinned constraints	No	No	No	No	Yes	Full: 128; aligned: 128
Validation rights	None	None	None	None	NLI	Full: method-native; aligned: NLI
Promotion rights	Basic	Basic	Summary	Schema	Schema	Full: method-native; aligned: shared

The full configuration preserves each controller's integrated retention policy and answers which system to deploy. The aligned configuration changes pin, validation, and promotion access jointly, then reruns each controller's native ranking logic over the common feasible action set.

The utility model uses a reuse-probability predictor and a benefit regressor, both implemented with LightGBM and calibrated by isotonic regression. Default selection settings are $K = 512$ candidates per sleep cycle, cosine-similarity clustering with threshold 0.85, a per-cluster quota of 4, and a reserved 10% budget for rare or safety-critical clusters. Verifier replay is capped at the top 64 high-impact `ToolRecipe` candidates per cycle. Optional distillation uses LoRA rank 8 with a maximum of 400 update steps and a 24-example canary suite. LongToolBench v2.1 is the main derivative benchmark in the paper and is treated here as a frozen snapshot: the 200-identity roster, 20-session order, and identity-level split assignment are fixed before replay labeling and are unchanged across all reported runs. Each identity is rolled out over 20 ordered sessions, and the memory store surviving after session t is the only persistent state exposed to session $t + 1$ for that same task identity. Later sessions preserve the same broad goal but perturb tool arguments, backend state, documentation, or user constraints so that old memories can remain helpful, become stale, or become contradictory. A representative mutable-trace pattern is that a previously correct `ToolRecipe` becomes stale after an API field, approval constraint, or backend state changes. Replay supervision is created only on the training split by deterministic re-execution of logged tool traces with a candidate memory item retrievable versus withheld; calibration and headline evaluation use disjoint task identities and never feed labels back into the scorer. The split discipline is identity-level throughout. Memories never cross identities, replay labels use the training split, and calibration thresholds are selected

before headline evaluation. Every trajectory logs memory mutations, retrievals, validator outcomes, promotion decisions, and random seeds.

A.5 Extended Related Work

This paper sits at the intersection of retrieval-augmented / non-parametric memory (Lewis et al., 2020; Izacard et al., 2023; Khandelwal et al., 2020; Borgeaud et al., 2022; Ram et al., 2023; Mallen et al., 2023; Asai et al., 2024; Liu et al., 2024), long-context and external-memory transformers (Dai et al., 2019; Rae et al., 2020; Ainslie et al., 2020; Zaheer et al., 2020; Wu et al., 2022), staged-memory systems (Zhong et al., 2023; Wang et al., 2023; Fang et al., 2026; Salama et al., 2025), agent memory-management systems (Hu et al., 2025; Packer et al., 2023), tool-using and reflective agents (Yao et al., 2023; Schick et al., 2023; Shinn et al., 2023; Wang et al., 2023; Park et al., 2023; Madaan et al., 2023; Yang et al., 2024), replay-based continual learning (Kirkpatrick et al., 2017; Rolnick et al., 2019; Lopez-Paz et al., 2017; Chaudhry et al., 2019; Rebuffi et al., 2017; Aljundi et al., 2019), and benchmark / evaluation papers spanning large-scale tool use, multi-turn tool interaction, very long conversational memory, realistic tool-agent-user interaction, software-task reuse, and conversational-memory breadth (Qin et al., 2023; Wang et al., 2024; Maharana et al., 2024; Yao et al., 2025; Jimenez et al., 2024; Tan et al., 2025). MemoryBank, SCM, LightMem, and MemInsight emphasize staged storage or summarization; HiAgent studies live working-memory control; MemGPT studies hierarchical paging; long-context and external-memory transformers mostly enlarge accessible context or cached state; replay-based continual-learning methods focus on parameter adaptation or rehearsal buffers; and MemBench targets conversational recall rather than mutable tool traces with verifier replay. Those papers motivate why memory shape matters, but they do not by themselves separate controller quality from admissible-retention privileges. The comparison literature can be organized by intervention strength. Native-stack studies compare complete agents, preserving deployment realism. Shared-harness studies normalize model, retrieval, prompts, and resource budgets, improving component attribution. Capability alignment adds the missing action-set intervention: it equalizes pinning, validation, and promotion rights that determine which memories can enter protected tiers before ranking. This distinction is causally different from ordinary fairness ablations, which perturb one configuration while leaving its feasible action set intact. The paired full/aligned rows instead estimate how attribution changes when both methods act over the same feasible retention set. Demonstrating this estimand on LongToolBench and LOCOMO, and against routing plus paging/self-edit comparators, turns the conceptual distinction into a reusable empirical protocol.

