WHEN MORE EXPERIENCE HURTS: DIAGNOSING EXPERIENCE GROWTH AND READER ALIGNMENT IN SELF-EVOLVING LANGUAGE AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Experience-based self-evolution allows large language models (LLMs) to adapt after deployment without updating their parameters, while keeping accumulated experience readable and auditable. However, whether models can turn a growing memory bank into sustained improvement across model scales remains unclear. In this paper, we study these dynamics across four tasks and three representative strategies, evaluating successive memory checkpoints and varying the models that generate trajectories, edit memory, and use the resulting experience. We find that accumulating experience creates new successes without reliably preserving earlier ones, and that stronger experience sources do not consistently improve weaker readers. Controlled interventions show that the same experience can cause different losses across readers, while experience that helps alone can become ineffective alongside other entries. Guided by these findings, we explore explicit protection of experience applicability boundaries during reuse, with preliminary improvements suggesting a promising direction for more reliable self-evolution.

1 INTRODUCTION

As LLM agents increasingly tackle interactive tasks in web and computer environments (Zhou et al., 2024; Rawles et al., 2025), learning from past interactions offers a promising way to improve their behavior after deployment (Ouyang et al., 2026). Experience-based self-evolution follows this idea without updating model parameters: an agent records task trajectories and environmental feedback, extracts reusable experience, and uses it in later tasks (Shinn et al., 2023; Zhao et al., 2024). Storing this experience as text makes it easy to inspect, revise, and transfer. Figure 1(a) illustrates this loop and three representative ways to supply experience to a reader: trajectories with insights, a full experience pool, or selected entries (Zhao et al., 2024; Cai et al., 2025; Zhang et al., 2026b). All three approaches aim to turn past interactions into better future answers without changing model weights, making them particularly attractive for closed-source models.

However, adding experience does not ensure that an agent will keep improving. The model must extract useful experience, update its memory without losing existing benefits, and recognize when each entry applies (Figure 1(b)). Prior studies show that distracting evidence can hurt performance and that experience can transfer differently across models (Yoran et al., 2024; Chang et al., 2026). What remains unclear is how these difficulties affect continued learning from experience across model scales. A model may solve new questions while starting to fail on questions it previously answered correctly, as illustrated by the help–harm comparison in Figure 1(c). Understanding this balance requires following which answers improve or deteriorate as memory grows, then examining how both memory construction and reader use contribute to these changes.

In this paper, we study these changes throughout self-evolution. We evaluate the same questions at successive memory checkpoints and vary the models that construct experience and the contexts supplied to readers. Across four tasks and three representative strategies, new successes coexist with losses on previously solved questions, and stronger experience sources do not consistently improve weaker readers. Controlled comparisons further reveal *reader misalignment*: readers differ in how well they use the same supplied experience, and experience that helps alone can become ineffective alongside other entries. We examine reasoning and tool-use traces to understand how these changes

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.

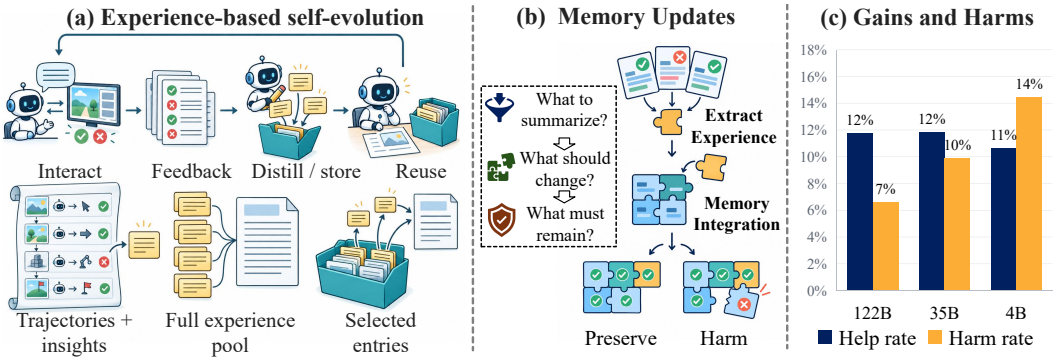


Figure 1: An overview of experience-based self-evolution and its challenges. (a) An illustration of how agents collect, summarize, and reuse experience. (b) The abilities needed to gain useful experience while preserving earlier benefits. (c) New successes and new failures coexist, as shown by help–harm rates averaged across datasets in our initial complete analysis cohort.

appear in task behavior. Guided by these findings, we explore a lightweight way to make the scope of retrieved experience clearer. Preliminary improvements suggest that explicitly explaining when experience applies is a promising direction for more reliable experience reuse. We contribute:

- **Question-level experience dynamics.** We follow the same set of test samples as memory grows and show that models solve new problems while starting to fail on ones they previously solved, with smaller models losing more of their earlier successes.
- **Reader–experience alignment.** We isolate reader-dependent losses under identical inputs and show how combining entries undermines experience that helps alone, identifying reader alignment and context interference as key barriers to sustained improvement.
- **Diagnosis-guided refinement.** We connect memory interventions to changes in reasoning and tool use, then demonstrate gains from scope protection, turning our diagnosis into a concrete direction for improving experience reuse.

2 Related Work

From reflection to experience learning. Early textual self-improvement methods used execution feedback to guide subsequent attempts: Reflexion retains verbal reflections, while ExpeL abstracts reusable insights from task trajectories (Shinn et al., 2023; Zhao et al., 2024). Work in 2025 developed richer experience-construction procedures: ReasoningBank distills strategies from self-judged successes and failures, and Training-Free GRPO uses group-relative trajectory comparisons (Ouyang et al., 2026; Cai et al., 2025). ACE explicitly addresses information loss during repeated rewriting through incremental playbook updates (Zhang et al., 2026a). In 2026, MemRL further separates a frozen reasoning model from a memory utility mechanism updated through task feedback (Zhang et al., 2026b). These approaches offer different ways to turn outcomes into future guidance; our analysis follows whether their accumulated guidance preserves earlier successes as well as creating new ones.

Memory organization and retrieval. Early systems such as MemoryBank and MemGPT addressed persistent information and context-window management (Zhong et al., 2024; Packer et al., 2023). In 2025, A-MEM introduced dynamically linked memory notes, while Mem0 combined extraction, consolidation, and retrieval of conversational information (Xu et al., 2025; Chhikara et al., 2025). More recent approaches diversify the representation: HyMEM studies hybrid structured memory for GUI agents, and ExpGraph organizes reusable experience in a graph (Zhu et al., 2026; Feng et al., 2026). AnchorMem separates atomic retrieval anchors from preserved source context, explicitly distinguishing what retrieves a memory from what the model subsequently reads (Shen et al., 2026). These systems span factual, conversational, and procedural memory rather than identical learning settings. They motivate separating bank contents from delivered context; we use that separation for controlled comparisons rather than proposing another storage structure.

Reliability of memory use. Context studies document sensitivity to evidence position and irrelevant passages (Liu et al., 2024; Yoran et al., 2024). Memory-specific evaluations examine hallucinations, forgetting, and broader failures through HaluMem, PersistBench, and MemFail (Chen et al., 2026; Pulipaka et al., 2026; Garg et al., 2026). State Contamination and MemTrace further investigate problematic persistent state and error tracing across memory operations (Wang et al., 2026; Deng et al., 2026). These studies show why successful retrieval need not imply useful memory. We complement them by tracking helped and harmed questions during construction and probing their context-dependent outcomes.

Model capability and memory transfer. MemCollab reports degradation under naive cross-model transfer and constructs shared memory using contrastive multi-model trajectories and task-aware retrieval (Chang et al., 2026). Circuit-level analysis across the Qwen-3 family distinguishes routing from content-grounding computations in memory operations (Mao et al., 2026). Complementing these construction and internal-mechanism perspectives, we combine checkpoint dynamics with identical reader inputs and context-composition controls to characterize reader-dependent utility without assuming a unique internal cause.

3 A Framework for Analyzing Experience Dynamics

Experience-based self-evolution has two main processes: constructing experience and using it to solve new tasks. In this section, we first formalize this loop, and then derive checkpoint comparisons and controlled interventions that distinguish accumulation from realized utility (Figure 2).

3.1 The textual self-evolution process

We model textual self-evolution as the construction and subsequent use of an external memory. Let M_t denote the memory bank after t construction tasks, including experience text and any retrieval metadata. A trajectory *builder* b attempts task x_t using the current bank M_{t-1}, producing a trajectory τ_t and receiving feedback f_t. An *editor* e then updates the bank:

$$M_t \sim U_e(\cdot \mid M_{t-1}, x_t, \tau_t, f_t). \tag{1}$$

At evaluation, the bank is frozen. Let $C_r(x, M_t)$ be the experience text actually supplied to *reader* r for question x, after selection, ordering, formatting, and truncation. The reader generates its answer from the question and this context:

$$\hat{y}_{r,t} \sim \pi_r(\cdot \mid x, C_r(x, M_t)). \tag{2}$$

Here U_e is the editor's update procedure and π_r is the reader's answer-generation distribution; sampling allows stochastic updates and answers. One model may fill all three roles. Updates may change experience text or retrieval metadata, while evaluation leaves both unchanged. In this paper, we choose three representative baselines with distinct memory representations, update granularity, and exposure mechanisms.

