MORE IS NOT ALWAYS BETTER: OVERLAPPING GAINS AND COMPOSITION FAILURES IN AGENT ADAPTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

An adaptation channel can improve an agent on its own without adding useful coverage when combined with another. We examine this gap through a frozen 2^3 evaluation of reflective instructions, retrieved memory, and low-rank weight updates on 77 coding tasks. Reflection, memory, and their combination each solve 16 tasks, yet adding memory to reflection rescues two tasks and loses two others. A task-level audit separates four cases of overlapping gains from three cases of composition damage: both can contribute negatively to an additive interaction, although only the latter loses an individual success. No factorial interaction survives Holm correction. We then ask whether this diagnosis supports choices made before generation. A frozen activation rule and a memory-quantity comparison on 169 new tasks establish no corrected advantage. On a further 142 tasks with three paired seeds, assisted applicability selection passes 17/426 evaluations, tying matched-count lexical retrieval and showing no significant advantage over random selection's 13/426. These results distinguish shared coverage from lost successes while exposing the unvalidated step between diagnosing composition failures and preventing them. Generation failures and a disclosed evaluation amendment constrain interpretation.

1 INTRODUCTION

An agent developer has three ways to reuse experience: add a reflective checklist, retrieve past examples, or update model weights. Each can help on its own. Should the developer enable all three? The answer cannot be read from the individual scores. Two channels may solve the same previously missed tasks, leaving little additional coverage when combined. A combination may also lose an answer that one channel produced correctly. These are different deployment outcomes, even when a single interaction statistic gives them the same sign.

The literature makes the composition question concrete. Reflexion retains verbal feedback across attempts, while ExpeL extracts insights and retrieves experience for later tasks (Shinn et al., 2023; Zhao et al., 2023); MetaReflection further studies instruction learning from past reflections (Gupta et al., 2024). In parameter space, task arithmetic and LoraHub study how learned updates can be combined (Ilharco et al., 2022; Huang et al., 2023), and TIES-Merging explicitly addresses interference in model merging (Yadav et al., 2023). These approaches operate on different adaptation representations. They motivate examining their behavioral composition, but do not imply that an observed loss from combining text and weights has the same mechanism as weight-space interference.

This distinction matters for coding assistants that accumulate guidance and examples over time. Their immediate choice is concrete: keep memory on, omit it for a particular task, or retrieve fewer examples. Aggregate accuracy is necessary for evaluating that choice, but it conceals which capabilities were preserved and which were exchanged. In our factorial study, reflection, memory, and their combination each solve 16 of 77 tasks. Yet adding memory to reflection rescues two tasks and loses two others. The combined score is unchanged; the set of solved tasks is not.

We study this gap between *aggregate improvement* and *preservation of individual gains*. Our setting combines a frozen reflective instruction artifact (π), task-dependent retrieved experience (M), and

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.

a frozen low-rank weight update (θ). Existing systems already combine language-based guidance and episodic experience (Shinn et al., 2023; Zhao et al., 2023), and LoRA provides an established parameter-adaptation mechanism (Hu et al., 2021). Our contribution is not another way to accumulate these resources. It is an evaluation that asks what survives when they are used together, and whether observations made after evaluation support a useful decision before the next generation.

Context also makes deployment choices consequential. Few-shot demonstration order can change performance (Lu et al., 2022), and access to relevant information depends on its position in long contexts (Liu et al., 2023). We take these findings as reasons to control presentation and token allowances, not as evidence that our short memory blocks reproduce the same effects. A composition study must distinguish changes in what information is supplied from changes in how it is presented.

The key interpretive issue arises even for binary outcomes. If a baseline fails, either channel succeeds, and their combination succeeds, the additive interaction is negative: the individual gains cover the same task. Nothing was lost. A negative contribution can also occur when the combination fails despite an individual success. Treating both as destructive interference conflates shared coverage with damage. We therefore report paired task outcomes alongside factorial contrasts. These contrasts concern the pass-probability scale; they do not identify a conflict between internal representations.

The evidence follows a sequence from measurement to diagnosis to prospective decision tests. First, a chronological LiveCodeBench split (Jain et al., 2024) supports a frozen 2^3 evaluation on 77 test tasks, with reversed reflection–memory presentation and a secondary matched-allowance regime. Its 1,386 condition records represent 1,232 distinct generations. No interaction survives Holm correction. A post hoc audit nevertheless distinguishes four overlapping-gain cases from three observed composition-damage cases. The distinction explains what an interaction can mean without claiming a population-level antagonism or a generally superior presentation order.

Second, we test whether a diagnosis can become a deployment rule. A frozen activation router and a memory-quantity comparison on 169 new tasks do not establish corrected gains. We then evaluate assisted applicability selection on 142 new tasks with three paired seeds and matched-count controls. Selection passes 17/426 evaluations, tying same-count lexical retrieval; its advantage over random selection's 13/426 is not statistically established. These completed tests are essential to the story: recognizing a potentially relevant past procedure did not, under the tested conditions, become a reliable rule for improving the next answer.

The paper makes three bounded contributions. It provides a controlled comparison of heterogeneous adaptation channels; it separates overlapping gains from lost successes using auditable task-level outcomes; and it tests the practical step from diagnosis to activation, quantity, and content selection without presenting nonsignificant point estimates as solutions. Output degeneration, evaluation incidents, single-model scope, and non-independent applicability review limit generalization. Our supported conclusion is correspondingly specific: evaluating composition requires knowing both how often the system succeeds and which individual successes it retains. Figure 1 summarizes the conceptual progression.

2 Composition as Preservation and Selection

The practical promise of adaptation is cumulative: experience acquired through one channel should remain useful when another is added. Yet a combined system is a new solver configuration, not the union of its components' successful answers. We therefore frame composition around two linked demands: *preservation*, which asks what happens to individual successes after channels are combined, and *selection*, which asks whether information available before generation can guide that combination. The first is observable through paired outcomes; the second requires a separate prospective test.