A.6 Additional Results

Table 10: LOCOMO full-controller operating points and utility transfer. Retained target gain is $(s20^{\text{transfer}}_{\text{target}} - s20^{\text{Store-all}}_{\text{target}})/(s20^{\text{RetainScope,in}}_{\text{target}} - s20^{\text{Store-all}}_{\text{target}})$.

A. LOCOMO v2 session-20 operating points

Method	Success (%)	Regr. / 100	Memory (GB)
Store-all	43.2 ± 2.0	18.6 [14.1, 23.4]	14.3 [12.1, 16.8]
Periodic summ.	44.5 ± 1.7	14.1 [10.3, 18.8]	4.9 [4.1, 5.9]
Offline consol.	44.8 ± 1.8	13.5 [9.8, 17.9]	4.2 [3.5, 5.0]
LFU+pin	45.6 ± 1.6	11.8 [8.6, 15.8]	3.4 [2.8, 4.1]
MemRouter	45.9 ± 1.7	11.2 [8.1, 15.0]	3.7 [3.0, 4.5]
RetainScope	48.7 ± 1.8	8.6 [5.9, 11.8]	2.9 [2.3, 3.5]

B. Utility/policy transfer

Transfer	Target s20	Retained gain
LT → LO zero-shot	47.3 ± 1.9	75%
LO → LT zero-shot	50.8 ± 1.8	71%
LT → LO + 50 eps	48.0 ± 1.8	87%
LO → LT + 50 eps	51.4 ± 1.7	87%

Table 11: Public τ-bench v0.4 operating points under the shared 20-batch adaptation. "s20" is final-batch success after 19 prior release-ordered batches.

Method	Retail s20 (%)	Airline s20 (%)	Memory (GB)	p95 (s)	p99 (s)
Store-all	39.1 ± 2.2	33.2 ± 2.6	13.4 [11.6, 15.5]	2.44	4.71
BGE-retriever-only	38.7 ± 2.3	32.9 ± 2.7	4.0 [3.3, 4.8]	1.89	3.68
MemGPT-reimpl	42.3 ± 2.1	35.7 ± 2.4	3.8 [3.1, 4.6]	1.94	3.85
LFU+pin	43.6 ± 1.9	37.4 ± 2.1	3.4 [2.8, 4.1]	1.78	3.48
MemRouter	44.0 ± 1.9	38.0 ± 2.1	3.6 [2.9, 4.3]	1.81	3.55
RetainScope	46.8 ± 1.8	40.6 ± 1.9	2.8 [2.2, 3.3]	1.69	3.36

Table 12: Held-out utility calibration across three task families. Values are mean±standard deviation over five seeds.

Metric	LongToolBench	SWE-bench	LOCOMO
AUC	$0.836 \pm .019$	$0.811 \pm .024$	$0.854 \pm .017$
ECE	$0.047 \pm .012$	$0.058 \pm .014$	$0.041 \pm .010$
Brier	$0.168 \pm .023$	$0.184 \pm .027$	$0.152 \pm .021$

The utility model remains rank-informative and calibrated across tool use, conversation, and recurring software tasks. LOCOMO has the best ECE (0.041), while SWE-bench's 0.058 ECE quantifies the expected shift for repository-centered reuse; both support the transfer results in Table 10.