Trajectory-and-insight memory with example retrieval. ExpeL (Zhao et al., 2024) retains task trajectories and distills reusable insights from accumulated experience. Construction combines trajectory accumulation with insight consolidation, which can add, revise, or remove rules. The reader receives retrieved trajectory examples together with insights. This separates concrete demonstrations from generalized advice: changes to either component can alter the supplied context.

Experience-pool memory with full-pool exposure. TF-GRPO (Cai et al., 2025) compares groups of task rollouts using their feedback and updates a textual experience pool. An update can affect multiple entries, while the reader receives the pool rather than a query-selected subset, subject to the context budget. Pool revisions therefore directly change the available reader context, although formatting and truncation determine which content is actually exposed. Related approaches such as ACE (Zhang et al., 2026a) also maintain and refine a structured textual playbook, representing a broader family of experience-pool optimization methods.

Entry-level memory with selective retrieval. MemRL (Zhang et al., 2026b) summarizes trajectories into individual experience entries and updates utility estimates from task outcomes. Retrieval combines semantic relevance with learned utility to select a small subset for the reader. Updating an entry's utility is distinct from rewriting its text: memory growth and metadata changes

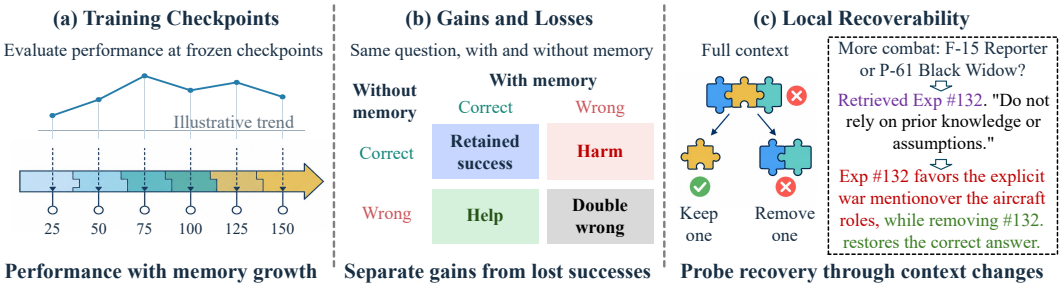


Figure 2: Our framework for analyzing experience dynamics. (a) Evaluating memory at successive construction checkpoints. (b) Comparing answers with and without memory to distinguish new gains from lost successes. (c) Probing whether failed answers recover when the supplied experience changes, illustrated by a HotpotQA case with Qwen3.5-35B and MemRL.

can affect which entries are subsequently retrieved, even when existing experience text remains unchanged. Selective entry retrieval is also used in systems such as Mem0 (Chhikara et al., 2025), without implying the same utility-learning mechanism. MemQ (Liao et al., 2026) extends utility learning with credit propagation over a memory provenance graph.

These distinctions motivate separating memory construction from reader exposure. A change in M_t need not produce the same change in C_r: its effect depends on the update granularity, selection mechanism, and context budget. We examine both checkpoint-level performance and controlled changes to the experience, rather than treating memory growth as guaranteed improvement.

3.2 Observing dynamics at fixed checkpoints

We segment one construction stream rather than rebuilding a bank for each measurement (Figure 2(a)). Banks saved after 25–150 construction examples, in steps of 25, are evaluated on the same 300 questions without further memory updates. No memory is the reference for total utility; an earlier checkpoint is the reference for observed construction-time change. The latter includes decoding variability, while best-to-final gaps are descriptive, not attainable early-stopping policies.

For binary correctness, pairing each answer with its no-memory counterpart yields retained success, harm, help, or double wrong (Figure 2(b)), with

$$\text{Accuracy}_r(M_t) - \text{Accuracy}_r(\varnothing) = \text{HelpRate}_r(t) - \text{HarmRate}_r(t). \tag{3}$$

We follow question identities across checkpoints to distinguish retained, lost, and newly gained successes, rather than inferring preservation from the net score. Rates over all questions and conditional rates over baseline successes or errors are reported separately; continuous HotpotQA F1 requires gain/loss magnitudes instead of binary rates. This separates overall utility from success preservation (definitions in Appendix B).

3.3 Controlled interventions on sources and reader context

The distinction between updating the bank (Equation 1) and answering from its supplied context (Equation 2) guides our interventions. Replacing e alone or both b and e tests stronger experience sources with the reader fixed, but does not isolate reader ability. Likewise, a shared bank M_t need not yield identical contexts C_r. We therefore compare readers under matched content, ordering, formatting, and truncation. Reader-dependent losses under this control provide behavioral evidence of *reader misalignment*, without certifying advice correctness or identifying a unique internal cause.

With the reader fixed, single-entry and leave-one-out probes test context sensitivity (Figure 2(c)), while common-template and length controls distinguish composition from presentation. Successful probes within harm or double-wrong cases establish local recoverability, not unique blame, a deployable selector, or the absence of useful advice in unrescued cases. Fixed reductions across all questions capture both rescues and new losses; partial-update controls distinguish localized edits from full rollback and unchanged inputs. Section 4 examines growth outcomes, and Section 5 uses these controls to narrow their explanations.

4 EXPERIENCE GROWTH: DYNAMICS AND LIMITATIONS

In this section, we trace experience accumulation (Q1), examine whether earlier successes are preserved (Q2), and investigate usable experience within failed cases (Q3).

Experimental settings. Our provenance cohort comprises 24 curves: two readers, four tasks, and three methods. Each evaluates successive banks on 300 fixed questions after up to 150 construction examples with seed-42 splits; decoding variability remains part of the observed changes.

Benchmarks. We study four task domains: multi-hop question answering with HotpotQA (Yang et al., 2018), multi-turn tool use with BFCL (Patil et al., 2025), mathematical reasoning with Omni-MATH (Gao et al., 2025), and multidisciplinary knowledge reasoning with SuperGPQA (Du et al., 2025). Source-replacement and frozen-context interventions form separate cohorts; final outputs identical to checkpoint 150 are not additional stages.

Metrics. We report answer token F1 for HotpotQA construction curves and task correctness for BFCL and SuperGPQA. Omni-MATH uses a diagnostic answer scorer rather than official benchmark scoring; later HotpotQA interventions use binary answer correctness. Paired help/harm outcomes and checkpoint transitions distinguish new gains from lost successes, with score-change magnitudes retained for fractional F1. Appendix A reports complete checkpoint help/harm rates; Appendices B and C provide evaluation details and additional numerical results.

4.1 Q1: DOES CONTINUED CONSTRUCTION YIELD STABLE PERFORMANCE GAINS?

Following the framework in Section 3, we evaluate utility across construction checkpoints and compare experience sources to examine how gains develop.

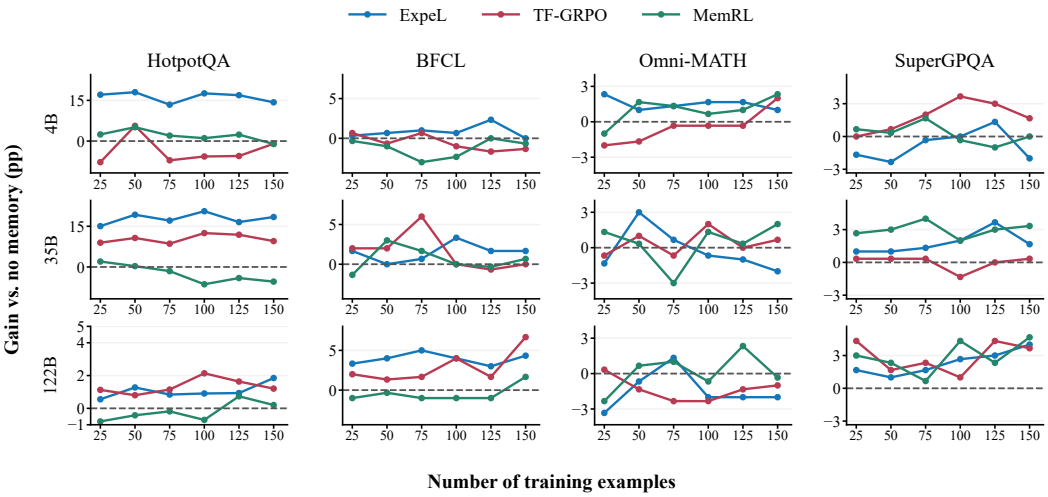


Figure 3: Task performance under different memory strategies as training examples increase. Gains are relative to each run's no-memory score (pp).

Performance along construction. As shown in Figure 3, progress is uneven across tasks and methods. Some curves continue improving late in construction, while others peak earlier and then lose ground; even the separate 122B reference exhibits late reversals on Omni-MATH. Help and harm coexist throughout this process (Table 1(a)). The value of further construction therefore depends on how newly acquired gains balance emerging losses.

Changing the experience source. One possible source of this uneven progress is the model's ability to construct useful experience. Table 1(b) compares self-evolution with 122B editing or full 122B construction, each relative to its own no-memory baseline. The results show that stronger editing improves average HotpotQA gains, but replacing the trajectory builder as well does not improve them further. Across tasks, stronger construction benefits 4B less consistently than 35B. The benefit of stronger experience sources varies with the receiving model and task. These historical cohorts establish a transfer pattern; paired construction controls would be needed to isolate its cause.

