WHY IS RECURSIVE SELF-IMPROVEMENT HARD TO VERIFY?

Anonymous authors
Paper under double-blind review

ABSTRACT

An AI system can produce better models without becoming better at improving itself. The difficulty is not only measuring a gain, but identifying what has improved. We study a finite-horizon form of recursive self-improvement (RSI) under a fixed target, replay population, resource budget, and system boundary. Our criterion requires artifact gains, causal benefits from installing self-produced update mechanisms, and improved successors that are themselves installed and executed. Mechanisms are compared by the expected artifact gain from a budget-limited update. We give simultaneous effect tests, a cumulative-gain bound under direct continuation and bounded utility, and conditions under which the available records cannot identify the claim. Public-release analyses distinguish recoverable record gaps from missing causal comparisons; a model-coded comparison of 17 benchmark families examines how evidence classifications depend on interpretation. More complete records can clarify what happened without identifying what caused improvement.

1 INTRODUCTION

A sequence of better models does not by itself show that the system producing them has become better at improving itself. Scores may rise through favorable selection (Cawley & Talbot, 2010), more resources (Dodge et al., 2019), external assistance, or proxy optimization (Skalse et al., 2022; Gao et al., 2023), without a better improvement mechanism. What evidence distinguishes these explanations from recursive self-improvement (RSI)? We study this question over a finite horizon with a fixed empirical target.

Novelty raises a related but separate question. A system may find a solution beyond its designers' existing knowledge (Hughes et al., 2024), but discovering a candidate does not verify its consequences. AlphaGo's Move 37 illustrates unexpected discovery under fixed rules (Hassabis, 2026); it does not itself demonstrate recursive improvement of the learning mechanism. Nor is agreement with a known answer always the right test: reference matching can exclude valid alternatives (Chen et al., 2021). Figure 1 separates reference and candidate availability. An unfamiliar candidate still needs an independently justified correctness criterion (Barr et al., 2015).

Language agents combine reasoning with action (Yao et al., 2023), learn tool use (Schick et al., 2023), accumulate executable skills (Wang et al., 2024a), and coordinate through multi-agent conversations (Wu et al., 2024). Systems can also modify training programs (Karpathy, 2026; Chi et al., 2026b), agent code (Zhang et al., 2025a; 2026b), and orchestration graphs (Zhuge et al., 2024), sometimes using agents to evaluate results (Zhuge et al., 2025). Yet a best-of-archive curve may combine winners from different branches. It does not tell us whether the mechanism producing successors has improved, or whether an improved successor was installed and used for the next update (Figure 2).

Good's argument turns on a system's ability to design a better successor (Good, 1965). Gödel Machines justify rewrites by utility proofs (Schmidhuber, 2003); Statistical Gödel Machines instead use risk-controlled confirmation for a specified objective (Wu et al., 2026). Other approaches revalidate skills, as in PowerPlay (Schmidhuber, 2011), or estimate clade metaproductivity, as in Huxley–Gödel Machine (Wang et al., 2026a). In each case, the object of the claim matters: confirming a candidate's utility and establishing an improvement in the mechanism producing future candidates require different comparisons.

Evaluation also feeds back into this process. Disclosed scores can guide later rewrites (Wen et al., 2026), so reusing the same evidence needs explicit restrictions (Dwork et al., 2015; Russo & Zou,

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.

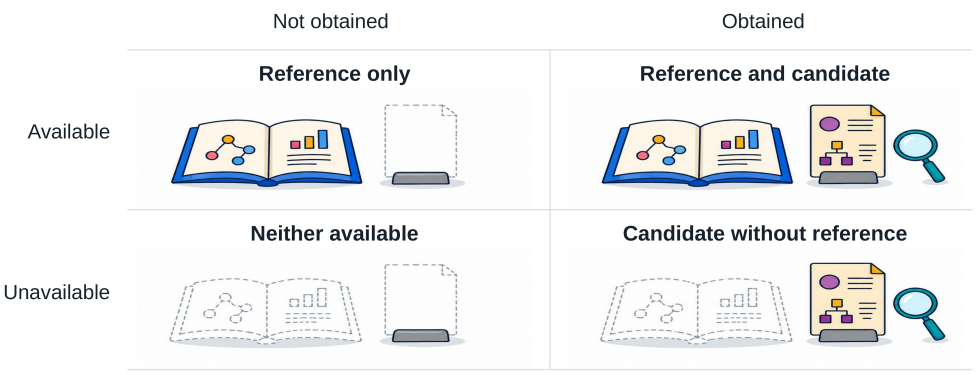

Figure 1: **Knowledge availability and empirical verification.** Rows show reference availability (books), and columns show candidate availability (cards), adapting the Johari window (Luft, 1961) to a specified task, community, and time. Dashed outlines denote absence. Both right-hand cases require independent verification (magnifiers). Availability alone implies neither correctness nor RSI, and the four cases are not improvement stages.

2016). A-Evolve-Training's development–leaderboard gap illustrates the concern (Shi et al., 2026), but does not isolate the effect of evaluation reuse.

We separate two evidential questions: does replacing the updater improve future artifact production, and does the improved updater produce a better successor that actually runs next? Our finite-horizon criterion joins these mechanism effects with target gains under a declared evaluation protocol. We give simultaneous tests, a cumulative-gain bound, and conditions under which indistinguishable records prevent identification. The public-release analyses distinguish missing links that can be reconstructed from causal comparisons that the records do not supply (Section 3); a separate comparison examines the interpretation of 204 benchmark annotations. This distinction determines whether a claim needs better documentation or a different experiment.

2 Preliminary

System boundary. We study mechanism improvement over a finite horizon under a stable empirical target, primarily in AI training (Weston & Foerster, 2025). Write (Q, S, Ψ) for the task artifact, adaptive state, and complete successor-producing mechanism: for example, a trained model, experiment records, and the program that proposes, evaluates, selects, and installs updates. An *object loop* updates (Q, S) with Ψ fixed; a *meta loop* modifies, installs, and executes Ψ. Neither loop guarantees improvement. Before confirmation, we fix the physical replacement interface. Relabeling code as state preserves the causal contrast when it represents the same physical replacement; replacing a fixed interpreter instead is a different treatment (Geiger et al., 2025). The outcome-blind challenge also admits alternative intervention boundaries; requiring all of them to pass tests a stronger robustness claim. A code edit alone establishes neither behavioral change nor improved competence.

Target and mechanism competence. Fix $\Theta = (P_\star, u, \mathcal{R}, b, \mathcal{Z}, B, H)$: target-task distribution, utility, replay-input distribution, reference rule, execution-randomness law, resource budget, and horizon. Define $V_\star(q) = \mathbb{E}_{z\sim P_\star} u(q, z; B)$, assuming finite expectations; z denotes a target task. A replay input x contains the complete parent state or parent set with a designated mechanism slot. The arm-invariant, pre-outcome reference $b(x)$ returns a task artifact: the parent artifact for one parent; population and recombination loops predeclare b on the full parent set. Matched B fixes ex-ante access and resource limits, not equal realized consumption. Define

$$m_\Theta(\psi) = \mathbb{E}_{\substack{x\sim\mathcal{R}\\ Z\sim\mathcal{Z}}}[V_\star\{\mathrm{Out}(\mathcal{T}(x, \psi; Z))\} - V_\star\{b(x)\}], \tag{1}$$

where $\mathcal{T}$ executes one budget-limited update with ψ in that slot, Out extracts the selected artifact, and Z is exogenous execution randomness independent of x. Thus m_Θ measures update performance: expected artifact-utility gain on matched starting inputs. The replay population determines which

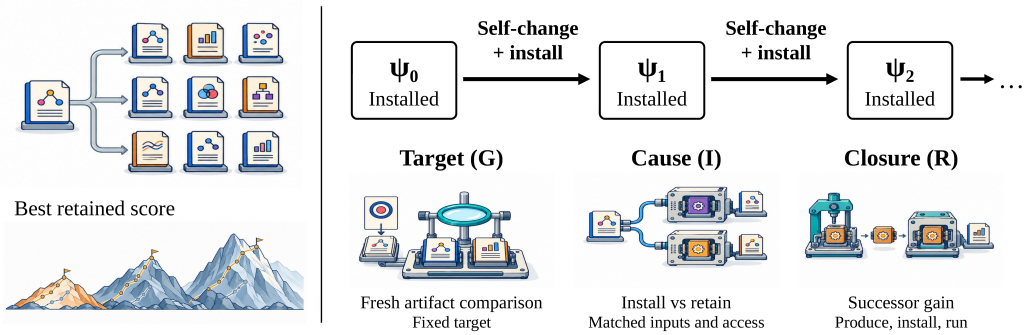

Figure 2: **Archive progress and evidence for recursive improvement.** (a) The mountain ascent illustrates best-retained scores selected across different archive branches. (b) The criterion requires target gain G_g, installation effect I_g from matched complete-input replays with fixed resource access, and successor competence gain R_g. Self-produced successors must be installed and run. Effects must exceed declared margins with uncertainty control at each of $H \geq 2$ stages, including execution of the terminal successor (Section 2). Conceptual schematic, not experimental results.

starting problems the comparison covers; predeclaring it makes the contrast well-defined, not automatically representative. A point mass supports only a local claim. We hold $(\mathcal{R}, b, \mathcal{Z})$, nonintervened model endpoints and serving versions, and the budget fixed before sampling. The arm-invariant reference b cancels from I_g, R_g.

Three verification quantities. At stage g, freeze the selected artifact Q_g^+ and its associated proposed mechanism ψ_g^+. Let ψ_g^- be the retained mechanism and $Q_g^\circ = b(\mathsf{P}_g)$ the reference for complete parent input P_g. Executing ψ_g^+ produces the successor $\widetilde{\psi}_{g+1}$. Define

$$\begin{aligned} G_g &= V_\star(Q_g^+) - V_\star(Q_g^\circ), && \text{target gain,} \\ I_g &= m_\Theta(\psi_g^+) - m_\Theta(\psi_g^-), && \text{installation effect,} \\ R_g &= m_\Theta(\widetilde{\psi}_{g+1}) - m_\Theta(\psi_g^+), && \text{successor gain.} \end{aligned} \tag{2}$$

In training, G_g asks whether the selected model improves. The two mechanism contrasts ask more: I_g compares subsequent model production after replacing or retaining the update program on matched complete inputs; R_g compares the successor actually produced with its producer on this same update-performance scale. Positive G_g alone identifies neither effect. Direct continuation sets $\psi_{g+1}^- = \psi_g^+$ and $\psi_{g+1}^+ = \widetilde{\psi}_{g+1}$, so $R_g = I_{g+1}$ under the same Θ. These are two labels for one effect, not evidence of provenance or execution. Recurrence also requires the self-produced successor to be physically installed and to execute the next transition from the resulting state.