On SWE-bench Lite, validated procedural memory improves resolve rate from 22.7% to 25.2%, raises median template reuse from 11.6% to 19.7%, and reduces regressions from 5.6 to 3.6 per 100 issues.

Table 13: Procedural-memory transfer on recurring SWE-bench Lite issues. Regressions are mean±standard deviation per issue batch; template reuse is median [Q25, Q75].

Method	Resolve rate (%)	Template reuse (%)	Regressions / 100 issues
Store episodic	21.4 ± 1.1	8.3 [4.1, 13.6]	6.4 ± 1.8
Periodic summarization	22.7 ± 0.9	11.6 [5.8, 18.4]	5.6 ± 1.5
RetainScope	25.2 ± 1.3	19.7 [12.4, 28.3]	3.6 ± 1.1

The increased template reuse shows that consolidation preserves actionable abstractions across recurring repository issues, complementing the long-horizon retention audits.

The two accounting views are complementary: persisted-store composition explains what consolidation leaves behind, while runtime memory measures the deployed process.

RetainScope shifts the full CCDF left and improves p99 by 28.4% over Store-all and 3.2% over MemRouter, showing that the operating-point advantage extends through the latency tail.

The bootstrap intervals show consistent full-controller ordering across both domains. Together with the memory and latency columns in Table 11, the plot supports the transport of RetainScope's operating point to public tool-agent workloads.

The 4.0 GB point lies at the accuracy–latency knee: 8.0 GB adds 0.2 pp success but raises p95/p99 substantially, whereas 1.5 GB over-compresses rare memories. This tradeoff motivates the common audit cap.

A.7 Low-Budget Operating Points

Table 15 summarizes two lower-cost operating points of the same audited recipe on LongToolBench v2.1 under the identical 4.0 GB cap and the identical base LLM, embedder, index, prompts, and tools. The *cheap* row reduces the sleep-compute budget from 25 to 5 GPU min/day and caps verifier replay