A score difference is a balance of gains and losses. Let $Y_i(c) \in \{0, 1\}$ denote the recorded evaluation outcome on task i under configuration c, with the study's fixed seed and scoring convention understood. For two configurations a and b, define the rescued and lost task sets

$$\mathcal{R}_{a\to b} = \{i : Y_i(a) = 0, Y_i(b) = 1\}, \qquad \mathcal{L}_{a\to b} = \{i : Y_i(a) = 1, Y_i(b) = 0\}. \tag{1}$$

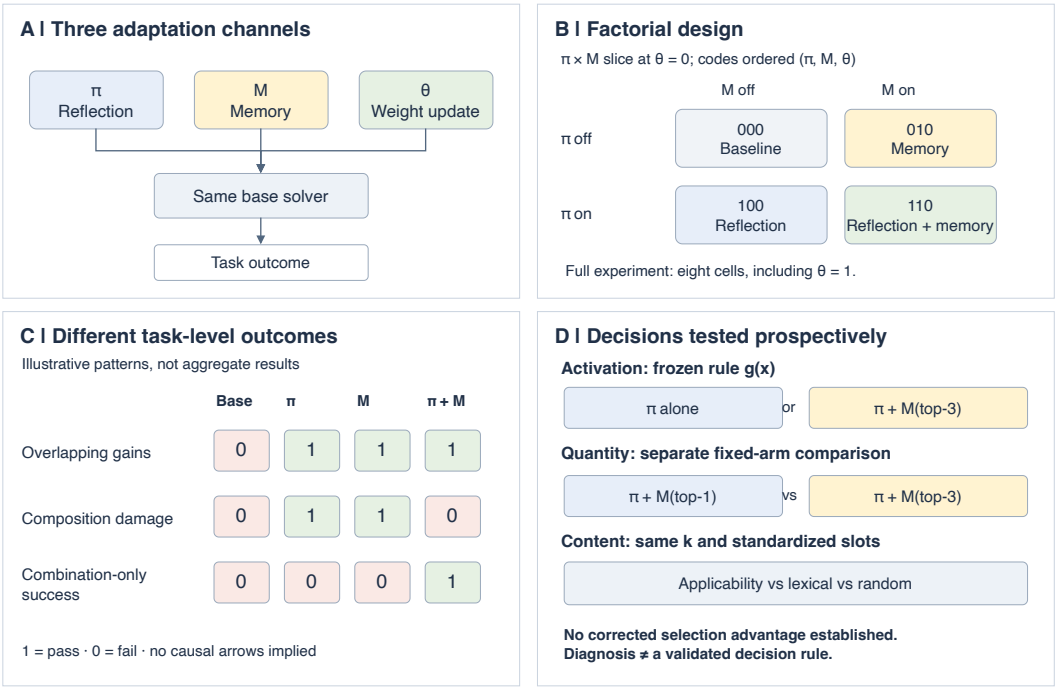

Figure 1: From composition measurement to prospective decisions. (A) Three adaptation channels influence the same base solver. (B) A reflection–memory factorial slice at fixed $\theta = 0$, with codes ordered (π, M, θ); the full design contains eight cells. (C) Illustrative binary outcome patterns distinguish overlapping gains, composition damage, and combination-only success. They are not aggregate estimates or a claim that every pattern was observed. (D) The implemented activation router chooses reflection alone or reflection plus top-three memory; quantity is tested separately. Matched-count content controls compare applicability, lexical retrieval, and random selection. None of the selection comparisons establishes a multiplicity-corrected advantage.

Their score difference satisfies the exact identity

$$\overline{Y}(b) - \overline{Y}(a) = \frac{|\mathcal{R}_{a\to b}| - |\mathcal{L}_{a\to b}|}{n}. \tag{2}$$

Thus an unchanged score can mean unchanged outcomes or an exchange of solved tasks. Reporting both sets exposes this difference. The identity is elementary accounting, not a new estimator or evidence of a causal mechanism. In multi-seed studies it applies within each paired seed and is then averaged; task-level inference retains the pairing. A recorded failure can include a protocol-defined execution or generation failure, so outcome preservation is narrower than a claim about latent reasoning ability.

Non-additivity does not identify what was lost. We instantiate a configuration as $c = (p, m, t; o)$: frozen reflective instructions p, retrieved memory m, a weight update t, and presentation order o. The three binary channel indicators define eight factorial cells. At fixed $\theta = 0$ and forward presentation, the reflection–memory contrast is

$$D_i^{\pi M|\theta=0} = Y_i(1,1,0) - Y_i(1,0,0) - Y_i(0,1,0) + Y_i(0,0,0). \tag{3}$$

Consider outcomes ordered as baseline, reflection, memory, and their combination. Pattern $(0,1,1,1)$ gives $D_i = -1$ because both channels cover the same previously failed task; the combination retains success. Pattern $(1,1,1,0)$ also gives $D_i = -1$, but now composition loses a success preserved by both individual channels. The same contrast therefore supports different descriptions of behavior. Overlap concerns shared coverage; damage concerns a lost success. Neither observation alone identifies why the model produced its answer.

The factorial contrasts and the transition sets consequently serve complementary purposes. Six conditional pairwise contrasts and one three-way contrast measure departures from additivity on the

pass-probability scale; paired outcomes explain which observed patterns contribute to them. The pattern audit is post hoc and distinct from the frozen inferential tests. Exchanging reflection and memory blocks while holding their content fixed probes sensitivity to presentation, not the order in which adaptation states were learned. It cannot establish general noncommutativity of learning operators.

A diagnosis becomes useful only if it informs the next decision. After grading, one can identify rescued tasks or select whichever configuration succeeded. A deployed solver cannot use those outcomes. Let x_i contain only information available before generation, let $g(x_i)$ choose a configuration, and let c_{dev} be the best fixed configuration selected on development data. The operational value of selection is

$$\Delta_g = \frac{1}{n} \sum_i \left[Y_i(g(x_i)) - Y_i(c_{\mathrm{dev}}) \right]. \tag{4}$$

The rule and target-task choices must be frozen before observing the target solver outcomes. An outcome-informed oracle can describe remaining coverage, but cannot establish a deployable advantage. In particular, a visible loss under composition does not imply that the loss was predictable from x_i.