Table 1: Provenance effects and cross-model experience transfer. Darker blue indicates larger values within each column in (a) and within each row in (b). In (a), Δ is Help minus Harm, not an accuracy gain for fractional HotpotQA F1. In (b), we use Qwen-3.5-122B as the editor and builder.

(a) Six-checkpoint means (%).

Reader	Help	Harm	Δ
Four tasks			
4B	11.75	11.34	+0.41
35B	12.03	9.53	+2.50
122B	5.71	4.58	+1.13
Reasoned SuperGPQA			
4B	10.13	11.69	-1.56
35B	8.22	5.43	+2.80
122B	5.98	3.28	+2.70

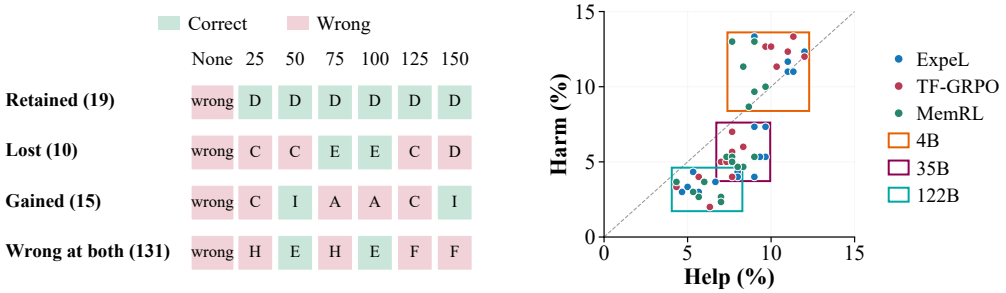
(b) Final gain, averaged over methods (pp).

Reader	Task	Self	Editor	Builder
4B	HotpotQA	-7.62	+5.98	+0.77
	BFCL	-1.89	+0.00	-4.44
	Omni-MATH	-2.44	-0.67	-0.11
	SuperGPQA	+2.00	+0.78	-0.33
35B	HotpotQA	+6.74	+11.47	+9.54
	BFCL	-1.89	+2.00	-1.33
	Omni-MATH	+3.44	+0.89	+5.11
	SuperGPQA	+1.56	+0.33	+0.56

4.2 Q2: Are earlier successes preserved as memory grows?

We next track previously solved questions across checkpoints, using the paired-outcome analysis in Section 3 to examine whether earlier benefits survive.

Success turnover. Among 4B TF-GRPO baseline errors on reasoned SuperGPQA, some cp75 rescues are lost by cp150 while other questions become correct (Figure 4(a)). A nearly unchanged score can thus accompany substantial turnover in which questions are solved. In the illustrative derivative case, the model retains a correct expression but selects the wrong option, locating the observed failure at the final answer choice. This fragility motivates examining preservation alongside aggregate progress.



(a) Answer turnover: 4B TF-GRPO

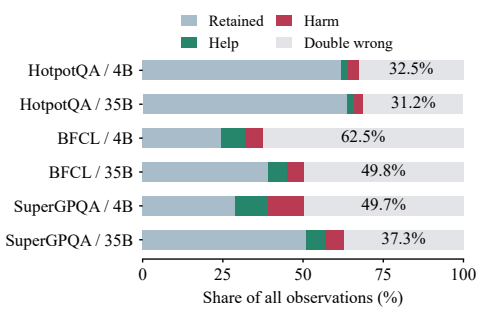
(b) Help–harm across model scales

Figure 4: Reasoned SuperGPQA. (a) Option timelines, colored by correctness; (b) checkpoint rates over 300 questions; colored outlines bound each model's observations, not confidence regions.

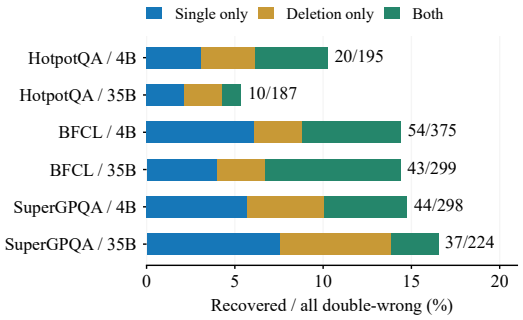
Preservation across model scales. To compare preservation at a common starting point, we measure conditional harm on reasoned SuperGPQA questions that all models initially solve. Across methods, the smaller model loses a larger share of these successes. This pattern agrees with the less favorable help–harm balance shown in Figure 4(b). Smaller-model runs are therefore less reliable at preserving initially shared successes. Since the banks are constructed independently, this gap reflects the combined construction-and-use loop.

4.3 Q3: Do some failures arise from ineffective use of retrieved experience?

We examine whether failed cases leave room for better use of available experience by isolating or removing entries from shared 4B-source MemRL contexts at cp75 and cp150. Figure 5(a) shows the full-context outcomes, including double-wrong cases that remain incorrect both without memory and with the supplied experience. Figure 5(b) focuses on this group: colored segments distinguish cases recovered only by single-entry isolation, only by entry deletion, or by both types of intervention. These recoveries reveal opportunities for better use of the existing bank: some persistent errors become correct after changing the exposed experience. Most remain unresolved under the tested probes. SuperGPQA probes only the first exposed entry, whereas HotpotQA and



(a) Full-context outcome composition



(b) Recoverability within double wrong

Figure 5: MemRL outcomes, pooling cp75 and cp150: (a) full-context outcome composition; (b) local recovery among double-wrong observations.

BFCL aggregate recoveries over tested entries; the different budgets limit cross-task comparisons. These post-hoc recoveries identify local opportunities whose practical value depends on selecting useful contexts without knowing the answer.

Self-directed memory updates do not reliably improve performance. The results reveal three connected patterns. As models build and update their own memory, performance does not improve consistently (Q1). New gains can come at the cost of answers that were previously correct (Q2). Some failed cases recover when the supplied experience is reduced or reorganized, suggesting that available experience is not always used effectively (Q3). These findings show that accumulating experience and benefiting from it are distinct challenges. This gap motivates further analysis of how readers use available experience: why the same advice affects readers differently, and how its usefulness changes when combined with other entries.

5 Understanding and Improving Reader-Side Experience Use

By holding the supplied experience fixed or changing its composition, we separate reader differences from context effects and trace how these effects alter reasoning and tool use.

5.1 How readers and surrounding context affect experience use

To isolate the reader's role in the lost successes observed in Section 4, we give both readers identical final contexts on selected cases from four tasks that can be solved by both readers without memory. We find that the identical final contexts produce more failures unique to 4B than to 35B (Table 2(a)) and the paired difference is statistically significant under bootstrap analysis. This suggests that the smaller reader is more likely to lose a previously correct answer when given retrieved experience.

Table 2: Reader-dependent harm under shared-context controls. In (b), accuracy is in percent; Gap is 4B minus 35B, and the last row is combined minus padded accuracy (pp).

(a) Shared-context outcomes.

4B	35B	Cases	%
Correct	Correct	346	68.24
Incorrect	Correct	**111**	**21.89**
Correct	Incorrect	22	4.34
Incorrect	Incorrect	28	5.52

(b) Common-template accuracy.

Context	4B	35B	Gap
Single	72.92	68.75	+4.17
Padded	67.71	61.98	+5.73
Combined	31.77	50.00	−18.23
Comb.–pad.	−35.94	−11.98	**−23.96**

To examine whether this effect comes from longer contexts or interference from other experience, we compare isolated advice, length-matched padding, and combined experience under a common template. On previously identified harm cases from HotpotQA and SuperGPQA, combined experience reduces accuracy more than length-matched padding, especially for 4B (Table 2(b)). This suggests that other experience can interfere with useful advice beyond simply making the context longer. Both readers recover some correct answers when given a single entry, showing that advice that helps alone can lose its benefit when combined with other experience.

Table 3: Representative cases where reducing experience context changes answer correctness.

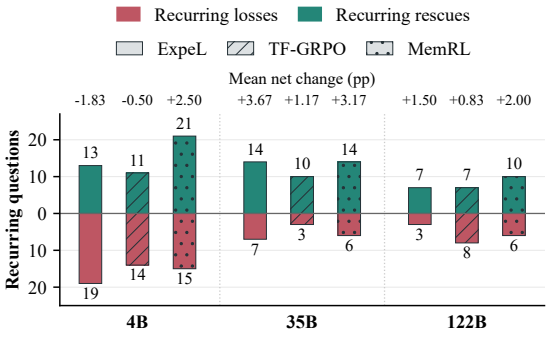
Task / reader	Before: task behavior	What we changed	After: task behavior
HotpotQA 4B, MemRL Special economic zone	**Wrong relation:** treats a zone *near* Kadaperi as *inside* it, answering Kadaperi rather than the economic zone asked for.	Remove entry 98, a venue-to-county lookup strategy.	**Wrong → correct:** describes Kadaperi as nearby and answers Madras Export Processing Zone.
SuperGPQA 4B, TF-GRPO Pendulums	**Correct calculation:** obtains the net-force factor $\sqrt{2} + 1/2$ and selects C.	Remove the first exposed experience unit.	**Correct → wrong:** distance/unit errors appear and the answer changes to B.
BFCL 35B, ExpeL Ticket creation	**Wrong tool argument:** adds booking details to the description `Request to cancel.` and fails the state check.	Replace the full context with one file-system trajectory.	**Wrong → correct:** preserves the exact description without additions, and the ticket passes the state check.