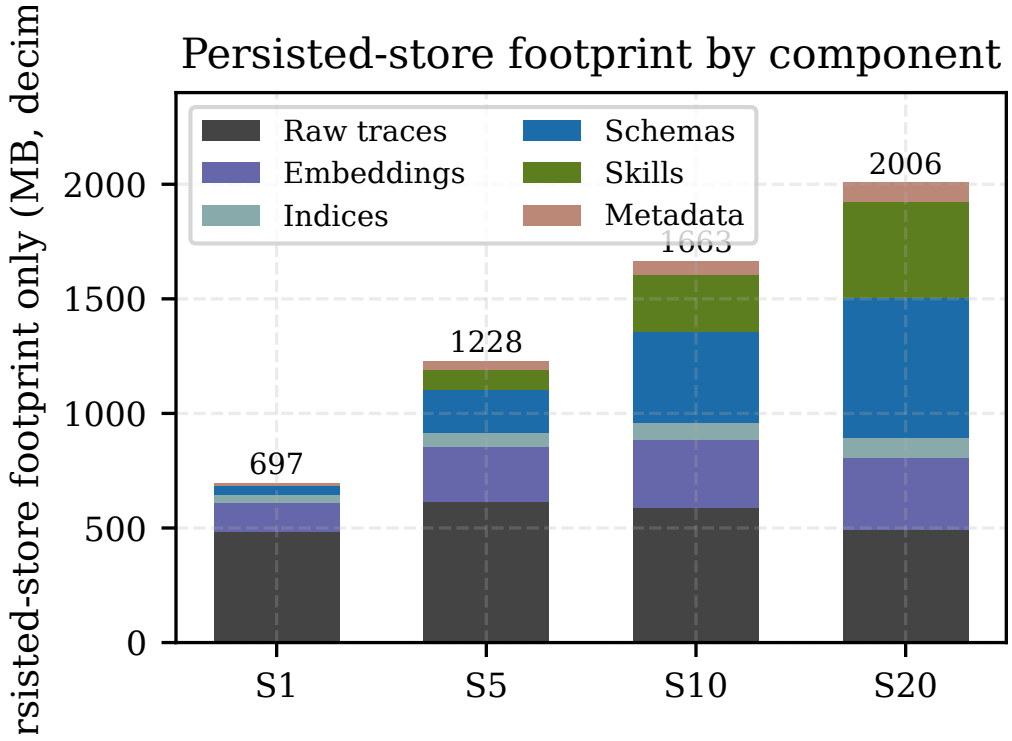

Figure 5: Persisted-store composition over LongToolBench sessions for RetainScope. Raw traces flatten as validated schemas and skills grow. Session 20 totals 2006 MB for persisted components; Table 2's 2.6 GB end-to-end value additionally includes resident controller state, allocator overhead, and temporary replay workspaces.

Table 14: Sensitivity to the end-to-end soft cap on LongToolBench session 20. Retrieval p95/p99 pool per-request latency across five seeds. Sleep overhead is the percentage increase in complete sleep-cycle wall time relative to replaying the same cycle with consolidation bypassed; the table reports its median and p95 across cycles.

Soft cap (GB)	Success (%)	p95 lat. (s)	p99 lat. (s)	Sleep ovh. p50 / p95
1.5	50.4 ± 1.9	1.41	2.89	+8.3% / +17.1%
2.5	51.3 ± 1.8	1.52	3.11	+7.6% / +15.6%
4.0	51.9 ± 1.7	1.63	3.30	+7.2% / +14.8%
8.0	52.1 ± 1.5	1.94	3.94	+5.6% / +12.3%