Our prospective tests make this distinction concrete at three levels. Activation chooses whether memory accompanies reflection; quantity compares a fixed top-one choice with top-three retrieval; content compares assisted applicability judgments with lexical and random choices. The quantity contrast changes both content and prompt length. The applicability study instead holds card presentation fixed, and its matched-count controls hold the number of selected memories fixed per task. A source procedure can supply a useful substep without solving the target problem, so applicability permits explicit prerequisites, failure warnings, and abstention rather than equating shared vocabulary with transfer.

This framework links diagnosis to a falsifiable application: can a choice made before generation improve outcomes over a fixed alternative? The completed experiments test particular rules, not the existence of every possible useful selector. They also separate two conclusions that should not be collapsed: composition can exchange individual successes, while a proposed policy for avoiding those exchanges may still fail to improve performance.

3 Experimental Design

We use a chronological LiveCodeBench split with 60 adaptation, 30 validation, and 77 test tasks. The benchmark is described by its official repository (LiveCodeBench, 2026). Reflection and memory artifacts are frozen; the weight channel uses a fixed LoRA adapter trained on successful adaptation examples. No test-dependent artifact updates are performed.

Nine conditions (eight forward cells and reversed 110) are evaluated under two allowance regimes. The primary output ceiling is 8,192 tokens. For task i, matched allowance sets $B_i = L_{i,000} + 8192$ and caps the output for condition c at $\min(8192, B_i - L_{i,c})$, retaining the full prompt. Equal allowance does not imply equal realized tokens or FLOPs. Identical complete requests can reuse one saved generation.

The task is the inferential unit. We use 100,000 platform-by-difficulty stratified paired bootstrap draws and pointwise intervals; the exact sign-flip test assumes null sign symmetry or exchangeability. Holm correction is applied to seven interactions within each regime, with the matched regime secondary; two order comparisons form a secondary family. Infrastructure errors score zero in the original fixed-denominator analysis. One grading-only replay is reported separately as sensitivity. The detailed protocol was frozen prospectively after validation, not retrospectively designated an earlier preregistration.

How the studies fit together. The factorial experiment measures composition, the task audit interprets its outcome patterns, and the later studies test operational decisions on new tasks. The 169-task study generates five arms once per task. The 142-task study generates four arms with three paired seeds, followed by two precommitted matched-count arms, for 2,556 evaluations. These studies use separate hypothesis families and are not pooled into a single significance claim. Unlike the his-

torical fixed-denominator error scoring, the Pro study stops on unresolved infrastructure incidents and retains explicit adjudications; its disclosed observer amendment is described in Appendix F. All reported selection decisions precede their target solver outcomes.

4 RESULTS

Evidence base. All 1,386 condition records completed, with 1,232 distinct generation requests and 154 reused records. Complete record and request audits passed. First grading produced 83 error records across 76 distinct requests. After one replay per request, 34 requests still returned errors, affecting 36 condition records. Original results remained unchanged.

Higher individual scores do not guarantee further gains. In first grading, reflection and memory each pass 16/77 tasks, compared with 10/77 for the baseline; their combination also passes 16/77. Turning on all three channels still yields 16/77, while reflection plus weight adaptation reaches 18/77. These are descriptive counts, not significance claims or evidence for a generally optimal configuration. Equal totals also need not mean the same tasks were solved, motivating the paired audit below.

Table 1: Pass counts out of 77 tasks. Counts are identical across the two allowance regimes within each analysis scope. Replay sensitivity does not replace the primary endpoint.

Condition	First grading	Sensitivity
000	10	10
100	16	18
010	16	19
001	13	13
110	16	19
101	18	18
011	14	14
111	16	16
110_reverse	13	16

Equal scores hide different solved tasks. Table 2 decomposes score changes into paired transitions on all 77 primary tasks. Adding memory to reflection rescues two tasks but loses two previously passed tasks. Adding memory to reflection plus weight adaptation rescues four and loses six. These post hoc counts retain the original failure scoring; "rescue" and "loss" describe observed test outcomes, not proven cognitive mechanisms. They motivate reporting both sides of a composition decision rather than only its net score.

Table 2: Adding memory: paired primary outcomes on all 77 tasks. Rescue is fail-to-pass; loss is pass-to-fail. Failures include the original error-scored records. No new generations or grading were performed.

Comparison	Rescue	Loss	Net	Both pass
$000 \rightarrow 010$	7	1	$+6$	9
$100 \rightarrow 110$	2	2	0	14
$001 \rightarrow 011$	8	7	$+1$	6
$101 \rightarrow 111$	4	6	-2	12

Negative estimates do not establish robust antagonism. All seven first-pass interaction Holm-adjusted p-values are 1. The smallest sensitivity adjusted p-value is 0.533. For reflection–memory at $\theta = 0$, the primary estimate is -7.79 percentage points (pp), with interval $[-16.88, 1.30]$; sensitivity gives -10.39 pp, interval $[-20.78, -1.30]$, exact $p = 0.076$ and adjusted $p = 0.533$. The percentile

interval and sign-flip test are different inferential constructions; the former does not override the prespecified test.

No stable presentation-order advantage is established. The order estimate is $+3.90$ pp, interval $[-1.30, 9.09]$, and adjusted $p = 0.750$. Neither an interaction nor a stable ordering advantage is established.

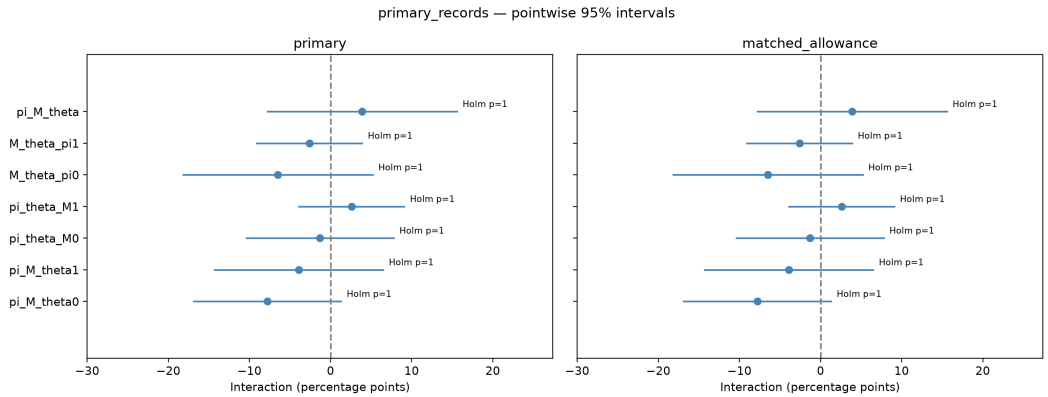

Figure 2: First-grading interactions. Intervals are pointwise; matched allowance is secondary. Complete source hashes and sensitivity figures accompany the analysis artifacts.