5.2 How experience changes reasoning and tool use

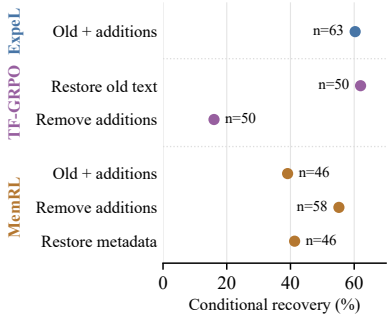
Following the analysis in Section 5.1, we inspect paired reasoning and tool-use traces to understand how context changes lead to correct or incorrect answers. As shown in Table 3, reducing context corrects a mistaken spatial relation in HotpotQA and prevents unwanted additions to a tool argument in BFCL, but disrupts a correct calculation in SuperGPQA. These cases suggest that experience can redirect how the reader interprets evidence and carries out the task. Removing context can prevent a mistaken inference or unwanted additions to a tool argument, but can also remove guidance that supports a correct calculation. The effect therefore depends on which guidance is retained, rather than simply how much context is supplied (Appendix E).

5.3 The gains and losses of reducing or revising memory

The case analysis motivates testing context reduction beyond individual examples. We evaluate whether it can recover failed answers while preserving previously correct ones across the full test set. As shown in Figure 6(a), repeated final-checkpoint deletion tests across three readers produce recurring losses even when the average score improves. Simply removing experience can therefore recover some answers at the cost of previously correct ones, including for the largest reader.



(a) Fixed deletion: gains and losses



(b) Distinct partial updates

Figure 6: Recovery does not certify update safety. (a) Recurring rescues/losses and mean net change on SuperGPQA; net change is not the difference between recurring counts. (b) Conditional recovery on degraded cases; differing case sets preclude direct method rankings.

Exploring more targeted changes. These losses motivate examining whether changing specific parts of a memory update can recover failed answers. We compare interventions on added entries, revised text, and retrieval metadata, checking that each produces a genuine partial change rather than leaving the input unchanged or restoring the entire old bank (Appendix C). As shown in Figure 6(b), ExpeL text restoration largely recovers answers through full rollback, while partial changes to MemRL entries and metadata and TF-GRPO text also recover some degraded cases. These results identify several possible targets for repair; their overall benefit depends on whether they also preserve answers that remain correct. See full results in Appendix D.

5.4 Exploring improvements that preserve useful experience

Building on this analysis, we introduce three lightweight experience-level variants that aim to improve performance by reducing interference or making experience easier to apply. *(1) Top-1 Memory* tests whether retaining only the highest-ranked entry can reduce interference while preserving the most relevant guidance. Alternatively, we retain the retrieved entries and refine their presentation. *(2) Compact Examples* expresses experience as short, actionable demonstrations, testing whether more concrete guidance is easier to follow. *(3) Scope Protection* addresses a different difficulty: advice useful for its source task may mislead the reader when applied to a new task. It separates source-task information from current-task evidence and removes explicit source-answer cues, aiming to preserve reusable guidance without carrying over task-specific details. We evaluate these variants on four benchmarks, comparing their final performance with native MemRL and measuring help and harm against a shared no-memory reference.

Table 4: 4B accuracy (%) per benchmark. Help/Harm counts use a shared no-memory baseline.

Method	**HotpotQA**	**BFCL**	**Omni-MATH**	**SuperGPQA**	*Avg.*	*Help / Harm*	
MemRL	62.3	**34.7**	39.0	40.7	44.2	98 /	95
+ *Top-1 Memory*	**65.0**	33.7	37.3	40.7	44.2	92 /	**89**
+ *Compact Examples*	63.3	30.0	36.0	38.7	42.0	84 /	107
+ *Scope Protection*	64.7	**34.7**	**39.3**	**43.7**	**45.6**	**109 /**	**89**

As shown in Table 4, Scope Protection achieves higher average accuracy than native MemRL. We suspect that reducing the number of entries can remove useful guidance, while making advice more concrete still leaves open when it should apply. Scope Protection instead helps distinguish reusable guidance from source-task details, making the retained experience easier to apply appropriately. Its same-batch gains come from additional corrections outweighing new errors, outweighing new errors, suggesting that clearer applicability boundaries can improve overall performance. Appendix F details the comparison cohorts and additional update-control experiments.

6 Discussion and Future Directions

Our results suggest that learning from textual experience requires both new gains and preserved successes. Stronger experience sources help in some settings, but their value depends on the reader and surrounding context. Memory updates should therefore be evaluated on both outcomes, with scope protection illustrating how this perspective can guide improvements. Two directions remain: (1) **Across models and tasks.** Our experiments cover one model family, three memory strategies, and four task domains, with repeated construction runs covering part of this setting. Extending the analysis to other model families and longer experience streams would reveal which patterns persist and where they change. (2) **From behavior to mechanism.** Our controls establish differences in how readers respond to the same experience and how their answers change with the surrounding context. A next step is to distinguish errors in the advice itself, application beyond its intended scope, and mistakes in carrying it out. Targeted interventions on these factors could guide more precise memory updates and reader-side improvements.

7 Conclusion

We analyze textual experience-based self-evolution through checkpoint-level dynamics and interventions that separate experience construction from reader-side use. Across the studied settings, accumulating experience does not ensure sustained improvement: newly solved tasks coexist with regressions, and previously acquired gains can be lost during subsequent updates. Stronger builders alleviate some failures without guaranteeing successful transfer to weaker readers. Matched-input and context-intervention results further show that experience utility depends on both the reader and the surrounding advice, supporting reader–experience alignment as an important limitation of this self-evolution paradigm. Our exploratory mitigation results suggest that explicitly preserving applicability boundaries is a promising direction, while reliable harm control remains unresolved. Together, these findings motivate evaluating experience systems by their ability to acquire useful guidance and preserve successful behavior throughout the evolution process.

REPRODUCIBILITY STATEMENT

The supplementary material provides code for memory construction, evaluation, and controlled-context interventions, together with task-specific prompts and scoring implementations. It also includes compact result tables, machine-readable checkpoint scores, selected split metadata, and checkpoint-bank hashes. The accompanying documentation describes setup requirements, protocol differences, and the scope of reproduction.

AI USE STATEMENT

AI tools were used to assist with implementation, manuscript preparation, and visual presentation:

- **Code implementation.** External AI coding tools assisted with implementing the experimental and analysis code. AI-assisted code was subsequently reviewed by the authors and checked using tests and validation scripts.
- **Language editing.** AI tools assisted with proofreading grammar and refining wording for clarity. The authors reviewed and edited the resulting text.
- **Visual presentation.** Some illustrative assets in Figures 1 and 2 were AI-generated and then manually edited and assembled to improve clarity and presentation. These illustrations are distinct from experimental results.

The authors take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Yuzheng Cai, Siqi Cai, Yuchen Shi, Zihan Xu, Lichao Chen, Yulei Qin, Xiaoyu Tan, Gang Li, Zongyi Li, Haojia Lin, Yong Mao, Ke Li, and Xing Sun. Training-Free Group Relative Policy Optimization. arXiv preprint arXiv:2510.08191, 2025.

Yurui Chang, Yiran Wu, Qingyun Wu, and Lu Lin. MemCollab: Cross-Model Memory Collaboration via Contrastive Trajectory Distillation. arXiv preprint arXiv:2603.23234, 2026.

Ding Chen, Simin Niu, Kehang Li, Peng Liu, Xiangping Zheng, Bo Tang, Xinchi Li, Feiyu Xiong, and Zhiyu Li. HaluMem: Evaluating Hallucinations in Memory Systems of Agents. In *Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing*, 2026. Accepted, to appear.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building Production-Ready AI Agents with Scalable Long-Term Memory. In *ECAI 2025: 28th European Conference on Artificial Intelligence*, volume 413 of *Frontiers in Artificial Intelligence and Applications*, pp. 2993–3000. IOS Press, 2025. doi: 10.3233/FAIA251160.

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.