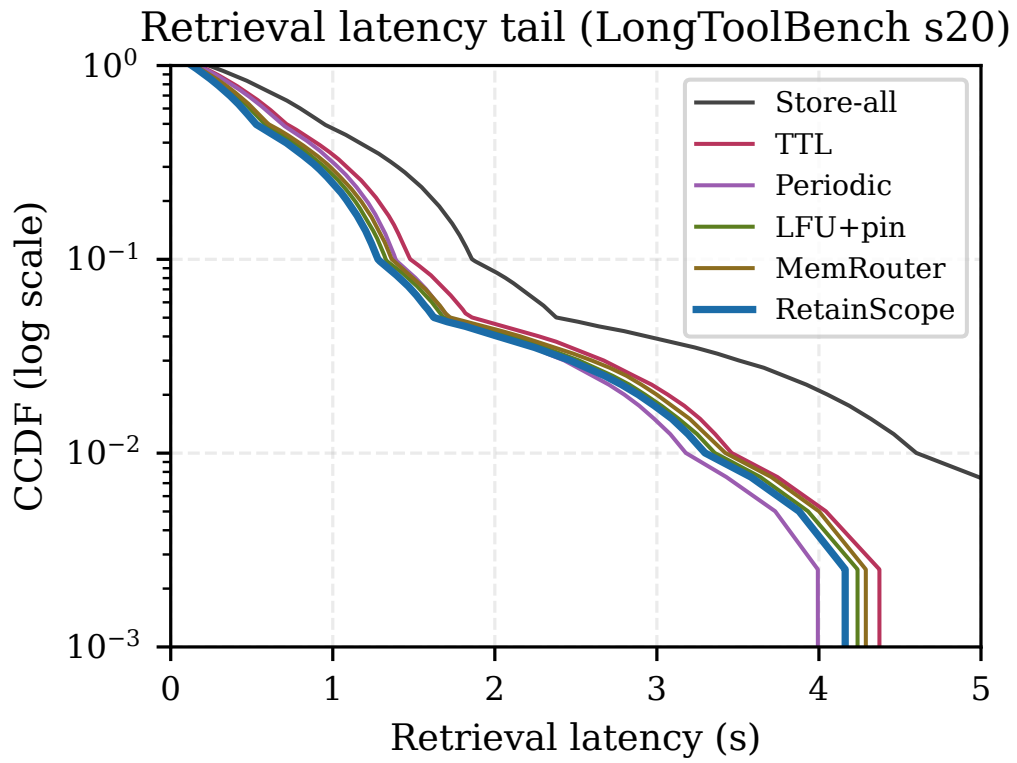

Figure 6: Retrieval-latency CCDF on 80,000 LongToolBench session-20 requests per method across five seeds. The (p50, p95, p99) tuples in seconds are Store-all $(0.94, 2.38, 4.61)$, TTL $(0.71, 1.83, 3.47)$, Periodic $(0.68, 1.72, 3.18)$, LFU+pin $(0.58, 1.69, 3.36)$, MemRouter $(0.60, 1.71, 3.41)$, and RetainScope $(0.52, 1.63, 3.30)$.

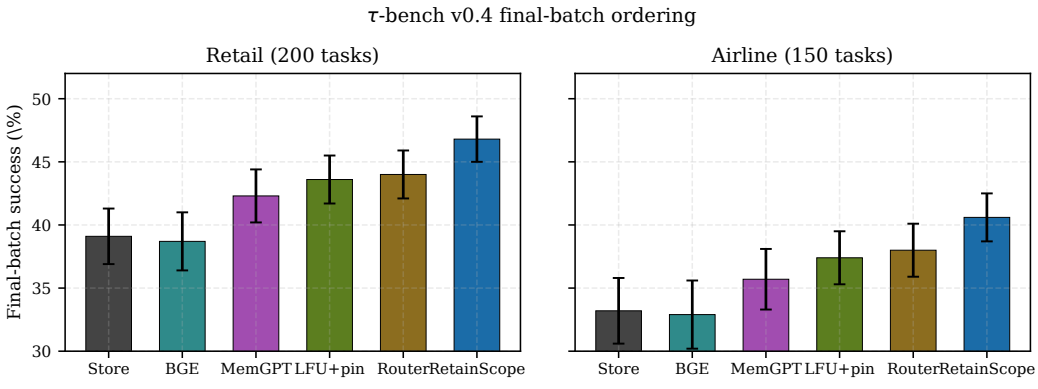

Figure 7: Public τ-bench v0.4 under the 20-batch shared-harness adaptation in Table 11. Retail uses 20 release-ordered 10-task batches; Airline uses 20 release-ordered batches of 7 or 8 tasks. Bars show final-batch success and 95% bootstrap intervals over five seeds.

at the top-16 `ToolRecipe` candidates per cycle instead of top-64. The *CPU-only* row removes the GPU budget entirely (validator-lite replaces GPU-accelerated verifier replay with a 3-second CPU NLI check), at the cost of a slightly larger CPU budget of 45 CPU min/day.

Table 15: Low-budget RetainScope operating points on LongToolBench v2.1. All managed rows use the same 4.0 GB cap and online stack. The 5-GPU-minute and CPU-only rows preserve 81.6% and 71.1% of the default gain over Store-all, and 93.3% and 88.0% of its p95 latency improvement.

Variant	s20 (%)	Regr. / 100	Mem. (GB)	p95 (s)	p99 (s)
Store-all (ref.)	48.1 ± 1.9	15.3 [11.4, 19.7]	12.8 [11.2, 15.1]	2.38	4.61
RetainScope default	51.9 ± 1.7	7.4 [5.1, 10.2]	2.6 [2.1, 3.2]	1.63	3.30
RetainScope-cheap (5 GPU min, top-16 replay)	51.2 ± 1.8	8.9 [6.3, 12.1]	2.7 [2.2, 3.3]	1.68	3.37
RetainScope-CPU-only (0 GPU, validator-lite, 45 CPU min)	50.8 ± 1.9	9.6 [6.8, 12.9]	2.8 [2.3, 3.4]	1.72	3.44

Most of the fixed-budget benefit persists when replay is throttled or moved to CPU validation, giving deployers a measured compute–quality curve around the default operating point.