The same contrast can describe different outcomes. A post hoc audit of reflection–memory quartets distinguishes four overlapping-gain tasks from three tasks on which composition loses a success of at least one individual channel. Figure 3 contrasts two examples with the same task-level contribution, $D_i = -1$. On `abc382_a`, both individual channels and their combination pass, so the negative contrast reflects shared coverage. On `abc384_f`, the baseline and both individual channels pass the saved tests, whereas the combination makes an invalid algebraic substitution and fails. These selected cases explain the meaning of the contrast; they are not an independent estimate of failure prevalence.

5 From Diagnosis to Composition Decisions

A rule must choose before it sees the answer. The task audit distinguishes preserved and lost successes after evaluation. The activation test asks whether information available beforehand is sufficient to make a useful choice. We fit a one-split decision tree on the old 30-task validation set, using only top-memory lexical Jaccard similarity or task word count, with at least eight tasks per leaf. Its actions are the four $\theta = 0$ configurations. The best fixed development configuration is 110 (12/30); the selected tree fits 13/30. These are development scores, not evidence of generalization. Historical development checkpoint and grading limitations carry over.

The frozen rule uses 110 when the top-memory similarity is at most 0.0940239625949737 and 100 otherwise. Its direction is learned from development outcomes, not a claim that relevant memories are generally harmful. On the new-task manifest it selects 110 for 122 tasks and 100 for 47, before any candidate generation. For the quantity test, a separate fixed arm retains only the highest-ranked memory while keeping reflection and presentation order unchanged.

New tasks provide the decision test. The protocol includes 169 tasks in 39 contest clusters, later than the old task window and screened for ID and lexical near-duplicate overlap. Two of 171 candidates were excluded before generation because their full test requests exceeded the fixed size bound; tests were not shortened. Each task receives five configurations with one paired seed, for 845 completed generations. Router performance uses the outcome of its preselected arm; no extra generation or test-time answer selection is involved. We report pass rate, gained and lost successes,

Task	**Shared coverage** `abc382_a`	**Lost correctness** `abc384_f`
Baseline / π / M / $\pi + M$	Fail / Pass / Pass / Pass	Pass / Pass / Pass / Fail
$D_i = Y_{110} - Y_{100} - Y_{010} + Y_{000}$	$1 - 1 - 1 + 0 = -1$	$0 - 1 - 1 + 1 = -1$

A. Different code, same count.
Let C be the number of cookies.
π: `N - (C - D)`
M: `N - len(cookie_boxes[D:])`
$\pi + M$: `N - max(0, C - D)`
Under the task constraint $C \geq D$, all three expressions give the same answer. The combined solution retains success.

B. A transformation changes the answer.
Let f remove all factors of two.
Single channels sum $f(A_i + A_j)$.
The combination instead sums $f(f(A_i) + f(A_j))$.
For the saved input $A = [4, 8]$:
original: $1 + 3 + 1 = 5$;
transformed: $1 + 1 + 1 = 3$.
The rewrite does not preserve the task.

Figure 3: Two audited tasks with equal negative interaction contributions at fixed $\theta = 0$, using primary first-grading records. Code expressions in A are shortened with shared notation; B summarizes the implemented transformation and an existing failing test. No new generation or grading was performed. Passing denotes the saved evaluator outcome, not a guarantee at maximum input size.

Table 3: New-task composition decisions, fixed $\theta = 0$. Errors count as zero. Router rows reuse preselected outcomes; they are not extra generations.

Configuration	Pass / 169	Eval. errors	Mean input tokens
No adaptation	44	18	558
Reflection	46	19	1,315
Memory	42	12	837
Reflection + top-three	42	17	1,594
Reflection + top-one	44	15	1,417
Frozen router	47	19	1,518

errors, and token cost. The activation and quantity comparisons form a separate Holm family, with contest-cluster resampling for intervals. Appendix E gives the frozen design.

Positive point estimates do not establish a decision rule. The frozen router passes 47/169 tasks, compared with 42/169 for the development-selected fixed combination: six gains and one loss, a 2.96 pp difference (pointwise 95% contest-bootstrap interval $[0.00, 5.85]$; Holm $p = 0.25$). This positive estimate does not establish a reliable advantage. Reflection alone passes 46/169, only one fewer than the router; this descriptive comparison was not a prespecified test. Top-one memory passes 44/169 versus 42/169 for top-three (1.18 pp, interval $[-3.57, 6.08]$; Holm $p = 0.82$), with eight gains and six losses.

The router uses 4.8% fewer input tokens and 8.7% fewer completion tokens than fixed 110 on average; routing overhead is not separately timed. The outcome-informed four-arm oracle passes 56/169 and is a diagnostic upper bound, not deployable performance. Across the five generated arms, 81 evaluation errors comprise 74 candidate-or-harness execution failures and seven container OOMs. They limit algorithmic interpretation of the operational endpoint. Neither prespecified comparison establishes a useful general rule; the experiment makes the decision question measurable without claiming it solved.

Content selection under matched quantity. The content-selection test asks whether selecting a transferable procedure improves on lexical retrieval. Reflection and the model are fixed across four arms: no memory (A), original lexical top-three summaries (B), the same lexical choices in standardized cards (C), and assisted applicability choices using those cards (D). C and D have identical full prompt lengths per task. All 142 tasks and three paired seeds are evaluated, followed by two precommitted controls that match each task's selected card count. Appendix F details construction, review limitations, and execution.