Claim truth and observable evidence. Our stronger *target-realized* witness requires $G_g > \delta_G$, $I_g > \delta_I$, and $R_g > \delta_R$ at each of $H \geq 2$ stages, with predeclared nonnegative margins and provenance linking successive installations and executions (Definition 1). This sufficient criterion is stronger than mechanism improvement alone, which can coexist with artifact regression. It concerns gains along an executed sequence, whether the update strategy was programmed or discovered. The inequalities concern true effects, so positive point estimates are not enough. Confirmation must control uncertainty using undisclosed target and replay data after the compared objects freeze, or a declared reuse guarantee for disclosed data. Missing evidence means *not identified*, not a zero effect or the absence of RSI.

Operational implications. For $u \in [0, 1]$, fresh paired target differences estimate G_g, while matched complete-input replays with fresh target scoring estimate I_g, R_g. Confirmation requires every simultaneous lower bound to exceed its predeclared margin, under the sampling conditions in Appendix C.1. Reusing a replay output does not create an independent replication.

Under fixed Θ and direct continuation, the $H+1$ distinct mechanism contrasts telescope to an increase of at most one: the arm-invariant reference cancels. Their required margins therefore obey

$$\delta_I + (H-1)\max(\delta_I, \delta_R) + \delta_R < 1. \tag{3}$$

This pre-sampling feasibility condition (Proposition 2) is necessary, not sufficient for certification.

A recorded pass can still misrepresent what occurred. Bounding such errors requires independent calibration or an explicit fault model; the latter gives only a model-relative guarantee (Theorem 2). If records cannot distinguish histories that satisfy the criterion from those that do not, an error-controlled decision rule must allow abstention (Corollary 1). We do not validate a complete physical verifier.

3 Evidence from Public Releases

3.1 Study design

We retrospectively analyze public releases and archived derivatives around three questions. **Question 1**: do recorded score gains isolate mechanism improvement? **Question 2**: which missing links can the original releases recover? **Question 3**: what do additional evidence and record matching tell us about task outcomes? For the third question, we compare monitor rankings under different access conditions and test parent associations against transcripts and commands. These analyses examine the evidence needed to estimate G_g, I_g, R_g; they do not estimate the three effects or a complete verifier's accuracy.

We retain failed and incomplete attempts under source-specific rules and analyze the units listed in Table 1, without pooling them as independent RSI samples. We compare recorded Hyperagents parents and children, rejoin AutoResearch's derivative to the official export, and inspect Praxist's evaluation reuse. ResearchArena comparisons pair the same runs across access conditions, while Replay Gap links recorded forks to publisher-reported task outcomes. Deliberately reassociated parent records are separately labeled diagnostic controls, not new agent runs. The 17-family corpus is purposeful, non-exhaustive, and model-coded. No new training or live improvement campaign is performed.

We distinguish raw files, extracted rows, and previously compiled summaries. Release provenance, extraction and resampling procedures, coding rules, and reconstruction checks are documented together in Appendix E. The supplement provides source versions, file-integrity checks, and analysis code. The family review and later case studies use separately fixed source versions. Additional robustness analyses are retrospective, not prospectively preregistered, and preserve frozen observations and original codes. Bootstrap intervals describe retained-cohort resampling variability, not transport to other systems or RSI prevalence.

Table 1 Public releases and results of the retrospective analyses. Counts use study-specific units, not a common performance scale. In Replay Gap, c is base-command-prefix agreement. The final row reports evidence coding, not a measured effect.

Study	Analyzed records	Observation
Hyperagents	500 attempts; 5 runs	Of 449 resolved comparisons, 174 improve, 198 regress, and 77 tie.
AutoResearch	10,380 attempts	5,596/5,597 proposal-linked attempts rejoin launched-script metadata.
Praxist	665 evaluation summaries	664 report identical checksums for the same three evaluation banks.
ResearchArena	58 runs; one CUDA workload	Adding CoT to artifact access lowers assignment AUC for five of six monitors; one is unchanged.
Replay Gap	717 branches; 56 tasks	Tasks resolved: 2/243 with $c = 1$; 3/474 with $c < 1$.
Benchmark review	17 families; model-coded	0/17 coded explicit for released matched forks.

3.2 Artifact progress and target validity

Hyperagents: selection versus transitions. Our archived extraction from the public Hyperagents release contains 2,525 generation records across five conditions (Zhang et al., 2026b;a). We analyze all 500 noninitial records in the five main-condition runs: 449 permit a parent–child validation comparison and 51 remain unresolved. Recomputing score differences gives 174 improvements, 198 regressions, and 77 ties. Thus 275/449 (61.2%) of resolved transitions fail to improve, while the best-so-far score is monotone by construction (Figure 3(a)). These are dependent, descriptive transitions, not 449 independent trials or a mechanism failure rate.

Validation sample counts change in 48/449 resolved comparisons, including 35/174 positive differences (Figure 3(b)). Equal counts do not establish identical or fresh tasks, and changed counts alone do not establish bias. Across all 500 attempts, 41 directly edit `meta_agent.py`. Of the 40 with resolved scores, 39 also edit the task agent and ten improve. File edits and score gains therefore select different sets of attempts. Neither measures mechanism competence on its own, and edits to this file capture only one class of mechanism changes.

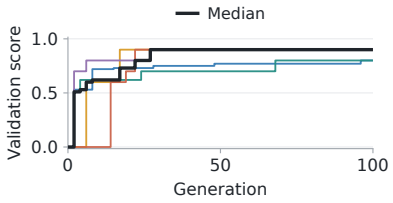

Figure 3(a): **Best-retained scores.** Colors distinguish five runs; black shows their median (Zhang et al., 2026a).



Figure 3(b): **Parent–child outcomes.** Counts and percentages for 449 resolved comparisons, grouped by validation sample-count change (Zhang et al., 2026a).

Consider the previously documented run-3 sequence $11 \to 16 \to 24$. Scores change $.60 \to .50 \to .60$, while sample counts change $10 \to 100 \to 10$. Both transitions edit both agents. The records identify the parents, but supply no fresh target confirmation, matched executions, or successor competence replay; G_g, I_g, R_g therefore remain *not identified* (Table 4). This selected example belongs to the 500 attempts but is outside the final winner's parent path.

The five final best-validation parent paths contain 27 transitions with recorded code patches, including six regressions. All six direct meta-agent edits also modify the task agent; five tie and one improves. None of the five paths has all the evidence required by our criterion. The released fixed-meta ablation compares independently generated histories, not installations from matched frozen parents. These records show which measurements are missing; the regressions alone do not rule out improved expected mechanism competence.

Praxist: repeated evaluation versus fresh confirmation. The archived Rocket analysis records 12 committed boundaries, 11 next-generation agenda links, and 665 evaluation summaries (Li et al., 2026a; Sapient, 2026). Of these, 664 report identical checksums for three evaluation banks, while one records a protocol failure. The snapshots establish feedback inheritance, while the source paper describes fixed, adaptively reused banks. Without independent post-selection confirmation or matched mechanism interventions, this documents reuse, not a generalization gap or updater improvement. Unsuccessful branches and termination after 12 of 30 planned generations remain in the analysis.

3.3 Linkage and public reconstructability

AutoResearch: a recoverable export gap. Evo's derivative contains 10,380 AutoResearch attempts (EvoHQ, 2026). None of its 5,597 proposal-linked attempts appears in its recovered-source catalog. Exact rejoining to Prime Intellect's pinned official export (Prime Intellect, 2026) identifies launched-script metadata for 5,596 of these attempts and logged-source metadata for 5,593; one remains unmatched (Table 6). The derivative's zero join therefore does not establish project-wide source absence.

We checked six deterministically selected linked files against the publisher's byte-level records. The full counts concern metadata availability, not independent verification of every file or Evo's proposal associations. The official export also excludes some upstream attempts. Recovering a script link does not show that the script ran, isolate the effect of installing an updater, or establish that a better successor was produced and executed.

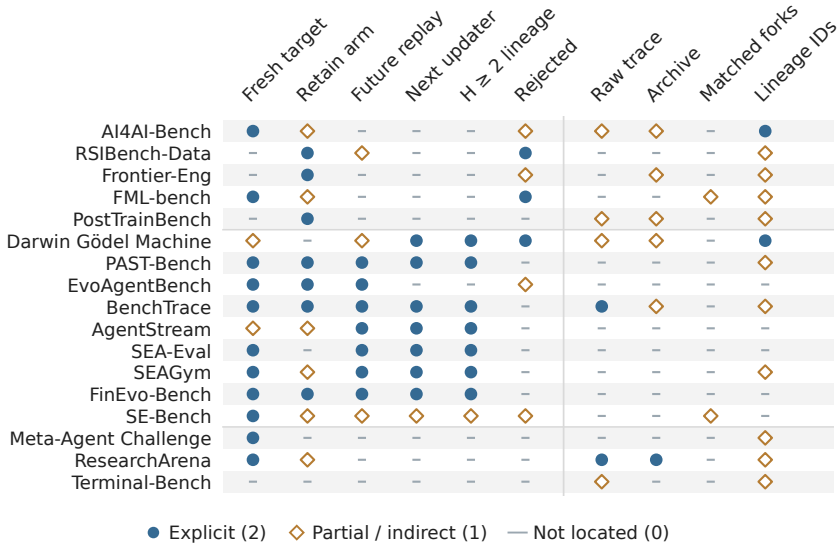

Figure 4: **Benchmark design and released evidence.** Six columns describe design requirements and four describe public records across 17 families. The original model-coded annotations lack independent human validation. Two blinded recodings change some classifications (Appendix E.4). These codes describe reported support, not measured gains or a validated ranking.