A.8 Security Evidence Matrix

A.9 Failure Accounting and Case Studies

In one representative session, a cluster of repeated `http.post` failures was consolidated into a validated retry-with-backoff `ToolRecipe` that was later reused on the affected task family and improved local completion on that cluster. A representative false-prune failure was a rare safety constraint, "never auto-commit on protected branches," that was initially under-ranked because of low access frequency and was pruned before we strengthened the pinned-eligibility rule with safety-verb features. The validator also blocks malformed promotions: one proposed `BugFixTemplate` invoked a non-existent tool `git.squash_all`, and the live-signature checker rejected it before promotion. Under adversarial writes at $\rho = 10\%$, a small number of misleading `ToolRecipe` candidates survived NLI consistency but were still rejected by verifier replay after failing the targeted tests. Together, the ledger and case studies localize residual errors in stale schemas, borderline validator decisions, and rare high-value items. This decomposition connects the aggregate stress results to concrete enforcement failures and identifies validator replay, pin eligibility, and live-signature checks as the highest-leverage repair points.

A.10 External-Baseline Parity (τ-bench MemGPT-reimpl)

MemGPT-reimpl preserves the architecture's hierarchical main, recall, and archival tiers plus system-prompt-driven self-edit. The shared harness normalizes the online backbone, retriever, reranker, index, tool adapter, and 4.0 GB cap so that paging policy remains the comparator's defining logic. The operating-point row retains native rights; the capability-aligned row adds the same 128-pin, NLI, and promotion interface used by RetainScope and MemRouter. This construction preserves comparator-defining paging and self-edit behavior across the two evaluation regimes summarized in Table 19.

A.11 Cross-Model Transfer

Replacing the inference backbone with Qwen2.5-72B-Instruct preserves the full-controller ordering without utility-model retuning. Across LongToolBench, LOCOMO, and τ-Retail, 71–92% of RetainScope's Llama-based advantage over MemRouter carries over, averaging 84%.

Table 16: Security model and enforcement boundary. The comparator column names the strongest relevant row sharing the online stack; the trace column points to quantitative evidence. Added overhead is the median percentage increase in complete sleep-cycle wall time relative to the same cycle with the listed guard bypassed.

Protected object	Adversary capability	Trusted boundary	Tested property	Comparator / setting	Evidence trace
Pinned constraint	Delayed contradictory write	Newer verifier-passed evidence	Consistent pin demotion	MemRouter + NLI; 10% contradiction	Table 3; Table 18
Promotion queue	Adaptive malicious writes	NLI, replay, tool tests	Quarantine before promotion	MemRouter + NLI; $\rho = 10\%$	Tables 3 and 17
ToolRecipe	API or tool-state drift	Live signature and replay oracle	Fresh executable recipe	LFU+pin; major drift	Table 3; Table 18
Semantic schema	Ambiguous promotion candidate	End-of-run held-out oracle	Low verifier FAR/FRR	MemRouter; promotion audit	Table 18
Raw trace	Verbatim canary insertion	Pre-write redaction	Verbatim non-disclosure	MemGPT-reimpl; canary suite	Table 18
Derived schema	Semantic paraphrase of canary	Redaction plus validator	Derived non-disclosure	MemRouter; paraphrase suite	Table 3; Table 18