A and C each pass 21/426 evaluations; B and D each pass 17/426. D–C and D–A are both -0.94 pp, with Holm $p = 0.9375$ for each. In the separate matched-count family, lexical retrieval passes 17/426 and random selection passes 13/426. D–lexical is 0.00 pp (95% interval $[-2.07, 1.53]$, Holm $p = 1$); D–random is $+0.94$ pp (interval $[-0.25, 2.44]$, Holm $p = 0.75$). Selection neither demonstrates a gain over lexical retrieval nor establishes equivalence to it. The random contrast is a positive estimate without a confirmed advantage.

The operational result includes generation reliability. Across the four primary arms, 598 of 1,704 evaluations fail at generation: 594 reach the output limit and four stopped outputs contain syntax errors. These outcomes remain in the denominator. A repeated-line audit describes degeneration but cannot attribute it to memory content. The experiment therefore evaluates the full configured solver, rather than an isolated measure of semantic applicability. Taken together, the prospective studies show the boundary of the diagnostic result: observing lost successes does not itself supply an effective rule for preventing them.

6 Discussion and limitations

The clearest implication is about what to measure. An aggregate score tells a developer the net change in success; paired transitions show which successes were gained and lost. An interaction estimate adds a different question—whether joint performance is additive—but its sign alone cannot distinguish shared coverage from damage. The audit makes that distinction concrete without establishing population-level antagonism: no factorial interaction survives correction, and nonsignificance is not evidence of equivalence or absence of interaction.

The cases also suggest different interventions. Shared coverage calls for examining complementary tasks. An invalid algebraic rewrite calls for checking whether a transformation preserves the problem. Repetitive generation calls for investigating termination and extraction reliability. We tested one of these ideas in a separately frozen exploratory study on 30 previously used validation tasks with three seeds. Targeted transformation checks pass 24/90 runs, versus 28/90 for token-matched general review and 29/90 for the original prompt. The targeted-minus-general difference is -4.44, interval $[-11.11, 1.11]$. Fewer length finishes do not translate into more passes. Because the intervention follows the audit and reuses validation tasks, it neither independently confirms the original interaction nor identifies a mechanism.

Several limits remain material. The study is conditional on one model setting, one adapter, the artifact construction, and the task-seed policies. Historical checkpoint provenance is incomplete. Evaluation failures affect scores and cannot be assumed missing at random; later synthetic evaluator diagnostics do not identify the cause of each historical error. Pointwise intervals, small strata, and sign-symmetry assumptions constrain inference. Independent-task power simulations guide a future factorial study but do not establish power for the contest-clustered selection experiments (Appendix B).

Applicability selection has additional limitations. Review is assisted and non-independent; prior engineering-answer exposure prevents a claim of blind review. Padding makes an empty card selection different from reflection alone, and equal token counts do not make padding inert. A single random draw per task does not average over all possible memory sets. Frequent generation failures and the post-start evaluation amendment further limit attribution to memory content. These conditions bound the negative result; they do not justify excluding it or concluding that every applicability selector must fail.

7 Related work

Reflection and memory already coexist in agent systems. Reflexion turns feedback into retained reflective text, while ExpeL extracts insights from experience and retrieves both insights and earlier attempts (**??**). MetaReflection learns instructions from past reflections (**?**). ACE maintains an evolving playbook through generation, reflection, and curation, and MemGPT manages movement between memory tiers under a limited context window (**??**). Our distinction between π and M is operational: a shared instruction artifact versus task-dependent retrieved summaries. Holding them

fixed permits an audit of joint behavior; it does not propose a universal taxonomy of memory or evaluate online playbook evolution.

Self-Refine iterates feedback and revision on a current output (**?**). Our principal experiment instead uses frozen cross-task artifacts and one generation per condition, so it does not compare composition with iterative self-repair under additional model calls. This separation matters when interpreting a failed combination: a further repair attempt is a different intervention from choosing which accumulated resources accompany the initial answer.

LoRA provides the weight-adaptation mechanism (**?**). Task arithmetic, LoraHub, and TIES-Merging examine composition in parameter space (**???**). We combine one weight update with two textual channels rather than merge independently trained updates. The behavioral contrast in Equation (3) cannot identify the redundant changes or sign disagreements studied in model merging, nor does this fixed-task evaluation measure forgetting across sequential training stages.

Demonstration order can affect few-shot performance, and relevant information can become harder to use at particular positions in long contexts (**??**). These findings motivate our presentation and allowance controls without implying that short memory blocks reproduce the same effects. LiveCodeBench provides timestamped contest tasks for chronological separation (**?**). Such separation prevents constructing our adaptation artifacts from the later test split, but does not guarantee absence of pretraining contamination. Within this scope, our contribution is the empirical link between paired outcome preservation and separately tested selection decisions, rather than a new interaction formula or adaptation mechanism.

8 Conclusion

Adding memory to reflection leaves the aggregate score unchanged in our factorial study while rescuing two tasks and losing two others. The paired audit further shows that overlapping gains and lost successes can produce the same negative interaction contribution. Composition should therefore be evaluated through both total success and preservation of individual successes. The prospective studies establish no corrected advantage for the tested activation, quantity, or applicability policies. They leave a precise open problem: identifying a composition failure after evaluation is easier than choosing a configuration that prevents it before generation.

AI use statement

The conceptual illustration in Figure 1 was redrawn as an editable vector diagram with AI coding assistance, using an author-supplied schematic as a visual reference. Its labels were corrected to match the experimental design; it is not empirical data. Generative AI assistance, including OpenAI Codex, was used to propose experimental and analytical designs, implement and inspect experiment and analysis code, search for literature, and draft and revise manuscript text. Automated checks compared reported counts with saved records and checked compilation and citations. Such checks do not substitute for scientific or source verification. The evaluated language model also generated candidate programs and adaptation materials as described in the experimental protocol. Final human review of the AI-assisted text, code, references, and interpretations remains pending in this working draft.

Reproducibility statement

Frozen configurations, task manifests, model and artifact identifiers, generation journals, evaluator records, analysis scripts, and audit receipts are maintained with this study. The main text describes the evaluation and inference procedures; the appendices describe task-level diagnostics and subsequent exploratory studies. Historical checkpoint provenance and residual evaluation errors are disclosed as limitations. An anonymous supplementary release has not yet been prepared for this working draft.