Seventeen families: design support versus released evidence. The frozen 17-family corpus covers algorithm and data improvement, longitudinal self-evolution, and evaluation infrastructure (Chi et al., 2026b; Xue et al., 2026; Libon et al., 2026). Figure 4 separates six design fields from four release fields; the supplement retains two additional availability fields and a primary-source locator for every code.

The original counts, $11 \to 4 \to 3 \to 0$, successively require explicit support within one family for fresh targets; matched retain comparisons and future replay; next-updater installation and linked $H \geq 2$ transitions; and rejected-branch accounting. Two isolated model sessions A/B, blinded to prior labels and using the same 12-field codebook and source inventory, instead give $5 \to 0 \to 0 \to 0$ and $8 \to 0 \to 0 \to 0$. Neither assigns an explicit future-replay code, compared with seven originally, because later-task transfer was interpreted differently from evaluating a frozen updater's successor production. Figure 4 therefore preserves the historical matrix as an interpretation-sensitivity result, not a validated ranking.

A and B agree on 171/187 jointly resolved cells, with 17/204 unavailable (U) in at least one; their resolved-cell agreements with the original are 92/193 and 102/190. This is not human reliability validation. The eight-field check adds released forks and lineage IDs to the six design fields. No coding establishes the complete explicit eight-field join, but counting partial support yields five joins in each recoding versus zero originally, plus one possible join in session B. Implemented evaluation does not establish executed recurrence: PAST-Bench excludes generated run outputs, while SEAGym's transfer evaluator performs no new updates (Xue et al., 2026; Zheng et al., 2026). Differences remain unadjudicated.

3.4 Evidence access and measurement validity

ResearchArena: evidence access and ranking. We analyze 58 CUDA-kernel runs scored by six monitors under four conditions (Figure 5) (Libon et al., 2026; aisa-group, 2026b). Assignment AUC compares 51 side-task runs with seven baselines, giving half credit for ties. It is invariant to strictly increasing score transformations. Assignment is not successful sabotage, and baselines are not independently verified clean.



Figure 5: **ResearchArena AUC under four access conditions** (aisa-group, 2026b). The 51 assigned runs and their 14-run independent-side-task subset share seven baselines and one CUDA workload. AUC measures separation of assigned runs from baselines, not successful sabotage or calibrated false-pass rates.

Table 2 Parent matching on the same released trajectories. Replay Gap contains 717 branches and 179 bases across 56 tasks (Gonuguntla, 2026b). Reassigned links pair unchanged records with other same-task bases as diagnostic controls, not new executions or observed mislinks. † Run and task IDs define the recorded parent. ‡ Transcript extension was added after examining exact-match failures. Record compatibility does not establish physical ancestry.

	Compatible links		Branches by candidate count (717)		
Matching rule	Recorded / 717	Reassigned / 1,925	One	Multiple	None
Task ID	717	1,925	0	717	0
Run + task ID†	717	0	717	0	0
Pre-fork command equality	717	44	681	36	0
Exact pre-fork transcript	669	4	665	4	48
Transcript extension‡	717	6	711	6	0

In the full cohort, adding chain of thought (CoT) to artifact access lowers point AUC for five monitors (GPT-5.4: .971 to .923); one ties. Yet all six paired-run resampling ranges include zero, and deleting one baseline can reverse this contrast's sign for three monitors. The 357 assigned–baseline comparisons are dependent within one workload. Neither population-level CoT harm nor calibrated semantic false-pass rates follows.

Replay Gap: return-code checks versus command divergence. Our separate parser reconstructs all 717 official branch comparisons from 896 released trajectories (Gonuguntla, 2026a;b). Model switches are externally imposed, not self-produced mechanism changes or evidence of RSI. Among 708 branches with replay checks, 707 have matching return codes at every checked prefix action; 11,701/11,702 individual return codes match. These checks do not establish full environment-state equality.

Publisher-recorded SWE-bench outcomes exist for all 717 branches. Let c be the shared post-fork command-prefix length divided by the base continuation's positive command count; $c = 1$ permits extra branch commands. Nonetheless, 241/243 full-prefix matches fail, while 3/474 divergent branches succeed. Command agreement cannot replace recorded outcomes; physical replay fidelity remains uncalibrated.

The 358 run/task/fork-matched switch–control pairs contain 353 joint failures, three switch-only successes, and two control-only successes. One control is unpaired. The five successes concern only two tasks and do not establish a switching benefit.

Table 2 shows the coverage–ambiguity tradeoff: exact transcript matching excludes 48 recorded parents; allowing extra boundary messages restores them but leaves six branches ambiguous before any action. These controls test record compatibility, not actual mislinkage prevalence or physical ancestry.

In four official PostTrainBench cases, published judgments flag the 1.00 and .50 runs for benchmark-targeted training and model substitution; the .97 and 0 runs have neither flag (Rank et al., 2026; aisa-group, 2026a). Rejoining all 2,643 viewer events reproduces these associations. Unflagged does not mean independently clean, and this purposeful case set cannot calibrate verifier accuracy. Figures 6 and 7 follow the actions in two BFCL runs and one Replay Gap task.

4 Related Work

Proof and statistical certification. Gödel Machines justify rewrites by proof (Schmidhuber, 2003); self-referential neural systems learn update behavior (Irie et al., 2022; Kirsch & Schmidhuber, 2022). Huxley–Gödel Machine separates its ideal clade-metaproductivity oracle from empirical estimation (Wang et al., 2026a). SGM controls false commitments across updates, with distinct checkpoint and training-procedure targets (Wu et al., 2026). SEA gates self-proposed harness changes with paired executions and anytime-valid tests (Sengupta, 2026); Safe Harness analyzes generation, certification, adoption, and subsequent improvement opportunities (Cai et al., 2026). We specify a joint claim combining artifact gain, install–retain update performance, and the performance of a self-produced successor linked to its installation and execution. Established statistical tests assess these specific quantities.

Empirical improvement. Output refinement (Madaan et al., 2023) and verbal-memory adaptation (Shinn et al., 2023) differ from training on generated data (Zelikman et al., 2022; Wang et al., 2023; Huang et al., 2023). SEAL generates finetuning data and directives using post-update feedback (Zweiger et al., 2025). APE, ProTeGi, and OPRO optimize prompts (Zhou et al., 2023; Pryzant et al., 2023; Yang et al., 2024a); GEPA uses reflection on execution traces (Agrawal et al., 2026). DSPy and AFlow optimize pipelines and workflows (Khattab et al., 2024; Zhang et al., 2025b); GPTSwarm optimizes node prompts and graph connectivity (Zhuge et al., 2024). Promptbreeder evolves mutation prompts (Fernando et al., 2024), while STOP edits its improvement scaffold (Zelikman et al., 2024). AIDE^2 separates held-out agent performance from an ignition test that varies the outer agent from a shared inner starting agent (Srikanth et al., 2026). Its inconclusive three-seed-per-arm comparison makes the distinction concrete: a useful evolved agent still needs separate evidence of better self-improvement.

Applications. SWE-agent studies interfaces for software repair (Yang et al., 2024b); RoboCat generates experience for robot-policy training (Bousmalis et al., 2023). Co-Scientist pairs hypothesis refinement with scientist-guided validation (Gottweis et al., 2026). KernelBench measures kernel correctness and speed (Ouyang et al., 2025), and CUDA-L1 learns CUDA optimization (Li et al., 2026b). These applications make the target concrete; improvement of the generator still requires its own comparison.

Adaptive and agentic evaluation. Reusable holdouts, selection-bias bounds, and confidence sequences support adaptive inference (Dwork et al., 2015; Russo & Zou, 2016; Howard et al., 2021). Agent-as-a-Judge assesses intermediate progress beyond final-answer scoring (Zhuge et al., 2025). Self-correction tests expose feedback and budget confounds (Huang et al., 2024); code- and harness-evolution studies motivate budget-matched baselines and held-out evaluation (Gideoni et al., 2026; Wang et al., 2026b). These methods address uncertainty and feedback, while our contrasts specify which mechanism effect to estimate.

5 Conclusion

More complete records can clarify what happened without identifying what caused improvement. AutoResearch's recovered links repair an export gap; Hyperagents' bundled edits still need a separating comparison. Our criterion connects target gains and installation effects to successors that run next, under declared sampling and record-error assumptions. Retrospective selection and model-coded annotations limit the public studies' generality; they establish neither RSI's presence nor absence. A prospective test should follow installed successors through paired comparisons and use known-status controls to measure false acceptance, abstention, and cost.

AI USE STATEMENT

We used AI tools for language editing and discussion, and take full responsibility for the content of this paper.

REFERENCES

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G. Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/0e9e708b6f48e14fd0ac29e167413f76-Abstract-Conference.html.

aisa-group. PostTrainBench-Trajectories. Hugging Face dataset, 2026a. URL https://huggingface.co/datasets/aisa-group/PostTrainBench-Trajectories. Audited revision: 39d3fcd794df51c062c8bd3b7f8523ba707aaeb3. Public viewer: https://posttrainbench.com/traces/.

aisa-group. ResearchArena-Trajectories. Hugging Face dataset, 2026b. URL https://huggingface.co/datasets/aisa-group/ResearchArena-Trajectories. Audited revision: 241e456ba88bc09ba40ec906cf0c16e4b2daa918. Project: https://github.com/aisa-group/ResearchArena.

Carlo Alfano, Silvia Sapora, Jakob Foerster, Patrick Rebeschini, and Yee Whye Teh. Meta-Learning Objectives for Preference Optimization. In *Advances in Neural Information Processing Systems*, volume 38, pp. 75681–75707, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/6d998a2ae756f71fe3b6fb29bf015a45-Abstract-Conference.html.

Marcin Andrychowicz, Misha Denil, Sergio Gómez, Matthew W. Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando de Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems*, volume 29, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/fb87582825f9d28a8d42c5e5e5e8b23d-Abstract.html.