Stress row	Benign / stressed s20	Added wall-clock overhead	Observed failure	Denominator	95% interval
Contradictory pins	51.9 / 50.6	3.4%	2.7% miss	approx. 780 flagged pairs	[1.7, 4.1]%
Malicious writes, $\rho = 10\%$	51.9 / 49.3	6.8%	81.7% quarantined	490/600	[78.4, 84.6]%
Tool-recipe drift	51.9 / 49.7	4.1%	7.1% stale	86/1,211	[5.8, 8.7]%
Verifier acceptance / rejection	51.9 / 51.4	2.6%	$\sim$2% FAR; $\sim$6% FRR	approx. 640 each	[1.1, 3.3]%; [4.3, 8.1]%
Verbatim canary	51.9 / 51.9	1.4%	0.0% leak	0 / 412	[0.0, 0.89]%
Paraphrased canary	51.9 / 51.6	2.2%	0.49% leak	2 / 412	[0.13, 1.75]%

Table 17: Adversarial-write accounting by attack rate and method. Each tuple reports attempted / quarantined initially / caught later by replay / promoted. Methods without replay have zero in the third position.

ρ	Method	Attempted / quarantined / replay-caught / promoted
0%	RetainScope	0 / 0 / 0 / 0
0%	MemRouter	0 / 0 / 0 / 0
0%	MemGPT-reimpl	0 / 0 / 0 / 0
2%	RetainScope	120 / 104 / 7 / 9
2%	MemRouter	120 / 80 / 0 / 40
2%	MemGPT-reimpl	120 / 46 / 0 / 74
5%	RetainScope	300 / 252 / 19 / 29
5%	MemRouter	300 / 191 / 0 / 109
5%	MemGPT-reimpl	300 / 105 / 0 / 195
10%	RetainScope	600 / 490 / 39 / 71
10%	MemRouter	600 / 354 / 0 / 246
10%	MemGPT-reimpl	600 / 191 / 0 / 409
20%	RetainScope	1,200 / 918 / 91 / 191
20%	MemRouter	1,200 / 628 / 0 / 572
20%	MemGPT-reimpl	1,200 / 337 / 0 / 863

Table 18: Failure accounting for the LongToolBench run. Exact audit sets retain exact denominators; larger cohorts are rounded to the nearest percentage point.

Metric	Rate	Denominator / meaning
Stale-schema rate	$\sim$5%	approx. 1.2k schemas referencing later-changed APIs (cohort rounded)
Contradiction-miss rate	$\sim$3%	approx. 780 flagged contradictory pairs (cohort rounded)
Stale-pin rate	$\sim$2%	approx. 310 pinned items later contradicted by newer verifier-passed evidence (cohort rounded)
False-prune (pinned-eligible)	3.1%	420 released pinned-eligible items in the audit set
False-prune (rare/high-value)	2.6%	190 held-out rare critical items in the audit set
Verifier FAR on promotion	$\sim$2%	approx. 640 promoted schemas later shown wrong by end-of-run oracle (cohort rounded)
Verifier FRR on promotion	$\sim$6%	approx. 640 rejected schemas later shown correct by oracle (cohort rounded)
Verbatim secret leak	0.0%	0 / 412 canaries
Paraphrase / derived leak	0.49%	2 / 412 canaries

Table 19: MemGPT-reimpl parity for the public τ-bench operating point and the capability-aligned audits. Architectural paging and self-edit remain fixed across both columns.

Dimension	Operating-point row	Capability-aligned row
Base LLM	Llama-3.1-70B-Instruct, temperature 0.2, top-p 0.9	
Retrieval	bge-large-en-v1.5; bge-reranker-v2-m3; FAISS HNSW	
Tool adapter / cap	Shared tool schema; 4.0 GB soft cap	
Paging / self-edit	Native main/recall/archival tiers	Native main/recall/archival tiers
Pinned set	Native configuration	Shared 128-pin capability adapter
Validator	Native self-edit checks	Shared NLI-only validator
Promotion interface	Native tier writes	Shared interface mapped to native tiers

Table 20: Cross-model transfer with Qwen2.5-72B-Instruct and all memory, retrieval, budget, and evaluation components held fixed.