REFERENCES

Priyanshu Gupta, Shashank Kirtania, Ananya Singha, Sumit Gulwani, Arjun Radhakrishna, Gustavo Soares, and Sherry Shi. MetaReflection: Learning instructions for language agents using past reflections. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 8369–8385, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.477. URL https://aclanthology.org/2024.emnlp-main.477/.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. arXiv:2106.09685, 2021. URL https://arxiv.org/abs/2106.09685. arXiv record; accessed 2026-09-14.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. LoraHub: Efficient Cross-Task Generalization via Dynamic LoRA Composition. arXiv:2307.13269, 2023. URL https://arxiv.org/abs/2307.13269. arXiv record; accessed 2026-09-14.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing Models with Task Arithmetic. arXiv:2212.04089, 2022. URL https://arxiv.org/abs/2212.04089. arXiv record; accessed 2026-09-14.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and Contamination Free Evaluation of Large Language Models for Code. arXiv:2403.07974, 2024. URL https://arxiv.org/abs/2403.07974. arXiv record; accessed 2026-09-14.

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. arXiv:2307.03172, 2023. URL https://arxiv.org/abs/2307.03172. arXiv record; accessed 2026-09-14.

LiveCodeBench. Livecodebench: Official repository. https://github.com/LiveCodeBench/LiveCodeBench, 2026. Accessed 2026-09-14.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8086–8098, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.556. URL https://aclanthology.org/2022.acl-long.556/.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language Agents with Verbal Reinforcement Learning. arXiv:2303.11366, 2023. URL https://arxiv.org/abs/2303.11366. arXiv record; accessed 2026-09-14.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-Merging: Resolving Interference When Merging Models. arXiv:2306.01708, 2023. URL https://arxiv.org/abs/2306.01708. arXiv record; accessed 2026-09-14.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM Agents Are Experiential Learners. arXiv:2308.10144, 2023. URL https://arxiv.org/abs/2308.10144. arXiv record; accessed 2026-09-14.

A REPRODUCIBILITY AND REMAINING MANUSCRIPT WORK

The project includes frozen inputs and execution records, primary and replay audits, independent statistical recomputation, and six source-indexed figures. See the companion reproduction guide and delivery index. Further manuscript work includes further related-work coverage checks, complete artifact-construction details, all contrast tables, and the final venue format. This working draft uses the official ICLR 2027 anonymous submission style; scientific and author review remain in progress.

B Prospective Sample-Size Planning

After inspecting the completed study, we adopted an absolute interaction of 10 percentage points (pp) and at least 80% target power as planning criteria for an independent follow-up. These are prospective design choices, not preregistered criteria for the completed experiment or equivalence margins. A 5 pp effect remains a desirable precision goal, while 15 pp provides a large-effect reference.

A dedicated 171-task simulation comprised 32 scenarios with 1,000 replicates each. For a negative 10 pp interaction, target rejection rates ranged from 6.0% to 40.6%. We subsequently evaluated 600, 1,200, and 1,800 independent tasks in 72 additional scenarios, each with 1,000 replicates. Both directions of the 10 pp effect were considered, alongside global-null scenarios. No additional model generations were performed.

Table 4: Synthetic planning results for the target reflection–memory interaction at $\theta = 0$. Ranges span 16 nonzero scenarios per sample size; they are not confidence intervals. The final column is the smallest scenario-wise Wilson 95% Monte Carlo lower bound.

Independent tasks	Target rejection rate	MC lower bound
600	38.2–98.0%	35.2%
1,200	75.2–100.0%	72.4%
1,800	93.6–100.0%	91.9%

All eight cells shared baseline success probability 0.25 or 0.50, except cell 110, whose probability was shifted by ± 0.10. Task-level uniforms were shared across cells with mixture probability 0 or 0.7 and otherwise independent; this parameter is not a Pearson correlation. Independent cell deletion at rate 0 or 0.03 was scored as zero, reducing the observed effect magnitude to 9.7 pp in the latter case. This mechanism changes multiple true contrasts. Target power refers only to the reflection–memory contrast at $\theta = 0$, with all seven two-sided sign-flip tests retained in the Holm family.

The 1,800-task setting was the first tested grid point to exceed 80% target power across these scenarios. It is neither an exact minimum nor a guarantee under the actual task distribution. That historical 171-task pool subsequently yielded the 169-task decision study after eligibility checks. This simulation concerns the seven factorial contrasts; it does not establish power for the separate 142-task assisted-selection draft, which uses contest clusters and two primary contrasts. Contest dependence, task-dependent evaluator failures, model variation, and asymmetric null distributions are not covered by this simulation. A new task source, dependence-aware analysis, and attainable sample-size plan must be frozen before follow-up generation.

C Exploratory Task-Level Composition Audit

This post hoc audit uses existing generations and grades only. For the four conditions $(000, 100, 010, 110)$, the primary 77-task contrast sums to -6. Four tasks exhibit pattern $(0, 1, 1, 1)$: both individual channels and their combination succeed, so the negative contribution reflects overlapping gains rather than a failed combination. Three tasks exhibit combination failure despite success of at least one individual channel, contributing -4 in total. One task has pattern $(1, 0, 0, 1)$, contributing $+2$ by restoring baseline performance. The remaining patterns contribute zero.

The three observed damage cases have different signatures. On task 3649, the combination repeats comments until the 8,192-token limit and yields no extractable complete program, whereas both individual channels pass. On `abc384_f`, the combined program substitutes $f(f(a) + f(b))$ for $f(a + b)$, where f removes powers of two. For the saved input $A = [4, 8]$, it outputs 3 instead of 5. The other three programs pass the available tests using quadratic enumeration; their test passes do not establish suitability at the largest problem sizes. On task 3625, the combined program mishandles turn state, while the reflection-only condition retains an evaluation error. This quartet cannot be treated as four clean model outcomes.

There are four primary tasks on which the combination passes while at least one individual channel fails. Three preserve the successful individual channel's performance and contribute zero interac-

tion; the fourth restores an already successful baseline. Replay sensitivity adds two overlapping-gain tasks, producing a contrast sum of -8, while the three damage task identities remain unchanged. Eleven primary quartets contain evaluator errors, falling to six after the previously allowed replay. No additional grading or generation was performed.