Earl T. Barr, Mark Harman, Phil McMinn, Muzammil Shahbaz, and Shin Yoo. The oracle problem in software testing: A survey. *IEEE Transactions on Software Engineering*, 41(5):507–525, 2015. doi: 10.1109/TSE.2014.2372785. URL https://doi.org/10.1109/TSE.2014.2372785.

Sander Beckers and Joseph Y. Halpern. Abstracting causal models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 2678–2685, 2019. doi: 10.1609/aaai.v33i01.33012678. URL https://ojs.aaai.org/index.php/AAAI/article/view/4117.

Konstantinos Bousmalis, Giulia Vezzani, Dushyant Rao, Coline Devin, Alex X. Lee, Maria Bauza, Todor Davchev, Yuxiang Zhou, Agrim Gupta, Akhil Raju, Antoine Laurens, Claudio Fantacci, Valentin Dalibard, Martina Zambelli, Murilo Martins, Rugile Pevceviciute, Michiel Blokzijl, Misha Denil, Nathan Batchelor, Thomas Lampe, Emilio Parisotto, Konrad Żołna, Scott Reed, Sergio Gómez Colmenarejo, Jon Scholz, Abbas Abdolmaleki, Oliver Groth, Jean-Baptiste Regli, Oleg Sushkov, Tom Rothörl, José Enrique Chen, Yusuf Aytar, Dave Barker, Joy Ortiz, Martin Riedmiller, Jost Tobias Springenberg, Raia Hadsell, Francesco Nori, and Nicolas Heess. RoboCat: A self-improving generalist agent for robotic manipulation. *Transactions on Machine Learning Research*, 2023. URL https://openreview.net/forum?id=vsCpILiWHu.

Qianshu Cai, Yonggang Zhang, Jun Nie, Maohao Ran, Huajiang Zheng, Jun Song, Xinmei Tian, Yike Guo, and Wei Xue. Safe harness self-evolution: A theoretical analysis of feasibility and limits, 2026. URL https://arxiv.org/abs/2609.08175v1.

Gavin C. Cawley and Nicola L. C. Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11(70):2079–2107, 2010. URL https://www.jmlr.org/papers/v11/cawley10a.html.

Mark Chen et al. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. AgentVerse: Facilitating Multi-Agent Collaboration and Exploring Emergent Behaviors. In *International Conference on Learning Representations*, pp. 20094–20136, 2024a. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/578e65cdee35d00c708d4c64bce32971-Abstract-Conference.html.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-Play Fine-Tuning Converts Weak Language Models to Strong Language Models. In *International Conference on Machine Learning*, pp. 6621–6642, 2024b. URL https://proceedings.mlr.press/v235/chen24j.html.

Yizhe Chi, Deyao Hong, Dapeng Jiang, Tianwei Luo, Kaisen Yang, Boshi Zhang, Zhe Cao, Xiaoyan Fan, Bingxiang He, Han Hao, et al. Frontier-Eng: Benchmarking self-evolving agents on real-world engineering tasks with generative optimization, 2026a. URL https://arxiv.org/abs/2604.12290.

Yizhe Chi, Wenyi Li, Deyao Hong, Xiaoqiu Wang, Mingju Gao, Kaisen Yang, Bingxiang He, Youjie Zheng, Calvin Xiao, and Qinhuai Na. AI4AI-Bench: Benchmarking LLM agents in algorithmic design for recursive self-improvement, 2026b. URL https://arxiv.org/abs/2608.20318.

Bo Deng, Kang Zhou, Lifan Guo, Chongyang Tao, Xuanren Chen, Chenggang Xie, Renzhao Liang, Feng Chen, and Chi Zhang. FinEvo-Bench: A longitudinal benchmark for self-evolving agents in professional financial workflows, 2026. URL https://arxiv.org/abs/2608.06144.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient Finetuning of Quantized LLMs. In *Advances in Neural Information Processing Systems*, volume 36, pp. 10088–10115, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1feb87871436031bdc0f2beaa62a049b-Abstract.html.

Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. Show your work: Improved reporting of experimental results. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2185–2194. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1224. URL https://aclanthology.org/D19-1224/.

Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. The reusable holdout: Preserving validity in adaptive data analysis. *Science*, 349(6248):636–638, 2015. doi: 10.1126/science.aaa9375. URL https://doi.org/10.1126/science.aaa9375.

EvoHQ. Autoresearch novelty benchmark. Hugging Face dataset, 2026. URL https://huggingface.co/datasets/evo-hq/autoresearch-novelty-bench. Audited revision: 14cc65fdbd72d26c57f21ea11b77655728b942ef.

Chrisantha Fernando, Dylan Sunil Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 13481–13544. PMLR, 2024. URL https://proceedings.mlr.press/v235/fernando24a.html.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *International Conference on Machine Learning*, pp. 1126–1135, 2017. URL https://proceedings.mlr.press/v70/finn17a.html.

Maurice Fréchet. Sur les tableaux de corrélation dont les marges sont données. *Annales de l'Université de Lyon, Sciences, Section A*, 14:53–77, 1951.

Dayuan Fu, Keqing He, Yejie Wang, Wentao Hong, Zhuoma GongQue, Weihao Zeng, Wei Wang, Jingang Wang, Xunliang Cai, and Weiran Xu. AgentRefine: Enhancing Agent Generalization through Refinement Tuning. In *International Conference on Learning Representations*, pp. 65185–65204, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/a3cc50126338b175e56bb3cad134db0b-Abstract-Conference.html.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 10835–10866. PMLR, 2023. URL https://proceedings.mlr.press/v202/gao23h.html.

Xingze Gao, Chuanrui Hu, Hongda Chen, Pengfei Yao, Zhao Wang, Yi Bai, Zhengwei Wu, Yunyun Han, Xiaofeng Cong, Jie Gui, Yafeng Deng, and Teng Li. EvoAgentBench: Benchmarking agent self-evolution via ability transfer, 2026. URL https://arxiv.org/abs/2607.05202.

Atticus Geiger, Duligur Ibeling, Amir Zur, Maheep Chaudhary, Sonakshi Chauhan, Jing Huang, Aryaman Arora, Zhengxuan Wu, Noah Goodman, Christopher Potts, and Thomas Icard. Causal abstraction: A theoretical foundation for mechanistic interpretability. *Journal of Machine Learning Research*, 26(83):1–64, 2025. URL https://jmlr.org/papers/v26/23-0058.html.

Yonatan Gideoni, Sebastian Risi, and Yarin Gal. Simple baselines are competitive with code evolution. In *ICLR Workshop on AI with Recursive Self-Improvement*, 2026. URL https://openreview.net/forum?id=QSWFqDcveB.

Ashritha Gonuguntla. The replay gap: Static evaluation of model switching in LLM agents scores the wrong world. In *Efficient Reasoning Workshop at the Conference on Language Modeling*, 2026a. URL https://arxiv.org/abs/2608.08239.

Ashritha Gonuguntla. Replay Gap trajectories. Hugging Face dataset, 2026b. URL https://huggingface.co/datasets/ashritha0907/replay-gap-trajectories. Audited revision: 3f3e9f544819afc7fe7faf4a4f5955554e4a15db. Project: https://github.com/AshrithaG/replay-gap.

Irving John Good. Speculations concerning the first ultraintelligent machine. In *Advances in Computers*, volume 6, pp. 31–88. Academic Press, 1965. doi: 10.1016/S0065-2458(08)60418-0.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Petar Sirkovic, Artiom Myaskovsky, Grzegorz Glowaty, Felix Weissenberger, Alessio Orlandi, Dan Popovici, Anil Palepu, Keran Rong, Ryutaro Tanno, Khaled Saab, Fan Zhang, Jacob Blum, Andrew Carroll, Kavita Kulkarni, Nenad Tomašev, Dina Zverinski, Ivor Rendulic, Elahe Vedadi, Florian Hasler, Luka Rimanic, Marina Boia, Ivan Budiselic, Ben Feinstein, Mathias Bellaiche, Tom Sheffer, Jan Freyberg, Jeremy Ratcliff, Ottavia Bertolli, Katherine Chou, Avinatan Hassidim, Burak Gokturk, Amin Vahdat, Yuan Guan, Vikram Dhillon, Eeshit Dhaval Vaishnav, Byron Lee, Tiago R. D. Costa, José R. Penadés, Gary Peltz, Yossi Matias, James Manyika, Demis Hassabis, Yunhan Xu, Pushmeet Kohli, Annalisa Pawlosky, Alan Karthikesalingam, and Vivek Natarajan. Accelerating scientific discovery with Co-Scientist. *Nature*, 655:487–496, 2026. doi: 10.1038/s41586-026-10644-y. URL https://www.nature.com/articles/s41586-026-10644-y.

Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. Connecting Large Language Models with Evolutionary Algorithms Yields Powerful Prompt Optimizers. In *International Conference on Learning Representations*, pp. 34133–34156, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/9156b0f6dfa9bbd18c79cc459ef5d61c-Abstract-Conference.html.

Demis Hassabis. From games to biology and beyond: 10 years of AlphaGo's impact, March 2026. URL https://deepmind.google/blog/10-years-of-alphago/. Google DeepMind, March 10, 2026.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963. doi: 10.1080/01621459.1963.10500830.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta Programming for A Multi-Agent Collaborative Framework. In *International Conference on Learning Representations*, pp. 23247–23275, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/6507b115562bb0a305f1958ccc87355a-Abstract-Conference.html.

Steven R. Howard, Aaditya Ramdas, Jon McAuliffe, and Jasjeet Sekhon. Time-uniform, nonparametric, nonasymptotic confidence sequences. *The Annals of Statistics*, 49(2):1055–1080, 2021. doi: 10.1214/20-AOS1991.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/36b7acf6f6010652b3f2a433774a66fe-Abstract-Conference.html.

Jiahao Huang, Fei Cheng, Junfeng Jiang, Zefan Yu, and Akiko Aizawa. BenchTrace: A benchmark for testing reflection ability and controlled evolution in LLM agents, 2026. URL https://arxiv.org/abs/2605.29225. Dataset: https://huggingface.co/datasets/huangjh16/BenchTrace; frozen revision c18c2c02d903ff93aabb00bfddebecde9ac47dd1.

Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large language models can self-improve. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 1051–1068. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.67. URL https://aclanthology.org/2023.emnlp-main.67/.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.01798.