Xeron Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, King Zhu, Minghao Liu, Yiming Liang, Xiaolong Jin, Zhenlin Wei, Chujie Zheng, Kaixin Deng, Shuyue Guo, Shian Jia, Sichao Jiang, Yiyan Liao, Rui Li, Qinrui Li, Sirun Li, Yizhi Li, Yunwen Li, Dehua Ma, Yuansheng Ni, Haoran Que, Qiyao Wang, Zhoufutu Wen, Siwei Wu, Tianshun Xing, Ming Xu, Zhenzhu Yang, Noah Wang, Junting Zhou, Yuelin Bai, Xingyuan Bu, Chenglin Cai, Liang Chen, Yifan Chen, Chengtuo Cheng, Tianhao Cheng, Keyi Ding, Siming Huang, Yun Huang, Yaoru Li, Yizhe Li, Zhaoqun Li, Tianhao Liang, Chengdong Lin, Hongquan Lin, Yinghao Ma, Zhongyuan Peng, Zifan Peng, Qige Qi, Shi Qiu, Xingwei Qu, Shanghaoran Quan, Yizhou Tan, Zili Wang, Chenqing Wang, Hao Wang, Yiya Wang, Yubo Wang, Jiajun Xu, Kexin Yang, Ruibin Yuan, Yuanhao Yue, Tianyang Zhan, Chun Zhang, Jinyang Zhang, Xiyue Zhang, Owen Zhang, Yue Zhang, Yongchi Zhao, Xiangyu Zheng, Chenghua Zhong, Yang Gao, Zhoujun Li, Dayiheng Liu, Qian Liu, Tianyu Liu, Shiwen Ni, Junran Peng, Yujia Qin, Wenbo Su, Guoyin Wang, Shi Wang, Jian Yang, Min

Yang, Meng Cao, Xiang Yue, Zhaoxiang Zhang, Wangchunshu Zhou, Jiaheng Liu, Qunshu Lin, Wenhao Huang, and Ge Zhang. SuperGPQA: Scaling LLM Evaluation across 285 Graduate Disciplines. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, pp. 125016–125089. Curran Associates, Inc., 2025. doi: 10.52202/085713-3766.

Tao Feng, Chongrui Ye, Tianyang Luo, Jingjun Xu, Xueqiang Xu, Haozhen Zhang, Zhigang Hua, Yan Xie, Shuang Yang, Ge Liu, and Jiaxuan You. ExpGraph: Model-Agnostic Experience Learning with Graph-Structured Memory for LLM Agents. arXiv preprint arXiv:2605.30712, 2026.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, Zhengyang Tang, Benyou Wang, Daoguang Zan, Shanghaoran Quan, Ge Zhang, Lei Sha, Yichang Zhang, Xuancheng Ren, Tianyu Liu, and Baobao Chang. Omni-MATH: A Universal Olympiad Level Mathematic Benchmark for Large Language Models. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, pp. 100540–100569, 2025.

Ishir Garg, Neel Kolhe, Dawn Song, and Xuandong Zhao. MemFail: Stress-Testing Failure Modes of LLM Memory Systems. In *Findings of the Association for Computational Linguistics: EMNLP 2026*, 2026. Accepted, to appear.

Junwei Liao, Haoting Shi, Ruiwen Zhou, Jiaqian Wang, Shengtao Zhang, Wei Zhang, Ying Wen, Zhiyu Li, Feiyu Xiong, Bo Tang, Weinan Zhang, and Muning Wen. MemQ: Integrating Q-learning into self-evolving memory agents over provenance DAGs, 2026.

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. doi: 10.1162/tacl_a_00638.

Xutao Mao, Jinman Zhao, Gerald Penn, and Cong Wang. What Happens Inside Agent Memory? Circuit Analysis from Emergence to Diagnosis. arXiv preprint arXiv:2605.03354, 2026.

Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. ReasoningBank: Scaling Agent Self-Evolving with Reasoning Memory. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, pp. 94327–94354, 2026.

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.

Shishir G. Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The Berkeley Function Calling Leaderboard (BFCL): From Tool Use to Agentic Evaluation of Large Language Models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 48371–48392. PMLR, 2025.

Sidharth Pulipaka, Oliver Chen, Manas Sharma, Taaha Saleem Bajwa, Vyas Raina, and Ivaxi Sheth. PersistBench: When Should Long-Term Memories Be Forgotten by LLMs? In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Chris Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, Wei Li, Folawiyo Campbell-Ajala, Daniel Toyama, Robert Berry, Divya Tyamagundlu, Timothy Lillicrap, and Oriana Riva. AndroidWorld: A dynamic benchmarking environment for autonomous agents. In *The Thirteenth International Conference on Learning Representations*, pp. 406–441, 2025.

Zhanyu Shen, Sijie Cheng, Zhicheng Guo, Weiqin Wang, Yile Wang, and Hui Huang. AnchorMem: Anchored Facts with Associative Contexts for Building Memory in Large Language Models. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the*

Association for Computational Linguistics: ACL 2026, pp. 34784–34798, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.1736.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652. Curran Associates, Inc., 2023. doi: 10.52202/075280-0377.

Yian Wang, Agam Goyal, Yuen Chen, and Hari Sundaram. State Contamination in Memory-Augmented LLM Agents. arXiv preprint arXiv:2605.16746, 2026.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-Mem: Agentic Memory for LLM Agents. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, pp. 17577–17604. Curran Associates, Inc., 2025. doi: 10.52202/085713-0593.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. Making Retrieval-Augmented Language Models Robust to Irrelevant Context. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, pp. 29862–29883, 2024.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, Urmish Thakker, James Y Zou, and Kunle Olukotun. Agentic Context Engineering: Evolving Contexts for Self-Improving Language Models. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, pp. 86069–86100, 2026a.

Shengtao Zhang, Jiaqian Wang, Ruiwen Zhou, Junwei Liao, Yuchen Feng, Zhuo Li, Yujie Zheng, Weinan Zhang, Ying Wen, Zhiyu Li, Feiyu Xiong, Yutao Qi, Bo Tang, and Muning Wen. MemRL: Self-Evolving Agents via Runtime Reinforcement Learning on Episodic Memory. arXiv preprint arXiv:2601.03192, 2026b.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM Agents Are Experiential Learners. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17):19632–19642, 2024. doi: 10.1609/aaai.v38i17.29936.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. MemoryBank: Enhancing Large Language Models with Long-Term Memory. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17):19724–19731, 2024. doi: 10.1609/aaai.v38i17.29946.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations*, pp. 15585–15606, 2024.

Sibo Zhu, Wenyi Wu, Kun Zhou, Stephen Wang, and Biwei Huang. Hybrid Self-evolving Structured Memory for Computer-Use Agents. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 11287–11304, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.549.

A Supplementary Evidence and Full Rates

A.1 Complete checkpoint help and harm rates

Table 5 expands the checkpoint means in the main text. For legacy HotpotQA, help/harm denotes an increase/decrease in token F1; elsewhere it denotes a change in binary correctness.

Table 5: Complete help and harm rates (%). Four-task rows average 12 task–method conditions; reasoned SuperGPQA rows average three methods. Avg. averages all six checkpoints. The 122B reference is a separate cohort.

Reader	Help (%)							Harm (%)						
	25	50	75	100	125	150	Avg.	25	50	75	100	125	150	Avg.
A. Four-task provenance cohort														
4B	10.75	12.50	11.64	12.28	11.83	11.50	11.75	10.61	12.64	11.11	11.19	11.17	11.33	11.34
35B	11.19	12.14	12.14	12.28	12.44	12.00	12.03	8.78	9.19	10.22	9.78	9.69	9.50	9.53
122B	5.28	5.44	5.39	5.61	6.00	6.56	5.71	4.58	4.61	4.39	4.58	4.69	4.61	4.58
B. Reasoned SuperGPQA														
4B	10.56	10.78	10.11	9.56	10.22	9.56	10.13	11.11	10.89	11.22	12.44	11.78	12.67	11.69
35B	8.22	8.33	8.22	8.44	7.67	8.44	8.22	5.67	4.78	6.56	5.22	4.89	5.44	5.43
122B	5.56	5.44	5.11	5.67	6.56	7.56	5.98	2.56	3.78	3.56	3.00	3.33	3.44	3.28

A.2 How the supplementary evidence supports the main text

Appendix B explains what each comparison measures and how it is evaluated. Appendix C reports checkpoint scores, source-replacement results, and controlled-context comparisons. Appendix D examines which failures recover and whether fixed context changes also introduce errors; Appendix E follows these changes in the reader's behavior. Appendix F documents the scope-protection comparison and complete 150/300 memory-admission tests. Each experiment is reported separately because its samples and controls answer a different question.

B Detailed Diagnostic Protocol

This appendix details the measures and controls introduced in Section 3.

B.1 State, roles, and comparison cohorts

As in Section 3, M_t contains experience text and retrieval metadata, b generates trajectories, e updates memory using feedback f_t, and r answers evaluation questions. The supplied context $C_r(x, M_t)$ is the experience text remaining after selection, ordering, formatting, and truncation; builders can use the current bank through the same steps. Evaluation freezes both the bank and model parameters, while generation can still vary. Editor replacement assigns designated editing roles to 122B while retaining the original trajectory generator and reader. Sharing a 122B-built bank fixes the stored experience; the shared-final-context control additionally checks that readers receive identical task and memory inputs, excluding the model identifier. This separates differences in supplied inputs from reader responses, while leaving advice correctness and internal causes open.

The common budget is 150 construction tasks and 300 evaluation tasks, with checkpoints at 25, 50, 75, 100, 125, and 150. Historical self-evolution, provenance runs, source replacement, and focused interventions remain separate cohorts; the 122B reference is not a matched third-reader replication of the two-reader provenance runs. Legacy HotpotQA curves use fractional token F1, whereas later answer-prefix interventions use recorded binary correctness. Answer-only and brief-explanation SuperGPQA protocols are also kept separate. Final results identical to cp150 do not count as an additional checkpoint or repeat.