These cases do not establish a conflict between reflection and retrieved memory. Observed truncation and invalid algorithmic rewriting motivate separate interventions with prompt-length and compute controls. A future mechanism test must use independent tasks and prespecified seeds; selecting only the present failures would not validate a general antagonism claim.

C.1 Three reproducibly selected illustrations

For this additional post hoc inspection, we choose the lexicographically first task ID in each of three outcome-defined groups: rescue and loss for $100 \to 110$, and overlap $(000, 100, 010, 110) = (0, 1, 1, 1)$. This deterministic rule selects `abc382_c`, 3649, and 3659; it does not make the selection prospective or representative. We inspect saved prompts, outputs, and grades without new generation or execution.

Rescue: removing an invalid changing state. On `abc382_c`, each sushi item must independently reach the first eligible person. The reflection-only program increments a persistent person index and encounters an `IndexError` on a saved test. The combined program independently scans people for each item and passes all 43 saved tests; its code is identical to the memory-only output. Retrieved summaries concern grid checks, ring movement, and color counts, so the record does not identify a particular memory as the cause of the correction. Moreover, the successful program has worst-case $O(NM)$ complexity: passing these tests is not evidence of scalability to all allowed inputs.

Loss: output degeneration rather than demonstrated instruction conflict. On 3649, the reflection-only output enumerates lock orders and passes the saved tests. The combined output repeatedly comments on energy accumulation, reaches the 8,192-token output limit, and supplies no extractable complete program. Its memories concern team assignments, substring counting, and triplet search. The observed failure is truncation with repetitive output; claiming that conflicting advice caused it would require a controlled intervention. The source comments themselves are not correctness evidence.

Overlap: two routes to the same covered task. On 3659, the baseline grid-XOR program reads predecessor states without boundary guards and fails a saved test. Reflection alone uses guarded dynamic programming, memory alone uses bounded memoized traversal, and the combination uses a 16-value XOR dynamic program; all three pass the 43 saved tests. The memories include partition search, a failed team-assignment attempt, and a grid-path attempt with invalid code. Their presence is not proof of causal transfer. The pattern demonstrates shared coverage, not a lost solution, despite a negative task-level additive contrast.

D Completed Exploratory Prompt Intervention

A separately frozen study completed 30 previously used validation tasks, three paired seeds, and three prompt conditions (270 generations): the original reflection–memory combination, an additional general-review instruction, and an additional instruction to verify algebraic or state transformations against the problem examples. The latter two conditions have exactly equal rendered input-token counts for each task and share the same 8,192-token output cap. The principal descriptive comparison is targeted versus general review; this study does not re-estimate factorial interactions.

All test cases were retained. An independently versioned client used the fixed v2 worker with a 16 MiB request bound, verified on a synthetic 10 MB input; the historical evaluator and results remain unchanged. Each request received one generation and one grading attempt, without retries. Full audits verified frozen hashes, 270 unique requests, generation–grading bindings, paired seeds and memory, and equal general/targeted input-token counts. Independent recomputation reproduced the counts and intervals.

Table 5: Completed exploratory validation study. Each arm contains 30 tasks with three seeds, not 90 independent tasks. Errors and generation failures are scored zero in fixed-denominator pass counts.

Prompt	Pass / 90	Eval. errors	Gen. failures	Length finishes
Original	29	7	21	21
General review	28	6	23	23
Targeted check	24	7	17	17

The targeted-minus-general difference is -4.44 pp, with a pointwise 95% task-bootstrap interval of $[-11.11, 1.11]$. Targeted-minus-original is -5.56 pp, interval $[-14.44, 1.11]$; general-minus-original is -1.11 pp, interval $[-8.89, 6.67]$. The 20,000 bootstrap samples resample whole tasks, retaining all three paired seeds; no confirmatory p-values are introduced. All intervals include zero. The targeted check did not demonstrate an improvement over general review.

Twenty evaluation errors comprise 14 candidate-or-harness execution failures and six container out-of-memory events. Their per-arm counts are respectively $5/2$, $5/1$, and $4/3$ (original, general, targeted). The 61 generation failures coincide with length finishes; these are distinct from evaluation errors. Targeted prompts had fewer length finishes but also fewer passes, so reduced truncation alone is not evidence of improved correctness. No assumption of randomly missing evaluation outcomes is made.

This intervention was designed after the test-case audit and uses previously examined validation tasks. It is exploratory and cannot independently confirm antagonism or identify an internal mechanism. The result neither validates the proposed check as a remedy nor establishes that semantic verification is generally ineffective. It leaves the frozen formal seven-test conclusions unchanged.

E Frozen New-Task Composition Policy Protocol

Selection and provenance. The new tasks come from the fixed LiveCodeBench materialized release revision `25d8cb8f0db2efe1b589941eb8c26a219850d4d2`. We screen against the old task tranche and earlier accepted candidates, excluding identical IDs and five-word-shingle Jaccard similarity at least 0.8. This is not exhaustive semantic deduplication or a pretraining-contamination guarantee. Before generation, `arc190_c` and `abc396_f` exceeded the 16 MiB evaluation-request bound with a 256 KiB code reserve; the other 169 retain all available tests.

Development-only fitting. For each of two features, candidate splits are midpoints between distinct development values. Each leaf has at least eight tasks and chooses among $000, 100, 010, 110$ by development pass count; ties prefer shorter total prompt length and then a fixed configuration order. The best split is selected by pass count, prompt cost, and deterministic feature/threshold ordering. It must improve the development count by at least one task over the best fixed configuration; otherwise the policy is constant. Baseline development records follow the existing historical validation-analysis selection rule (last non-error record), not new retries. These historical data have incomplete checkpoint provenance. No old test labels or new-task outcomes enter fitting. The development artifact, split threshold and per-task selections are hash-frozen.

Generation and comparisons. The fixed Qwen3-8B MLX 4-bit checkpoint is used without an adapter, with temperature 0.2, top-p 0.95, thinking disabled and an 8,192-token output cap. The five arms are $000, 100, 010, 110$ and reflection with top-one memory. The latter is a joint content/length intervention, not an equal-length semantic control. A single seed per task is shared across arms; five cyclic orders balance execution position. Router costs are taken from its selected arm, not the cost of collecting all experimental counterfactuals; lexical routing overhead is not separately timed.