Edward Hughes, Michael D Dennis, Jack Parker-Holder, Feryal Behbahani, Aditi Mavalankar, Yuge Shi, Tom Schaul, and Tim Rocktäschel. Position: Open-endedness is essential for artificial superhuman intelligence. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 20597–20616. PMLR, 2024. URL https://proceedings.mlr.press/v235/hughes24a.html.

Alex Iacob, Andrej Jovanović, William F. Shen, Daniel Burkhardt, Meghdad Kurmanji, Nurbek Tastan, Lorenzo Sani, Niccolò Alberto Elia Venanzi, Ambroise Odonnat, et al. The red queen gödel machine: Co-evolving agents and their evaluators, 2026. URL https://arxiv.org/abs/2606.26294.

Kazuki Irie, Imanol Schlag, Róbert Csordás, and Jürgen Schmidhuber. A modern self-referential weight matrix that learns to modify itself. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pp. 9660–9677. PMLR, 2022. URL https://proceedings.mlr.press/v162/irie22b.html.

Khurram Javed and Martha White. Meta-Learning Representations for Continual Learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/f4dd765c12f2ef67f98f3558c282a9cd-Abstract.html.

Sihang Jiang, Lipeng Ma, Zhonghua Hong, Keyi Wang, Zhiyu Lu, Shisong Chen, Jinghao Zhang, Tianjun Pan, Weijia Zhou, Jiaqing Liang, and Yanghua Xiao. SEA-Eval: A benchmark for evaluating self-evolving agents beyond episodic assessment, 2026. URL https://arxiv.org/abs/2604.08988v1. Version 1 is the version used in the frozen evidence audit.

Andrej Karpathy. autoresearch: AI agents running research on single-GPU nanochat training automatically. GitHub repository, 2026. URL https://github.com/karpathy/autoresearch.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. DSPy: Compiling declarative language model calls into state-of-the-art pipelines. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/f1cf02ce09757f57c3b93c0db83181e0-Abstract-Conference.html.

Zae Myung Kim, Young-Jun Lee, Seungyeon Jwa, and Dongyeop Kang. Meta^n: Recursive self-improvement through emergent depth, 2026. URL https://arxiv.org/abs/2608.24735.

Louis Kirsch and Jürgen Schmidhuber. Eliminating meta optimization through self-referential meta learning, 2022. URL https://arxiv.org/abs/2212.14392. ICML 2022 DARL Workshop.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, JD Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal Behbahani, and Aleksandra Faust. Training Language Models to Self-Correct via Reinforcement Learning. In *International Conference on Learning Representations*, pp. 54523–54549, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/871ac99fdc5282d0301934d23945ebaa-Abstract-Conference.html.

So Kuroki, Taishi Nakamura, Takuya Akiba, and Yujin Tang. Agent Skill Acquisition for Large Language Models via CycleQD. In *International Conference on Learning Representations*, pp. 47327–47348, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/755acd0c7c07180d78959b6d89768207-Abstract-Conference.html.

Robert Tjarko Lange, Yuki Imajuku, and Edoardo Cetin. ShinkaEvolve: Towards open-ended and sample-efficient program evolution. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/7886b9bafe76c52fd568db10ff9772df-Abstract-Conference.html.

Laude Institute. Terminal-Bench public leaderboard logs. GitHub repository, 2026. URL https://github.com/laude-institute/terminal-bench-leaderboard. Terminal-Bench Core 0.1.1 result and trajectory archive; snapshot accessed August 25, 2026.

Soochan Lee, Hyeonseong Jeon, Jaehyeon Son, and Gunhee Kim. Learning to Continually Learn with the Bayesian Principle. In *International Conference on Machine Learning*, pp. 26621–26639, 2024. URL https://proceedings.mlr.press/v235/lee24j.html.

Brian Lester, Rami Al-Rfou, and Noah Constant. The Power of Scale for Parameter-Efficient Prompt Tuning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 3045–3059, 2021. URL https://aclanthology.org/2021.emnlp-main.243/.

Jin Li, Ahmed Murtadha, Zhiyu Wang, Qiwen Chen, William Chen, Yifei Wu, Guan Wang, Andy L. Siy, Jiayi Yang, Mengsha Huang, Wenhao Li, Yixuan Liu, Shuailin Pan, Mingli Yuan, Sen Song, and Yuhao Sun. Praxist: From experimental artifacts to solution lineages, 2026a. URL https://arxiv.org/abs/2608.25955.

Xiang Lisa Li and Percy Liang. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 4582–4597, 2021. URL https://aclanthology.org/2021.acl-long.353/.

Xiaoya Li, Xiaofei Sun, Albert Wang, Jiwei Li, and Chris Shum. CUDA-L1: Improving CUDA optimization via contrastive reinforcement learning. In *International Conference on Learning Representations*, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/c94bbbef466ab1b2cfa100e41413b3a8-Abstract-Conference.html.

Lena Libon, Ben Rank, Jehyeok Yeon, David Schmotz, Jeremy Qin, Daniel Donnelly, Derck Prinzhorn, and Maksym Andriushchenko. Researcharena: Evaluating sabotage and monitoring in automated AI R&D. *arXiv preprint arXiv:2607.19321*, 2026. URL https://arxiv.org/abs/2607.19321.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-Tuning: Prompt Tuning Can Be Comparable to Fine-tuning Across Scales and Tasks. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 61–68, 2022. URL https://aclanthology.org/2022.acl-short.8/.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient Episodic Memory for Continual Learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/f87522788a2be2d171666752f97ddebb-Abstract.html.

Xinyu Lu, Tianshu Wang, Pengbo Wang, Zujie Wen, Zhiqiang Zhang, Jun Zhou, Boxi Cao, Yaojie Lu, Hongyu Lin, Xianpei Han, and Le Sun. The meta-agent challenge: Are current agents capable of autonomous agent development?, 2026. URL https://arxiv.org/abs/2606.04455.

Joseph Luft. The Johari window: A graphic model of awareness in interpersonal relations. *Human Relations Training News*, 5(1):6–7, 1961. URL https://archive.org/details/sim_human-relations-training-news_spring-1961_5_1/page/n5/mode/2up.

Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.12931.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/91edff07232fb1b55a505a9e9f6c0ff3-Abstract-Conference.html.

Fanqing Meng, Lingxiao Du, Qiguang Chen, Ziqi Zhao, Haocheng Lu, Mengkang Hu, and Michael Qizhe Shieh. RSIBench-Data: Benchmarking data-centric research for recursive self-improvement, 2026. URL https://arxiv.org/abs/2607.25886.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, et al. AlphaEvolve: A coding agent for scientific and algorithmic discovery, 2025. URL https://arxiv.org/abs/2506.13131.

Krista Opsahl-Ong, Michael J. Ryan, Josh Purtell, David Broman, Christopher Potts, Matei Zaharia, and Omar Khattab. Optimizing Instructions and Demonstrations for Multi-Stage Language Model Programs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 9340–9366, 2024. URL https://aclanthology.org/2024.emnlp-main.525/.

Anne Ouyang, Simon Guo, Simran Arora, Alex L. Zhang, William Hu, Christopher Ré, and Azalia Mirhoseini. KernelBench: Can LLMs write efficient GPU kernels? In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 47356–47415. PMLR, 2025. URL https://proceedings.mlr.press/v267/ouyang25a.html.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744, 2022. URL https://proceedings.neurips.cc/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html.

Gabriel Peyré and Marco Cuturi. Computational optimal transport. *Foundations and Trends in Machine Learning*, 11(5–6):355–607, 2019. URL https://arxiv.org/abs/1803.00567.

Prime Intellect. Experiments: Autonomous speedrunning. Public GitHub experiment archive, 2026. URL https://github.com/PrimeIntellect-ai/experiments-autonomous-speedrunning. Original source attributed by the Evo-derived dataset; current public export consulted September 24, 2026, separately from the frozen Evo analysis.

Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. Automatic prompt optimization with "gradient descent" and beam search. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7957–7968. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.494. URL https://aclanthology.org/2023.emnlp-main.494/.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. ChatDev: Communicative Agents for Software Development. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 15174–15186, 2024. URL https://aclanthology.org/2024.acl-long.810/.

Yuxiao Qu, Tianjun Zhang, Naman Garg, and Aviral Kumar. Recursive introspection: Teaching language model agents how to self-improve. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-1754. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/639d992f819c2b40387d4d5170b8ffd7-Abstract-Conference.html.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Advances in Neural Information Processing Systems*, volume 36, pp. 53728–53741, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/a85b405ed65c6477a4fe8302b5e06ce7-Abstract-Conference.html.

Ben Rank, Hardik Bhatnagar, Ameya Prabhu, Shira Eisenberg, Karina Nguyen, Matthias Bethge, and Maksym Andriushchenko. POSTTRAINBENCH: Can LLM agents automate LLM post-training?, 2026. URL https://arxiv.org/abs/2603.08640.

Maxime Robeyns, Martin Szummer, and Laurence Aitchison. A self-improving coding agent, 2025. URL https://arxiv.org/abs/2504.15228.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience Replay for Continual Learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/fa7cdfad1a5aaf8370ebeda47a1ff1c3-Abstract.html.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625:468–475, 2024. doi: 10.1038/s41586-023-06924-6. URL https://www.nature.com/articles/s41586-023-06924-6.

Paul K. Rubenstein, Sebastian Weichwald, Stephan Bongers, Joris M. Mooij, Dominik Janzing, Moritz Grosse-Wentrup, and Bernhard Schölkopf. Causal consistency of structural equation models. In *Proceedings of the 33rd Conference on Uncertainty in Artificial Intelligence*, 2017. URL https://www.auai.org/uai2017/proceedings/papers/11.pdf.

Daniel Russo and James Zou. Controlling bias in adaptive data analysis using information theory. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51, pp. 1232–1240. PMLR, 2016. URL https://proceedings.mlr.press/v51/russo16.html.