Method	LT s20 (%)	LO s20 (%)	τ-Retail s20 (%)
Store-all	46.8 ± 2.0	41.9 ± 2.1	37.6 ± 2.3
MemRouter	49.3 ± 1.8	44.5 ± 1.8	42.5 ± 2.0
RetainScope	50.5 ± 1.7	47.0 ± 1.8	44.5 ± 1.9

The transferred gaps are $+1.2$ pp on LongToolBench, $+2.5$ pp on LOCOMO, and $+2.0$ pp on τ-Retail. This result supports the portability of the complete RetainScope operating point along the backbone axis.

A.12 Implementation Invariant

Let $\mathcal{F}_t$ be the retained sets satisfying footprint budget B_f, sleep-compute budget B_s, per-cluster quotas, admissible actions, and pin inclusion $\mathcal{P}_t \subseteq \mathcal{M}_{t+1}$. RetainScope initializes $S_0 \leftarrow \mathcal{P}_t$ and scans candidates in decreasing $\hat{U}(m)$ order, adding an item exactly when the updated set remains in $\mathcal{F}_t$. By induction, every intermediate set is feasible and contains every pin; the final set S_{RS} therefore satisfies $S_{\mathrm{RS}} \in \mathcal{F}_t$ and $\mathcal{P}_t \subseteq S_{\mathrm{RS}}$. This invariant connects the implemented selector directly to the feasibility definition in Eq. (1).

A.13 Total Compute and Licenses

Tables 21 and 22 summarize the compute ledger and asset-license disclosures for the submitted experiments and dependencies.

Table 21: Total compute measured on the 4×A100-80GB SXM node (CUDA 12.4, driver 550.90.07). Pilot, hyperparameter, and reduced-cost runs are included. Rights-grid cells reuse logged trajectories and are charged within the LongToolBench, LOCOMO, calibration, and retention-audit suites.

Experiment suite	A100-80GB hours	CPU hours
LongToolBench v2.1 main table (7 configurations × 5 seeds)	136.5	42.0
LOCOMO v2 main table (6 methods × 5 seeds)	122.0	38.0
τ-bench v0.4 Retail + Airline (6 methods × 5 seeds)	84.5	28.5
Component ablations (Table 3, Panel A)	113.9	30.0
Mechanism test (Table 8, Panel A)	94.8	25.0
Safety stress suite & poisoning sweep (5 ρ values × 3 methods)	140.4	46.0
Transfer + cross-model (Qwen2.5-72B)	69.6	22.0
Retention audit (Table 8, Panel B)	28.7	10.0
Calibration runs (held-out AUC/ECE/Brier)	19.2	8.0
Pilot / hyperparameter search (not counted toward headline numbers)	54.4	18.0
Reduced-cost reproduction (single A100)	27.5	9.5
Total charged to this submission	891.5	277.0

Table 21 separates the research-time evidence cost from the deployed controller's resource envelope. The complete program uses 891.5 A100-80GB hours and 277.0 CPU hours, with the safety sweep, main benchmark runs, and component analyses contributing the largest shares. Reporting these costs beside the 25 GPU + 10 CPU minute daily controller ceiling makes the accuracy, attribution, stress-testing, and operational-efficiency evidence jointly interpretable on the measured hardware stack.

Table 22: Versions and licenses of third-party models, libraries, benchmarks, and baseline implementations.

Asset	Version	License
Llama-3.1-70B-Instruct	2024-07	Meta Llama 3.1 Community License
Qwen2.5-72B-Instruct	2024-09	Tongyi Qianwen License
bge-large-en-v1.5	2023-09	MIT
bge-reranker-v2-m3	2024-02	MIT
FAISS	1.8.0	MIT
vLLM	0.5.2	Apache 2.0
PyTorch	2.3.1	BSD-3-Clause
LOCOMO v2	v2.0	Apache 2.0
τ-bench	v0.4	Apache 2.0
SWE-bench Lite	2024-04	MIT
MemGPT (for reimpl baseline)	0.3.7	Apache 2.0