B.2 SCORE CHANGES AND CORRECTNESS TRANSITIONS

For a fixed reader and matched question i, let $s_i(M) \in [0,1]$ be its score and $z_i(M) \in \{0,1\}$ its protocol-defined correctness. For reference A and candidate B, write $d_i = s_i(B) - s_i(A)$:

$$G(A,B) = \frac{1}{N}\sum_i d_i = \frac{1}{N}\sum_i \max(d_i, 0) - \frac{1}{N}\sum_i \max(-d_i, 0). \tag{4}$$

This identity applies to fractional F1 as well as binary scores. For binary scores, help and harm count $0 \to 1$ and $1 \to 0$ transitions, so their rate difference equals the accuracy change. For fractional F1, score changes also differ in size; the frequency of increases and decreases alone cannot recover the mean gain.

Comparison with no memory measures the bank's overall effect; comparison with the preceding checkpoint measures change between versions, including decoding variation. With $\bar{s}(M) = N^{-1}\sum_i s_i(M)$, the descriptive best-to-final gap is $\max_{t\in\{25,50,75,100,125,150\}} \bar{s}(M_t) - \bar{s}(M_{150})$. This gap describes how much performance falls from its observed peak; using it for early stopping would require access to evaluation outcomes.

Let $\mathcal{S}_t, \mathcal{H}_t, \mathcal{G}_t, \mathcal{W}_t$ denote questions with correctness pairs $(1,1), (1,0), (0,1), (0,0)$, respectively, where the first component is correctness without memory and the second is correctness at checkpoint t. They partition the evaluation set; $\mathcal{W}_t$ means double wrong at checkpoint t, not failure at every checkpoint. Conditional rescue is $|\mathcal{G}_t|/(|\mathcal{G}_t| + |\mathcal{W}_t|)$ and conditional break is $|\mathcal{H}_t|/(|\mathcal{S}_t| + |\mathcal{H}_t|)$ when the denominators are nonzero. For successful-question sets $\mathcal{K}_a$ and $\mathcal{K}_b$, retained successes are $\mathcal{K}_a \cap \mathcal{K}_b$, losses are $\mathcal{K}_a \setminus \mathcal{K}_b$, and gains are $\mathcal{K}_b \setminus \mathcal{K}_a$. These sets reveal answer changes hidden by a stable total; identifying the responsible edits requires further tests.

B.3 LOCAL RECOVERY AND INTERVENTION FIDELITY

Let $\mathcal{P}_t$ be the questions included in a local-probe panel and $\mathcal{V}_i$ the tested single-entry and leave-one-entry-out contexts for question i. Write $z_i(c)$ for correctness when question i is answered with supplied context c. The double-wrong recovery fraction is

$$R_{\text{local}}(t) = \frac{|\{i \in \mathcal{W}_t \cap \mathcal{P}_t : \exists c \in \mathcal{V}_i,\ z_i(c) = 1\}|}{|\mathcal{W}_t \cap \mathcal{P}_t|}. \tag{5}$$

For a full-question evaluation, the denominator includes all double-wrong questions; selected panels use only their eligible questions. Each question counts once if any tested context restores correctness, including when both isolation and deletion succeed. This measures recovery found through testing several contexts. It does not measure how often a practical selector would choose the successful one, and untested contexts may offer further recoveries.

TF-GRPO exposes the global pool; ExpeL and MemRL select content before formatting. Probes preserve native units: an ExpeL unit may be a rule block or a successful trajectory, not a single abstract rule. Deletion does not retrieve replacements, but may change length, position, or numbering. For the selected forced-memory panel, candidates come from frozen banks rather than advice written using test answers; discovery selects candidates and a separate seed confirms them. Here, a behavioral oracle is an entry that succeeds in the tested setting; the term does not certify its factual correctness. Common-template isolated, padded, and combined replays control the wrapper and approximate token count; punctuation padding is not guaranteed to be semantically inert.

Update counterfactuals restore text, additions, deletions, shared metadata, or retrieval settings. We compare actual initial inputs with the old and new versions, distinguishing unchanged new inputs, restored old inputs, identical old/new inputs, and genuinely partial changes. This distinguishes recovery after a partial edit from recovery due to replay variation or complete rollback. ExpeL matching is pool-level, and MemRL metadata restoration is not an isolated Q-value intervention. Because operations can interact, their recovery counts cannot be added to assign each operation a share of the failures.

B.4 SELECTION, UNCERTAINTY, AND TRACEABILITY

Checkpoint evaluations cover the full question set. Panels selected for earlier harm and their matched controls describe those selected cases. The update panel selects the largest historical

deterioration per reader–task–method curve and requires old-correct/new-wrong to recur before measuring recovery. Case studies inspect complete traces but do not estimate the prevalence of their proposed mechanisms.

The strict paired-reader analysis bootstraps question IDs within datasets, retaining observations from different methods in the same question cluster (2,000 resamples). Appendix D gives the bootstrap settings for the fixed-probe and repeated-deletion analyses. These intervals describe uncertainty within the selected panels; they do not account for all variation from decoding or candidate search. Checkpoint identities, source-bank hashes, entry exposures, final contexts, predictions, and API requests are retained in the experiment records. Identical-input reuse is reported explicitly; it does not add independent evidence.

C Checkpoint Summaries and Controlled-Context Results

The checkpoint tables summarize gains at cp75, cp150, and across all six checkpoints. Gains subtract the no-memory score from the same cohort and checkpoint before averaging; this matters when repeated baseline generations differ. Legacy HotpotQA uses token F1; other scores use protocol-defined correctness, including diagnostic rather than official Omni-MATH scoring. The numerical archive, `appendix_full_tables.json`, preserves the full checkpoint scores, baselines, and subgroup results. The compact gains are calculated from those archived two-decimal scores and are descriptive to that precision. Final results identical to cp150 count once.

C.1 Growth and source replacement

Table 6: Source replacement: mean six-checkpoint gain over each cohort's own no-memory scores (pp). Self = historical self-evolution; Edit = 122B editor; Build = frozen 122B builder.

		ExpeL			TF-GRPO			MemRL		
Reader	Task	Self	Edit	Build	Self	Edit	Build	Self	Edit	Build
35B	BFCL	+1.56	+3.55	+1.06	-0.72	+0.95	-0.94	-1.61	-0.44	+0.61
	HotpotQA	+19.70	+20.37	+15.45	+5.72	+16.30	+11.94	-5.52	-1.89	-0.84
	Omni-MATH	+2.56	+1.17	+2.00	+3.73	+0.89	+3.39	+2.84	+1.16	+1.05
	SuperGPQA	-0.33	-1.00	+0.89	+1.06	+0.33	+0.72	+1.45	+0.99	+0.33
4B	BFCL	+0.27	+1.95	+0.34	+1.05	+1.89	-7.22	-2.45	+0.00	-0.28
	HotpotQA	+16.35	+13.86	+6.64	-32.86	+9.73	+6.99	+1.35	+3.34	+3.76
	Omni-MATH	-2.84	+0.39	-0.22	-4.50	+0.00	+0.00	-3.12	-1.27	-0.67
	SuperGPQA	-0.00	+0.16	+0.39	+4.33	+0.27	+1.45	-0.22	+1.89	+0.28

Incomplete historical cells. BFCL 4B ExpeL cp50/75/100 in frozen-builder transfer contain 299 scored examples out of 300 planned; the stored scores are preserved, not imputed. The cp50 wrapper exited 143 after writing the retained structured result; the later dataset completion is recorded in the archive. These separate-source cohorts are descriptive comparisons, not paired causal effects.

C.2 Separate construction repetition

We repeat construction on HotpotQA and BFCL to check whether the observed dynamics depend on a particular run. Both readers use the same ordered training and evaluation samples as the provenance reference, and record-level checks confirm coverage, bank hashes, and agreement with the reported scores. The API generation seed was unset and TF-GRPO rollout temperature was 0.7; this repeats the protocol rather than isolating a controlled seed effect. Other historical content and code hashes were not all fixed.

Table 9 shows that both final gains and losses after the best checkpoint vary across runs. The repeated 4B TF-GRPO run on HotpotQA declines much more sharply, while other conditions change less. Continued construction can therefore be unstable in degree as well as direction. We report this repeat separately from the prespecified provenance cohort.

Table 7: Provenance curves and separate 122B references. Gains in pp; Avg. covers six checkpoints. SuperGPQA is answer-only for 4B/35B and brief-explanation for 122B.