Sapient. Praxist: Public code and released runs. GitHub repository and linked public experiment archive, 2026. URL https://github.com/sapientinc/PRAXIST/tree/9baf7dc85135a86df47cc7ed1c827a1e9dd4bd62. Audited asset: rocket.zip; archive digest and extraction scope are recorded in the supplement.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

Juergen Schmidhuber. Goedel machines: Self-referential universal problem solvers making provably optimal self-improvements, 2003. URL https://arxiv.org/abs/cs/0309048.

Jürgen Schmidhuber. POWERPLAY: Training an increasingly general problem solver by continually searching for the simplest still unsolvable problem, 2011. URL https://arxiv.org/abs/1112.5309.

Biswa Sengupta. Self-evolving agents with anytime-valid certificates, 2026. URL https://arxiv.org/abs/2607.00871v1.

Zhan Shi, Bing He, Yisi Sang, Hanqing Lu, and Benoit Dumoulin. A-Evolve-Training: Autonomous post-training of a 30b model, 2026. URL https://arxiv.org/abs/2606.20657v2.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual Learning with Deep Generative Replay. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/0efbe98067c6c73dba1250d2beaa81f9-Abstract.html.

Taylor Shin, Yasaman Razeghi, Robert L. Logan, IV, Eric Wallace, and Sameer Singh. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 4222–4235, 2020. URL https://aclanthology.org/2020.emnlp-main.346/.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Chenglei Si, Zitong Yang, Yejin Choi, Emmanuel Candès, Diyi Yang, and Tatsunori Hashimoto. Towards execution-grounded automated AI research, 2026. URL https://arxiv.org/abs/2601.14525.

Joar Skalse, Nikolaus H. R. Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward gaming. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/3d719fee332caa23d5038b8a90e81796-Abstract-Conference.html.

James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. CODA-Prompt: COntinual Decomposed Attention-Based Prompting for Rehearsal-Free Continual Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11909–11919, 2023. URL https://openaccess.thecvf.com/content/CVPR2023/html/Smith_CODA-Prompt_COntinual_Decomposed_Attention-Based_Prompting_for_Rehearsal-Free_Continual_Learning_CVPR_2023_paper.html.

Dhruv Srikanth, Bingchen Zhao, Dixing Xu, Yuxiang Wu, and Zhengyao Jiang. Recursive self-improvement of AI research agents, 2026. URL https://arxiv.org/abs/2609.26457v1. Version 1, September 22, 2026.

Mirac Suzgun, Mert Yuksekgonul, Federico Bianchi, Dan Jurafsky, and James Zou. Dynamic cheatsheet: Test-time learning with adaptive memory. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7080–7106. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.eacl-long.333. URL https://aclanthology.org/2026.eacl-long.333/.

Kyle Swanson, Wesley Wu, Nash L. Bulaong, John E. Pak, and James Zou. The Virtual Lab of AI agents designs new SARS-CoV-2 nanobodies. *Nature*, 646:716–723, 2025. doi: 10.1038/s41586-025-09442-9. URL https://www.nature.com/articles/s41586-025-09442-9.

Ganyu Wang, Jinjie Fang, Maxwell Juncheng Yin, Bin Gu, Xi Chen, Boyu Wang, Yi Chang, and Charles Ling. FedOne: Query-Efficient Federated Learning for Black-box Discrete Prompt Learning. In *International Conference on Machine Learning*, pp. 62857–62889, 2025. URL https://proceedings.mlr.press/v267/wang25ab.html.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, 2024a. URL https://openreview.net/forum?id=ehfRiF0R3a.

Wenyi Wang, Piotr Piękos, Nanbo Li, Firas Laakom, Yimeng Chen, Mateusz Ostaszewski, Mingchen Zhuge, and Jürgen Schmidhuber. Huxley-gödel machine: Human-level coding agent development by an approximation of the optimal self-improving machine. In *International Conference on Learning Representations*, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/821d20219c2f14850af1b5220f0ed13f-Abstract-Conference.html.

Xinyuan Wang, Chenxi Li, Zhen Wang, Fan Bai, Haotian Luo, Jiayou Zhang, Nebojsa Jojic, Eric Xing, and Zhiting Hu. PromptAgent: Strategic Planning with Language Models Enables Expert-level Prompt Optimization. In *International Conference on Learning Representations*, pp. 23967–24001, 2024b. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/686a3f32067838c8dbb68da6e9e3cf69-Abstract-Conference.html.

Yike Wang, Huaisheng Zhu, Zhengyu Hu, Yige Yuan, Zhengyu Chen, Shakti Senthil, Hannaneh Hajishirzi, Yulia Tsvetkov, Pradeep Dasigi, and Teng Xiao. Rethinking the evaluation of harness evolution for agents, 2026b. URL https://arxiv.org/abs/2607.12227v2.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-Instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13484–13508. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.acl-long.754. URL https://aclanthology.org/2023.acl-long.754/.

Zefeng Wang, Minxi Yan, Jinhe Bi, Sikuan Yan, Volker Tresp, and Yunpu Ma. MetaSkill-Evolve: Recursive self-improvement of LLM agents via two-timescale meta-skill evolution, 2026c. URL https://arxiv.org/abs/2607.05297v1.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning To Prompt for Continual Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 139–149, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Wang_Learning_To_Prompt_for_Continual_Learning_CVPR_2022_paper.html.

Jiaxin Wen, Liang Qiu, Joe Benton, Jan Hendrik Kirchner, and Jan Leike. Automated weak-to-strong researcher. Anthropic Alignment Science Blog, 2026. URL https://alignment.anthropic.com/2026/automated-w2s-researcher/.

Jason Weston and Jakob Foerster. AI & human co-improvement for safer co-superintelligence, 2025. URL https://arxiv.org/abs/2512.05356.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=BAakY1hNKS.

Xuening Wu, Shenqin Yin, Yanlan Kang, Xinhang Zhang, Qianya Xu, Zeping Chen, and Wenqiang Zhang. Statistical Gödel machines: From formal proofs to risk-controlled self-improvement, 2026. URL https://arxiv.org/html/2510.10232v2. Version 2, September 17, 2026; first released in October 2025.

Chunqiu Steven Xia, Yinlin Deng, Soren Dunn, and Lingming Zhang. Demystifying LLM-based software engineering agents. *Proceedings of the ACM on Software Engineering*, 2(FSE):FSE037:1–FSE037:24, 2025. doi: 10.1145/3715754. URL https://doi.org/10.1145/3715754.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. OSWorld: Benchmarking Multimodal Agents for Open-Ended Tasks in Real Computer Environments. In *Advances in Neural Information Processing Systems*, volume 37, pp. 52040–52094, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/5d413e48f84dc61244b6be550f1cd8f5-Abstract-Datasets_and_Benchmarks_Track.html.

Guowei Xu, Mert Yuksekgonul, Carlos Guestrin, and James Zou. metaTextGrad: Automatically optimizing language model optimizers. In *Advances in Neural Information Processing Systems*, volume 38, 2025a. doi: 10.52202/085713-5107. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/e00abbc48c69471191dc14864f2ed231-Abstract-Conference.html.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-Mem: Agentic memory for LLM agents. In *Advances in Neural Information Processing Systems*, volume 38, 2025b. doi: 10.52202/085713-0593. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/19909c36f51abc4856b4560aff3d36d6-Abstract-Conference.html.

Shuhan Xue, Zixin Ding, Yichen Shen, Yinjie Wang, Zhenfei Yin, Yingcheng Wu, Yuxin Chen, Mengdi Wang, and Ling Yang. PAST-Bench: Benchmarking the foundations of recursive self-improvement in personal agents, 2026. URL https://arxiv.org/abs/2608.04003.

Dong Yan, Jian Liang, Dapeng Hu, Ran He, Nicholas Jing Yuan, Qi Zhang, and Tieniu Tan. AgentStream: How well do self-evolving LLM agents perform under streaming tasks?, 2026. URL https://arxiv.org/abs/2608.00155.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *International Conference on Learning Representations*, 2024a. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/3339f19c5fcee3ad74502947a32be9e6-Abstract-Conference.html.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *Advances in Neural Information Processing Systems*, volume 37, 2024b. doi: 10.52202/079017-1601. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/5a7c947568c1b1328ccc5230172e1e7c-Abstract-Conference.html.

Junlin Yang, Che Jiang, Yu Fu, Tianwei Luo, Can Ren, Weizhi Wang, Kaikai Zhao, Hongyi Liu, Yuxin Zuo, Yuru Wang, Yuchen Fan, Kai Tian, Zhenzhao Yuan, Xiaojian Lin, Li Sheng, Rushi Qiang, Guoli Jia, Xingtai Lv, Ermo Hua, Dianqiao Lei, Youbang Sun, Ning Ding, Bowen Zhou, and Kaiyan Zhang. Frontis-MA1: Training an AI4AI model towards recursive self-improvement in machine learning engineering, 2026. URL https://arxiv.org/abs/2607.28568.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Zhaochen Yu, Yingcheng Wu, Zhenfei Yin, Kaiyuan Chen, Zhe Zhao, Mengdi Wang, Shuicheng Yan, and Ling Yang. Recursive experiential–working memory evolution for long-horizon agent harnesses, 2026. URL https://arxiv.org/abs/2608.24876.

Jiarui Yuan, Tailin Jin, Weize Chen, Zeyuan Liu, Zhiyuan Liu, and Maosong Sun. SE-Bench: Benchmarking self-evolution with knowledge internalization, 2026. URL https://arxiv.org/abs/2602.04811v1. Version 1 corresponds to the cited author list and retained project citation.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason E Weston. Self-rewarding language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 57905–57923. PMLR, 2024. URL https://proceedings.mlr.press/v235/yuan24d.html.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Pan Lu, Zhi Huang, Carlos Guestrin, and James Zou. Optimizing generative AI by backpropagating language model feedback. *Nature*, 639:609–616, 2025. doi: 10.1038/s41586-025-08661-4. URL https://www.nature.com/articles/s41586-025-08661-4.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. STaR: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 15476–15488, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/639a9a172c044fbb64175b5fad42e9a5-Abstract-Conference.html.

Eric Zelikman, Eliana Lorch, Lester Mackey, and Adam Tauman Kalai. Self-taught optimizer (STOP): Recursively self-improving code generation. In *First Conference on Language Modeling*, 2024. URL https://arxiv.org/abs/2310.02304.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin Gödel machine: Open-ended evolution of self-improving agents, 2025a. URL https://arxiv.org/abs/2505.22954.