The two prespecified comparisons are router minus development-selected fixed 110, and top-one minus top-three memory. We use 20,000 whole-contest bootstrap draws for task-weighted mean differences and pointwise intervals, together with cluster sign-flip tests and Holm correction across these two comparisons. Cluster sign symmetry is an assumption. A four-configuration oracle is

reported only as an outcome-informed upper bound. One seed and 169 tasks do not guarantee detection of a 10 pp effect or establish stability across models.

Failure and stopping rules. Each request has one generation and one grading attempt, with durable started markers preventing silent retries. Evaluation errors count as zero for the operational fixed-denominator endpoint and are reported separately from generation failures. Three consecutive infrastructure failures halt the run. No new request starts after September 15, 2026 at 22:00 Asia/Shanghai; incomplete runs are disclosed as incomplete, without presenting partial comparisons as final. No threshold changes, regeneration or outcome-based task exclusions are permitted after freezing.

F Memory Applicability: Completed Prospective Evaluation

Status and scope. The prospective four-arm study completed all 1,704 prespecified task–seed evaluations on 142 LiveCodeBench-Pro tasks (139 families, 24 contests), using three paired seeds per task. All 142 task reviews against 60 source cards were completed before generation; 83 tasks selected at least one card and 59 selected none. Selection was assistant-authored and non-independent, with prior engineering-answer exposure disclosed. Review completeness is not independent semantic validation.

The source rule need not be the target rule. Review records identify a concrete transferable operation, the prerequisites evidenced in the task and source, and what must not transfer. Source experiences that did not pass their public evaluation are eligible only for applicable failure warnings. A successful source evaluation is not a proof of algorithmic correctness. Selection may cover an auxiliary step without solving the whole target task; unsupported applicability leads to abstention. Up to three eligible cards are ranked by core-goal coverage and then fixed memory ID. Reviewer explanations are kept out of solver prompts. Thus the solver receives the shared source card, not a newly authored target-specific solution, and may still misuse material outside the annotated scope.

Table 6: Completed applicability comparison. Reflection and base model are fixed; no weight update is used. Only C and D are designed to have identical full prompt token counts per task.

Arm	Memory choice	Presentation
A	None	Reflection only
B	Lexical top three	Original summaries
C	Same choices as B	Three standardized cards
D	Assisted applicability, up to three	Same cards; empty slots allowed

Each standardized slot has 192 tokens, preserving the complete source summary and its feedback, with marked padding. C and D share the card library and slot structure. D–C tests the assisted selection procedure within that format; D–A tests its operational value over reflection alone, including the added memory prompt. C–B describes the bundled change in formatting, feedback presentation, padding, and length, not a pure formatting effect. D–B combines selection and presentation changes.

A development check that requires these controls. We ran one old development task, `abc381_b`, once per arm with a paired seed and a pinned local Qwen3-8B checkpoint. A, C, and D each passed the task's 43 tests. B failed the third test, `zzzzzz`, returning `Yes` instead of `No`; evaluation stopped at that failure. Its generated predicate rejected odd frequencies rather than requiring each occurring character to appear exactly twice. There were no generation failures, evaluation errors, or retries in these four calls. Input token counts were 1,139 (A), 1,385 (B), and 1,766 (both C and D).

This is one reused task and one seed per arm, not 43 independent tasks. Since C retains B's memory IDs and also passes, and A passes without memory, the observation neither establishes D's superiority nor identifies retrieval as the cause of B's error. The check validates an executable contrast and records a program-level failure; it is not a new-task effect estimate. Cards and selection rules are not revised from these solver outcomes.

Prespecified comparisons and observed results. Task means across three seeds are the analysis units. D–C and D–A form a two-comparison Holm family with exact contest-cluster sign flips and 20,000 contest-bootstrap draws; joint sign symmetry is an assumption, not guaranteed by the design. A and C each pass 21/426 evaluations (4.93%), while B and D each pass 17/426 (3.99%). D–C is -0.94 pp with 95% interval $[-2.82, 0.87]$; D–A is -0.94 pp with interval $[-3.03, 0.70]$. Both Holm-adjusted p-values are 0.9375. D–B is zero descriptively. These results do not establish an assisted-selection benefit; 426 evaluations per arm are not 426 independent tasks.

Generation failures constrain interpretation. There are 598 generation failures (A:140, B:163, C:141, D:154): 594 reach the 8,192-token limit, and four stopped outputs contain syntax errors. A post hoc text audit finds 582 failures with a stripped nonempty line repeated at least ten times. This is a repetition heuristic, not a causal mechanism label. Relaxing the closing-fence requirement yields nonempty parseable syntax in 455 truncated texts, but syntax alone does not establish a complete or correct program. No such outputs were rescored. Original extraction decisions reproduce; the low overall success rate and output degeneration limit conclusions about memory content.

Execution amendment and matched-count controls. All original prompts, choices, seeds, resource limits, and the 72-hour deadline remain fixed. One evaluation produced an ambiguous container exit137 and did so again in an explicitly authorized retry of the unchanged answer. An independently versioned observer then retained Docker exit metadata before cleanup: a further evaluation confirmed `OOMKilled=true`, yielding a memory-limit failure. Original unresolved records were retained with a separate adjudication, rather than overwritten. Subsequent evaluation used this disclosed observer amendment; no model answer was regenerated for the incident. This post-start repair prevents a claim of an entirely unchanged execution protocol.

The precommitted matched-count follow-up completed all 852 additional evaluations, with 17/426 passes for lexical top-k and 13/426 for uniform random-k. The D comparison reuses its 17/426 main-study outcomes. D–lexical is 0.00 pp (95% interval $[-2.07, 1.53]$, Holm $p = 1$); D–random is $+0.94$ pp (interval $[-0.25, 2.44]$, Holm $p = 0.75$). Both tests use the separately prespecified two-comparison Holm family. Counts are task–seed evaluations; inference uses task means and whole-contest resampling. All 2,556 planned evaluations are resolved, with the original incident and adjudication retained. Independent review remains incomplete. No claim of reliable applicability selection follows from these results.