		ExpeL			TF-GRPO			MemRL		
Reader	Task	75	150	Avg.	75	150	Avg.	75	150	Avg.
122B	BFCL	+5.00	+4.34	+3.95	+1.67	+6.67	+2.89	-1.00	+1.67	-0.44
	HotpotQA	+0.84	+1.85	+1.06	+1.15	+1.21	+1.34	-0.18	+0.20	-0.19
	Omni-MATH	+1.33	-2.00	-1.45	-2.33	-1.00	-1.33	+1.00	-0.33	+0.11
	SuperGPQA	+1.66	+4.00	+2.33	+2.33	+3.66	+2.88	+0.66	+4.66	+2.88
35B	BFCL	+0.66	+1.66	+1.49	+6.00	+0.00	+1.55	+1.66	+0.66	+0.61
	HotpotQA	+17.12	+18.42	+17.83	+8.59	+9.54	+10.36	-1.52	-5.39	-2.50
	Omni-MATH	+0.67	-2.00	-0.22	-0.67	+0.67	+0.39	-3.00	+2.00	+0.39
	SuperGPQA	+1.34	+1.67	+1.78	+0.34	+0.34	+0.01	+4.00	+3.34	+3.00
4B	BFCL	+1.00	+0.00	+0.83	+0.66	-1.34	-0.56	-3.00	-0.67	-1.23
	HotpotQA	+13.42	+14.28	+16.21	-7.10	-1.00	-3.58	+2.01	-1.08	+1.98
	Omni-MATH	+1.34	+1.00	+1.50	-0.33	+2.00	-0.44	+1.34	+2.34	+1.00
	SuperGPQA	-0.33	-2.00	-0.83	+2.00	+1.67	+1.84	+1.67	+0.00	+0.23

Table 8: Brief-explanation SuperGPQA, independently built banks. Gain over the cohort's corresponding no-memory score (pp); Avg. averages six checkpoints.

		ExpeL			TF-GRPO			MemRL		
Reader	Task	75	150	Avg.	75	150	Avg.	75	150	Avg.
35B	SuperGPQA	+2.33	+4.00	+3.50	+0.66	+2.33	+1.94	+2.00	+2.66	+2.94
4B	SuperGPQA	+0.00	-4.34	-0.78	-3.00	-2.00	-1.67	-0.34	-3.00	-2.23

Table 9: Run variability: final gain over each run's no-memory baseline and best-to-final gap (pp).

Reader	Task	Method	Ref. gain	Ref. gap	Repeat gain	Repeat gap
4B	HotpotQA	ExpeL	+14.29	3.72	+18.09	0.61
	HotpotQA	TF-GRPO	-0.99	6.61	-36.51	26.43
	HotpotQA	MemRL	-1.08	6.19	-1.22	2.36
	BFCL	ExpeL	+0.00	2.33	+0.00	2.33
	BFCL	TF-GRPO	-1.33	2.00	+0.00	2.33
	BFCL	MemRL	-0.67	0.67	+0.33	2.67
35B	HotpotQA	ExpeL	+18.42	2.13	+19.22	0.13
	HotpotQA	TF-GRPO	+9.54	2.95	+12.09	0.52
	HotpotQA	MemRL	-5.39	7.40	-7.21	5.22
	BFCL	ExpeL	+1.67	1.67	+2.33	1.67
	BFCL	TF-GRPO	+0.00	6.00	-1.67	2.00
	BFCL	MemRL	+0.67	2.33	+1.00	1.67

C.3 Common-template controls, including stable cases

The common-template comparison asks whether the combined-context penalty extends beyond previously harmed questions. We include stable controls that both readers previously answered correctly with and without memory, fixing candidates and panels before confirmation. Repeats are averaged within each case, and intervals bootstrap question clusters. Stable controls remain in the panel even if a replay becomes incorrect, avoiding selection by the new outcomes.

The combined-context penalty is stronger on the historical-harm panel and is not consistent across stable controls (Tables 10 and 11). Some stable conditions remain unchanged or improve, while HotpotQA ExpeL still loses accuracy for 4B. Surrounding experience therefore affects usability differently across cases and methods; adding context is not uniformly harmful.

Table 12: Distinct partial-update recoveries / exposures. – means no distinct partial exposure, not zero recovery.

Operation	ExpeL	TF-GRPO	MemRL
Old + additions	38/63	31/50	18/46
Restore text	–	31/50	–
Remove additions	18/63	8/50	32/58
Restore deleted	35/63	–	–

Table 10: All five conditions on both selected panels (accuracy, percent).

Panel	Reader	Cases	None	Natural	Single	Padded	Combined
Historical harm	4B	96	80.73	19.79	72.92	67.71	31.77
	35B	96	75.52	47.92	68.75	61.98	50.00
Stable control	4B	48	100.00	95.83	97.92	100.00	94.79
	35B	48	97.92	100.00	93.75	93.75	98.96

Table 11: Combined minus padded accuracy (pp). Each task–method cell has 16 historical-harm and eight stable-control cases per reader; zero denotes equal accuracy, not a fixed reference.

		Historical harm		Stable	
Task	Method	4B	35B	4B	35B
HotpotQA	ExpeL	-40.62	+18.75	-31.25	+0.00
	TF-GRPO	-43.75	-3.12	+0.00	+12.50
	MemRL	-40.62	-12.50	+0.00	+6.25
SuperGPQA	ExpeL	-50.00	-28.12	+0.00	+0.00
	TF-GRPO	-34.38	-28.12	+0.00	+0.00
	MemRL	-6.25	-18.75	+0.00	+12.50

C.4 Paired-reader diagnostic comparisons

We use matched inputs to distinguish differences between readers from differences in the experience they receive. The shared-context comparison includes 507 cases that both readers solve without memory, drawn from the selected four-task cohort. Under identical final contexts, failures unique to 4B are more frequent than those unique to 35B (Table 2a); the paired excess remains positive under question-cluster bootstrap resampling.

A separate candidate-confirmation panel tests whether advice that helps alone remains useful alongside other experience. The pooled contrast shows a larger combined-context penalty for 4B, while isolated candidates can rescue either reader when both fail with natural context. Together, these controls link the observed losses to both the receiving reader and the surrounding context. They describe selected diagnostic cases rather than failure rates across the full benchmarks.

C.5 Update-input identity audit

An apparent recovery after a memory edit may come from replay variation or complete rollback rather than the intended partial change. We therefore compare the actual supplied inputs with both checkpoint versions before interpreting outcomes. Tables 12 and 13 separate genuinely partial changes from input-equivalent controls. Some partial edits recover failed answers, but rollback and unchanged-input replays also produce recoveries; the nominal operation alone is insufficient to attribute a repair. Recovery counts overlap since one question can recover under several operations.

Table 13: Input-equivalence controls: recoveries / exposures pooled across operations. Repeated question exposures are not independent cases; – means no exposure.

Method	Full rollback	Unchanged new	Identical old/new
ExpeL	65/73	17/170	3/18
TF-GRPO	18/24	30/270	–
MemRL	50/58	30/374	4/6

D Cross-Method Recovery and Fixed-Context Controls

We extend the recovery analysis in two directions. BFCL tests whether failures also recover for ExpeL and TF-GRPO. Fixed SuperGPQA interventions then count both corrections and new errors across all questions, with repeated replays checking which effects recur.

D.1 BFCL beyond MemRL

We extend the recovery probes to ExpeL and TF-GRPO to check whether locally usable experience is specific to MemRL. Both readers receive the same frozen 4B-source contexts on all 300 BFCL questions at cp75 and cp150. Single-entry and leave-one-out probes cover questions that are not correct in both the no-memory and full-context conditions; each recovered case is counted once, regardless of how many probes succeed.

Table 14 shows recoveries under both methods, including cases that also fail without memory. Opportunities for better context use therefore extend beyond one memory representation. TF-GRPO tests more candidate contexts, so its larger recovery counts do not establish a better practical selection policy.

Table 14: BFCL local recovery: successful cases / eligible cases. Columns are post-hoc unions, not single-probe or policy accuracy.

		Harm recovered		Double wrong recovered	
Reader	Method	cp75	cp150	cp75	cp150
4B	ExpeL	9/19	9/19	22/180	23/180
	TF-GRPO	17/18	23/23	48/178	56/179
35B	ExpeL	15/25	15/25	15/146	16/147
	TF-GRPO	22/22	28/28	47/154	48/153

We retain each condition's recorded no-memory outcomes rather than forcing a shared baseline. Input-hash checks confirm that successful probes supplied changed, nonempty context. These full-question probes use one repeat, so decoding and candidate search can still affect recovery; the separate selected mechanism panel uses confirmation with another seed.

D.2 Reasoned SuperGPQA transfer baselines

The fixed-context probes require a baseline that separates reader use from new bank construction. We therefore evaluate 35B on frozen 4B-source contexts for three methods at cp75 and cp150, using the same 300 questions, brief-explanation prompt, and scorer. Table 15 shows that ExpeL loses net benefit between checkpoints while TF-GRPO gains it. These different trajectories motivate comparing each intervention with its corresponding full-context result.

Table 15: 35B reading frozen 4B-donor reasoned SuperGPQA contexts. No memory solves 170/300; H/L counts corrections/regressions relative to that baseline.

	cp75		cp150	
Method	Full correct	H/L	Full correct	H/L
ExpeL	172	17/15	161	16/25
MemRL	172	17/15	170	19/19
TF-GRPO	171	23/22	183	27/14

Checks confirm source/input hashes, coverage, and agreement with recorded metrics; no additional generations or independent rescoring were performed for this baseline check.

D.3 Fixed first-unit probes on reasoned SuperGPQA

Local recovery searches can find a useful context without showing whether a fixed reduction helps overall. We therefore retain or remove the first exposed unit on all 300 SuperGPQA questions, across three methods and two checkpoints per reader. Selection does not depend on observed success, and final inputs are matched across readers. Source, coverage, request, and score checks verify the replay records; identical-input conditions reuse existing outputs. Table 16 reports both corrections and newly incorrect answers relative to full context.