Jenny Zhang, Bingchen Zhao, Wannan Yang, Jakob Foerster, Jeff Clune, Minqi Jiang, Sam Devlin, and Tatiana Shavrina. Hyperagents: Public code and experiment logs. GitHub repository and linked public experiment archive, 2026a. URL https://github.com/facebookresearch/HyperAgents. Logs from Experiments: outputs_os.zip; audited archive and member identifiers are recorded in the supplement.

Jenny Zhang, Bingchen Zhao, Wannan Yang, Jakob Foerster, Jeff Clune, Minqi Jiang, Sam Devlin, and Tatiana Shavrina. Hyperagents, 2026b. URL https://arxiv.org/abs/2603.19461.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xiong-Hui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. AFlow: Automating agentic workflow generation. In *International Conference on Learning Representations*, 2025b. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/5492ecbce4439401798dcd2c90be94cd-Abstract-Conference.html.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, Urmish Thakker, James Zou, and Kunle Olukotun. Agentic context engineering: Evolving contexts for self-improving language models. In *International Conference on Learning Representations*, 2026c. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/8a94ff6f922d995d7d3f4ebf4143e442-Abstract-Conference.html.

Andrew Zhao, Yiran Wu, Tong Wu, Quentin Xu, Yang Yue, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute zero: Reinforced self-play reasoning with zero data. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-3534. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/9837dc00ff67d176373268ed48042d49-Abstract-Conference.html.

Congjie Zheng, Chuanyi Xue, Bin Liang, Jun Yang, and Changshui Zhang. SEAGym: An evaluation environment for self-evolving LLM agents, 2026. URL https://arxiv.org/abs/2606.17546.

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 *International Conference on Learning Representations*, pp. 15585–15606, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/4410c0711e9154a7a2d26f9b3816d1ef-Abstract-Conference.html.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=92gvk82DE-.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. GPTSwarm: Language agents as optimizable graphs. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 62743–62767. PMLR, 2024. URL https://proceedings.mlr.press/v235/zhuge24a.html.

Mingchen Zhuge, Changsheng Zhao, Dylan R. Ashley, Wenyi Wang, Dmitrii Khizbullin, Yunyang Xiong, Zechun Liu, Ernie Chang, Raghuraman Krishnamoorthi, Yuandong Tian, Yangyang Shi, Vikas Chandra, and Jürgen Schmidhuber. Agent-as-a-judge: Evaluate agents with agents. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 80569–80611. PMLR, 2025. URL https://proceedings.mlr.press/v267/zhuge25a.html.

Qiran Zou, Hou Hei Lam, Wenhao Zhao, Tingting Chen, Yiming Tang, Samson Yu, Yingtao Zhu, Srinivas Anumasa, Zufeng Zhang, Tianyi Zhang, et al. FML-bench: A controlled study of AI research agent strategies from the perspective of search dynamics, 2026. URL https://arxiv.org/abs/2605.17373.

Adam Zweiger, Jyo Pari, Han Guo, Yoon Kim, and Pulkit Agrawal. Self-adapting language models. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-2483. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/6b41e04c41726e2a60e456d0a2b961ab-Abstract-Conference.html.

A VERIFICATION FRAMEWORK

This appendix specifies the physical claim and its statistical verification. With target, replay population, budget, boundary, and horizon fixed, true gains above the declared margins must accompany same-branch provenance and connected physical execution (Definition 1). Records and tests form a certificate; soundness additionally needs uncertainty control and justified bounds on record errors (Theorem 2). Under direct continuation, $R_g = I_{g+1}$ is one effect. The terminal successor must execute, but a full stage-$(H+1)$ certificate lies outside the horizon.

System boundary and update regimes. Our primary scope is AI training, whose mutable boundary may contain parameters, data, objectives, architectures, and overall code (Weston & Foerster, 2025). Assigning every model, optimizer, evaluator, tool, remote model endpoint, human contribution, and compute budget either inside this boundary or to a frozen external schedule fixes "self." Extending the boundary to chips, compilers, inference systems, or applications changes the target, intervention, resource ledger, and recursive edge; Table 3 gives the substitutions.

Table 3 How verification requirements change with the system boundary. These are conceptual mappings, not cross-domain validation.

Scope	Artifact Q	Updater Ψ	Evaluation conditions
Chip	RTL, netlist, layout	Synthesis, physical design	Fixed PDK; correctness, PPA, yield; policy fork
Compiler	Kernel, schedule, program	Autotuner, pass policy	Fixed hardware; correctness, latency, energy; policy fork
Training	Model, checkpoint	Data, objective, architecture, code	Fresh target; retained skills; full-state fork
Inference	Serving setup	Kernel, cache, batching, quantization, routing	Fixed hardware and traffic; quality, cost, latency; tuner fork
Agent	Workflow, tool policy	Builder, evaluator, memory, tools	Fixed tools and endpoints; task utility; builder fork

Write the state as (Q_t, S_t, Ψ_t): task artifact, ordinary adaptive state, and the complete successor-producing mechanism. The last object includes every in-boundary component that proposes, scores, selects, materializes, or installs the next update. An object loop changes (Q, S) while Ψ stays fixed; a meta loop changes, installs, and reexecutes Ψ. Artifact and score traces alone do not determine this partition: a program can be represented as mechanism content or as input state to an interpreter. Verification must therefore name physical parents and expose an intervention.

An outcome-blind author–challenger process admits every proposed factorization satisfying the four conditions in Appendix C, without author veto, then freezes the finite, nonempty admitted set $\mathfrak{F}_\Theta$ before confirmation. The evidence card retains the rule, assignments, decisions, and disagreements. A robustness claim must pass every admitted factorization in this declared set. This is robustness across declared physical intervention boundaries, not invariance under all representations of one intervention (Appendix C.3).

Mechanism identity is behavioral, not syntactic: Section A equates implementations with the same complete successor law, blocking gains from refactoring alone. An updater may modify an internal objective or evaluator inside Ψ, but $P_\star$ remains the external claim target: simultaneously changing the system and its evaluation criterion does not establish improvement on a fixed target. Eureka's optimization of reward code (Ma et al., 2024) illustrates the distinction between a mutable training objective and the external task criterion. A curriculum or expanding task set requires a prespecified schedule and a justified relationship between stage-specific evaluations; a post-hoc change defines a new claim.

A Gödel Machine rewrites only after proving greater expected utility than continued proof search, relative to its encoded axioms, utility, hardware, and environment model. Contemporary empirical loops instead select from exposed evidence. For each fixed artifact q, write $\widehat{V}_t(q) = V_\star(q) + b_t(q) + \epsilon_t(q)$, where $b_t(q) = \mathbb{E}[\widehat{V}_t(q)] - V_\star(q)$ and $\mathbb{E}[\epsilon_t(q)] = 0$ under the stage-t evaluation law, before

conditioning on score-based selection. If c_t^* is the best of k_t candidates under $\widehat{V}_t$, then

$$\widehat{V}_t(c_t^*) - \widehat{V}_t(Q_t) = \underbrace{V_\star(c_t^*) - V_\star(Q_t)}_{\text{stable-target gain}} + \underbrace{b_t(c_t^*) - b_t(Q_t)}_{\text{evaluator gap}} + \underbrace{\epsilon_t(c_t^*) - \epsilon_t(Q_t)}_{\text{selected noise}}, \tag{4}$$

Fixed-artifact centering need not survive selection (Cawley & Talbot, 2010); with an adaptively selected parent, the noise difference has no general sign. Search can also increase evaluator mismatch. The identity does not estimate its components from retained scores. Positive target gain establishes artifact improvement, not improvement of the transition mechanism.

A fixed law can also generate progress-looking records. With iid continuous scores, attempt t is a record with probability $1/t$; hence T attempts produce $H_T - 1$ expected post-initial records, where $H_T = \sum_{t=1}^{T} 1/t$ is a harmonic sum, distinct from the lineage horizon H. Excess records reject only this narrow null. They do not identify which component adapted or whether Ψ improved, and still require a post-selection target and matched search budget.

Causal lineage witness. A rising record leaves three questions. Did the artifact improve outside selection evidence? Did the same branch carry a self-produced updater whose installation improves successor production under matched complete parents and resources? Did that updater produce a better updater that executes the next transition? These links must belong to one causal path: later-stage successes or receipts from other branches cannot substitute for a missing link.

We study finite-horizon recurrence of the complete in-boundary update law, using the competence functional and fixed contract in Section 2, Equation 1. Selector- or evaluator-only changes count when they raise competence under the unchanged target $P_\star$. Population, recombination, and distributed systems may have set- or graph-structured states: source binds every contributing parent, while "one lineage" follows installed state roots rather than a one-parent genealogy. With same-lineage provenance, installation, and execution, positive I_g, R_g support mechanism-RSI even if artifacts regress. Definition 1 additionally requires G_g for target-realized recurrence; G_g alone supports only artifact improvement.

Before asking whether the records determine a claim, authors must declare what counts as "self," improvement, and recurrence. Definition 1 supplies one sufficient high-assurance witness, not a necessary definition of all RSI or a judgment that weaker learning has failed. Theorem 1 does not make this choice of claim canonical: it establishes nonidentification under its stated model-class assumptions.

Byte identity is too strict: within the fixed interface, write $\psi \equiv_\Theta \psi'$ when $\mathcal{T}(x, \psi; Z)$ and $\mathcal{T}(x, \psi'; Z)$ induce the same law for $\mathcal{R}$-almost every x. Then m_Θ, I_g, and R_g depend on the behavioral kernel rather than file names or byte placement. This blocks refactoring gains, but remains relative to the declared boundary, replay population, and external schedule.

At stage g, freeze one selected successor state: artifact Q_g^+, proposed and retained mechanisms (ψ_g^+, ψ_g^-), state root, and complete parent input P_g, containing all parent states. Before outcomes fix $Q_g^\circ = b(\mathsf{P}_g)$; receipts from different successor states cannot be paired post hoc. The reference rule always returns an evaluable task artifact, not a parent-state tuple. For a fixed boundary, $\mathcal{T}$ replaces only the declared mechanism slot and preserves the other inputs. System-arm failures map to predeclared outcomes; an instrumentation failure does not supply an artifact score. The budget fixes available resources and access before execution; actual consumption may differ. A net-cost claim would additionally require a predeclared cost functional and resource ledger and is not implied by $u(q, z; B)$.