Table 16: Matched-input fixed probes: rescued / newly incorrect questions relative to full context, out of 300.

		First-only		Minus-first	
Method	CP	4B	35B	4B	35B
ExpeL	75	22 / 40	21 / 14	14 / 22	12 / 12
	150	27 / 32	26 / 13	26 / 25	22 / 7
MemRL	75	28 / 25	20 / 14	18 / 25	17 / 12
	150	26 / 21	20 / 15	32 / 21	22 / 8
TF-GRPO	75	32 / 36	22 / 19	21 / 28	16 / 17
	150	34 / 32	18 / 27	11 / 21	9 / 20

Every condition contains both rescues and losses: neither retaining one unit nor removing one is uniformly beneficial. Joint question-bootstrap comparisons suggest a less favorable mean deletion effect for 4B, while the corresponding first-only interval includes zero. The interventions change length and position together and use one repeat, so they characterize the trade-off for these fixed reductions rather than a universal reader ordering.

Inspected calculation changes. We inspected deterministic ID-ordered rescue and loss examples rather than treating every score flip as semantic blame. For TF-GRPO cp150 question `070eca7fd36b44dea50c7d6754696c69`, removing an isotonicity-related first unit changes 4B's electron-energy answer from 0.150 to the gold 0.105 MeV; 35B is correct in both conditions. For `117cab5d1df14f7e9130d033ce442ebb`, the same removal changes 4B's correct thermal-stress calculation into a much longer derivation ending in an explicitly assumed, incorrect option; 35B remains correct. Neither trace explicitly applies the removed pharmaceutical rule. The observable change is in calculation; connecting it to a specific rule would require further intervention. Protocol failures also contribute: for ExpeL question `06eb007eee9f4e29a75bb5537d94bfba`, 4B's full-context output repeats the correct `Answer: F` twice and is rejected by the fixed exactly-one-marker scorer; removing the first unit produces one valid marker. We do not classify this recovery as correction of factual knowledge.

D.4 Full-question replication with three readers

We repeat the fixed deletion intervention to check whether its gains and losses recur and whether they also affect the largest reader. All three readers receive the same frozen 4B-source contexts on 300 SuperGPQA questions for each method at cp150. Two paired replays use seeds 1730/1731, temperature zero, and an 8,192-token cap; identical-input conditions share outputs within each reader and repeat. Independent validation recomputes scores and verifies matched inputs. Banks remain unchanged, so this comparison tests repeated reader use rather than repeated construction.

Table 17 shows recurring losses for every reader–method combination, even where average performance improves. The benefit of deletion varies across readers and methods; the paired ExpeL comparison favors 35B, whereas the other tested cross-reader intervals include zero. Intervals use 10,000 paired question-bootstrap draws after averaging repeats within each question and are unadjusted for multiple comparisons. Temperature-zero replays still differ in their outputs and should not be treated as independent stochastic replications.

Table 17: Fixed deletion replication. Net change is in percentage points; recurring rescue/loss counts require the same direction in both replays.

Reader	Method	Net	95% interval	Rescue/loss
4B	ExpeL	-1.83	$[-5.83, 2.17]$	13/19
	MemRL	2.50	$[-1.83, 6.83]$	21/15
	TF-GRPO	-0.50	$[-4.33, 3.17]$	11/14
35B	ExpeL	3.67	$[0.33, 7.00]$	14/7
	MemRL	3.17	$[0.00, 6.50]$	14/6
	TF-GRPO	1.17	$[-1.67, 4.17]$	10/3
122B	ExpeL	1.50	$[-0.83, 4.00]$	7/3
	MemRL	2.00	$[-0.83, 4.83]$	10/6
	TF-GRPO	0.83	$[-2.00, 3.67]$	7/8

We also separate questions whose answer format remains valid from those whose format changes, retaining all questions in the decomposition. The ExpeL reader contrast persists among valid-format answers, so the observed difference is not confined to formatting failures. Because these groups are defined by generated outputs, the decomposition locates the losses without isolating their cause.

E Details of the Behavioral Cases

The following paired traces support the examples in Table 3. They describe changes in task behavior rather than estimate how often each failure occurs.

HotpotQA: a changed geographic relation. The 4B MemRL case `5a76f7e855429972597f1404` at cp150 asks which Chennai location has different trade laws. The supplied text defines a special economic zone and places Kadaperi near the Madras Export Processing Zone. With either full context or isolated entry 98, the reader instead places the zone inside Kadaperi and answers Kadaperi. Removing entry 98 restores the correct relation and zone name. The entry describes a venue-to-county lookup, suggesting that an extra geographic step may have redirected the answer. Removing entry 88 also restores correctness, so this single-run contrast does not identify one uniquely responsible entry.

SuperGPQA: a disrupted calculation. In the 4B TF-GRPO pendulum case, full context gives the net-force factor $\sqrt{2} + 1/2$ and approximately 1.5084 microcoulombs. Removing the first unit changes geometry and unit conversion, making the answer incorrect in both replays. Here, reducing context disrupts a repeated success rather than removing a source of error.

BFCL: a changed tool argument. In the 35B ExpeL cp150 case `multi_turn_base_152`, the reader passes without memory and with one isolated file-system trajectory, but fails with full context. The full-context run appends booking details to the requested ticket description `Request to cancel.`; the isolated run preserves the checker-required string. The user's wording also mentions reservation specifics, and the exposed trajectories are not ticket-specific, leaving ambiguity about which advice prompted the change. This contrast identifies the changed tool argument, not a uniquely responsible rule.

Context reduction also changes length, position, and sometimes numbering: ExpeL renumbers surviving rules after deletion. These cases therefore describe observed behavioral changes without assigning each change to a single semantic instruction.

F Scope Protection and Full-Loop Comparisons

This appendix records the evaluation conditions for Scope Protection and complete 150/300 tests of validation before memory admission. Small-panel development pilots are omitted; the full-loop negative results are retained.

Scope-protection comparison in the main text. The main-text comparison tests whether changing experience quantity, presentation, or applicability boundaries improves reuse. All four variants are evaluated on the same 1,200 question IDs using binary correctness, with HotpotQA scored by EM. Native MemRL and Top-1 share a replay batch, Compact Examples rebuilds experience, and Scope Protection uses an earlier complete four-task batch. The main table reports help and harm against a shared no-memory reference.

Scope Protection also improves over native MemRL within its own batch: additional corrections outweigh additional errors, although harm itself increases. This supports exploring clearer applicability boundaries, rather than claiming consistent harm reduction. No-memory outcomes vary across batches, evaluation sets were reused during exploration, and Omni-MATH uses diagnostic scoring; independent evaluation remains necessary to establish reliability.

F.1 Full 150/300 development and replication

Table 18: Full HotpotQA loops: 4B, 150 construction and 300 evaluation questions. Panels separate development, frozen-gate transfer, and proposal-source ablations. Tokens are millions.

Development split: admission and update granularity					
	Immediate	Local risk	Local gain	Batch gain	Batch risk
Active entries	150	0	1	40	10
F1	66.67	68.58	68.23	65.81	68.82
EM	52.33	53.33	53.33	51.33	55.33
Calls	900	747	747	694	729
Tokens	1.872	1.363	1.541	1.566	1.676
New split: frozen batch gates				*Development ablations*	
	None	Batch gain	Batch risk	Source-only	Directional
Active entries	0	20	0	0	0
F1	72.36	69.78	72.36	68.58	68.58
EM	58.33	56.67	58.33	53.33	53.33
Calls	300	711	734	774	777
Tokens	0.513	1.493	1.347	1.465	1.474

To test whether validation before admission can preserve performance while acquiring useful experience, all arms in Table 18 use 4B, 150 construction tasks, and 300 final evaluation tasks, with a MemRL-style writer and fixed-Q retrieval. Immediate admission appends proposals without revision and is not the native MemRL baseline. Local gates test individual proposals; batch gates test patches. Candidates remain outside the active bank until paired training-task evaluations support admission. Gain-only gates use the mean score change; risk gates subtract the loss from that change.

The development arms share one split. The new-split evaluation freezes the batch gates before testing (training seed 43, development seed 1042), with question IDs disjoint from the registered historical inventory. The source-only and directional ablations use the development split; the latter guides rewriting with training-time deletion effects.

Calls and tokens include construction, shadow validation, and final evaluation; immediate admission also includes the shared no-memory evaluation. The reused development baseline is 68.5762 F1 and 53.3333 EM; the new split has a fresh baseline. These runs use the historical HotpotQA scorer, not the later answer-prefix protocol.

Table 18 shows that risk-aware batch admission improves on gain-only admission during development, but does not establish an advantage over no memory. On the new split, the frozen risk gate rejects every proposal and preserves the baseline, while gain-only admission grows the bank but lowers performance. Both rewriting ablations also fail to admit useful experience. These results expose a practical difficulty: preventing harmful updates can halt learning rather than make it more effective. Comparisons also change validation exposure, and uncertainty estimates condition on learned banks rather than repeated construction.