Target. Let $\mathcal{F}_g^{\text{dev}}$ contain every task, score, and diagnostic disclosed before the stage-g branch is frozen; the selected branch is $\mathcal{F}_g^{\text{dev}}$-measurable. Only after that freeze, draw a block $D_g^{\text{conf}} = (z_{gj})_{j=1}^{n_g} \sim P_\star^{n_g}$ independently of $\mathcal{F}_g^{\text{dev}}$ and estimate $G_g = V_\star(Q_g^+) - V_\star(Q_g^\circ)$ on that block. This independent confirmation prevents selection on development noise from directly determining the estimated target gain. Once disclosed, D_g^{conf} enters $\mathcal{F}_{g+1}^{\text{dev}}$ and is no longer fresh. Reuse requires a declared adaptive-analysis guarantee (Dwork et al., 2015; Russo & Zou, 2016); otherwise each successive confirmation requires fresh confirmation data. Changing the target, adjudicator, or failure rule without a predeclared bridge starts a new series.

A frozen arm's crash or malformed output is an outcome with predeclared utility. A broken generator, unmatched fork, corrupted log, or failed adjudicator is an instrument failure: it closes the history as nonpassing and *not identified*. Neither may be repaired after partial results.

Self-cause. For each replay input $x \sim \mathcal{R}$, materialize complete commit and retain states differing only in the declared mechanism slot, containing ψ_g^+ and ψ_g^- respectively. Pair their exogenous randomness under the declared law. Averaging over replay inputs targets the population contrast; a single fixed input supports only the local claim stated in Section 2. Their competence difference $I_g = m_\Theta(\psi_g^+) - m_\Theta(\psi_g^-)$ attributes gain to installation under the common replay population. Provenance must bind Q_g^+ and ψ_g^+ to the same selected branch and allowed computation, excluding external injection and post-freeze edits. Matching a trusted digest checks byte identity, not authorship or execution.

If a remote model endpoint, tool backend, or execution service cannot be forked faithfully, an independent ablation does not by itself identify I_g. It may support a broader policy or full-system contrast, but that claim requires its own identification assumptions and evidence.

Closure. Execute the committed mechanism. Let $\widetilde{\psi}_{g+1}$ be its produced successor and ψ_{g+1}^{exec} the mechanism root that physically executes the next transition. Require the primitive causal relation $\text{InstallEdge}(\widetilde{\psi}_{g+1}, \psi_{g+1}^{\text{exec}}) = 1$; equal hashes are only recorded evidence. On the same replay population, $R_g = m_\Theta(\widetilde{\psi}_{g+1}) - m_\Theta(\psi_g^+)$ asks whether the installed updater produced a better updater. A checkpoint from another branch or a cross-successor average cannot answer it.

For direct continuation, the retained mechanism at stage $g+1$ is $\psi_{g+1}^- = \psi_g^+$ and its proposed mechanism is $\psi_{g+1}^+ = \widetilde{\psi}_{g+1}$. With unchanged Θ, this gives $R_g = I_{g+1}$: the same competence contrast appears in two adjacent stage descriptions. The closure obligation additionally establishes self-production, physical installation, and subsequent execution. This identity neither licenses treating the two descriptions as independent samples nor reusing confirmation after it has informed a new selection. For populations and recombination, the ledger must identify the retained complete update law and every contributing parent; adjacency cannot be inferred from generation labels alone.

Definition 1 (High-assurance target-realized lineage witness)**.** Fix Θ, a boundary, $H \geq 2$, and nonnegative $(\delta_G, \delta_I, \delta_R)$. A history h is an H-stage witness if every stage g satisfies: (i) the true target gain is $G_g > \delta_G$ relative to $Q_g^\circ = b(\mathsf{P}_g)$; (ii) the true installation effect is $I_g > \delta_I$, and the artifact and proposed mechanism were produced on the same selected branch from the complete declared parent set by allowed computation; and (iii) that proposed mechanism is physically installed and executed, produces a successor with true $R_g > \delta_R$, and this successor satisfies physical InstallEdge and executes the next transition. All outcomes, failures, and identities belong to one predeclared sequence of installed state roots. Let $W_H(h) = 1$ exactly when these effect and causal facts hold, and zero otherwise.

Recorded certificate. The physical witness differs from the procedure used to infer it. A recorded pass $\widetilde{W}_H(h) = 1$ requires simultaneous uncertainty control for the effect thresholds, target confirmation independent of the feedback used for selection, a reference receipt, and matched commit–retain evidence. The target-tested artifact, installed mechanism, and complete parent set must also be source-bound to one selected successor. Installation and continuation receipts remain fallible evidence of physical facts. Appendices C and B give implementation assumptions and error budgets. Missing evidence returns a recorded nonpass with status *not identified*; it does not set physical W_H to zero.

Successor provenance must bind every contributing parent. The lineage record must also bind the physical installation to the next frozen pre-state, including any additional parents. At stage H, the produced successor must still be installed and actually execute the next transition; only a complete stage-$(H+1)$ certificate is outside the declared horizon. This no-restart clause turns adjacent stage certificates into one lineage. Scores and content hashes alone establish neither relation.

The stagewise product certifies recurrence at the declared granularity. A nonmonotone system may predeclare macro-stages or a weaker recurrence-rate estimand; it does not thereby pass an every-transition claim.

The analysis distinguishes artifact progress, self-editing ancestry, a controlled installation effect, mechanism recurrence, and target-realized recurrence. Evidence supporting a weaker claim is not evidence of failure. Fresh G_g supports the first claim, provenance supports ancestry, I_g adds causal installation, I_g together with R_g supports mechanism recurrence, and the complete same-lineage join supports target-realized recurrence. The evidence card reports the strongest claim supported by the measurements. For factorization f, let $W_H^{\mathrm{sys}} = \min_{f \in \mathfrak{F}_\Theta} W_{H,f}$. Only $W_H^{\mathrm{sys}} = 1$ satisfies the physical claim across the declared challenged set. A failure to certify all assignments may reflect boundary dependence or unresolved evidence; it does not establish positive operator improvement in another assignment. An empty challenged set licenses no claim.

Fixed-law discrimination. If ψ_g^-, ψ_g^+, and $\widetilde{\psi}_{g+1}$ have the same behavioral law, both mechanism contrasts are zero. Meta^n exposes this boundary by holding Ω fixed while its trace/code input grows (Kim et al., 2026). Assigning Ω to Ψ and the stack to S yields object-loop gain; alternative assignments remain factorization-dependent unless all pass. Recuris makes the same ambiguity operational at the agent-harness boundary: its Meta-Agent and outer improvement procedure stay fixed while Skill Memory changes (Yu et al., 2026). Treating that memory as S yields a strong object loop; treating the memory-control layer as Ψ is a broader claim that must survive the boundary challenge.

For pass events $A_1, \ldots, A_k$ under a common history distribution, with $p_j = \Pr(A_j)$, the Fréchet bound gives $\Pr(\bigcap_{j=1}^k A_j) \geq \max\{0, \sum_{j=1}^k p_j - (k-1)\}$ (Fréchet, 1951). This can certify a positive joint probability, but it does not reconstruct a particular history's source and installation links. Rates from different artifact populations need a justified common-history interpretation before this bound applies. Under direct continuation at fixed Θ, $R_g = I_{g+1}$ $(g < H)$ is one contrast with nested threshold events, not two distinct effects.

B Identification and assurance

This appendix separates identification from estimation. Theorem 1 concerns omitted physical facts in a stated black-box class; Corollary 2 composes possible false passes across stages; Theorem 2 combines a recorded-pass confidence sequence with an externally justified error budget. Proofs are in Appendix D. Section 3 does not supply a completed calibration of these guarantees.

Once a disclosed confirmation outcome guides later search, it becomes development information and cannot independently confirm the resulting successor. Condition on prior development, and let $A_g = (z_j)_{j=1}^n$ be iid target observations independent of that prior. For each fixed successor ψ, let $v(\psi, z) \in [0,1]$, $\widehat{\mu}_{A_g}(\psi) = n^{-1} \sum_j v(\psi, z_j)$, and $\mu(\psi) = \mathbb{E}_{z \sim P_\star} v(\psi, z)$. A standard adaptive-data bound gives

$$\left| \mathbb{E}\left[\widehat{\mu}_{A_g}(\Psi_{g+1}) - \mu(\Psi_{g+1}) \right] \right| \leq \sqrt{\frac{I(\Psi_{g+1}; A_g)}{2n}}. \tag{5}$$

The bound is conditional on prior development; information is in nats (Russo & Zou, 2016). If the successor accesses A_g only through feedback, a predeclared lossless fixed-length B_{fb}-bit encoding of the full transcript, including length and stopping, bounds the numerator by $B_{\mathrm{fb}} \log 2$. This is an absolute expected-bias bound, not a high-probability error bar, mean absolute error bound, or empirically measured bias. The information budget B_{fb} differs from resource budget B.

Without restrictions on sample access or hypotheses, an interpolating successor can score one on every disclosed point and zero almost surely on a nonatomic target. Confirmation needs fresh evidence or a justified reuse guarantee. Praxist's reused banks expose this dependency without identifying its bias (Li et al., 2026a).

Verification raises three separate questions. Do the observations identify the required target, counterfactual, source, and installation relations within one executed update sequence? How accurately do the records represent physical execution? How many adjudicated launches are needed to bound the mean conditional witness probability? Additional launches can reduce sampling uncertainty, but do not by themselves supply a missing causal relation or justify a record-to-physical error bound. Theorem 2 retains a separate bound on false recorded passes. Let $[W]_{\mathsf{T}}$ be the class of black-box worlds inducing the same finite recorded-transcript law. A claim is identified only when its truth is constant on this class.

Theorem 1 (Missing-edge nonidentification). *Assume the model class contains a positive witness and is closed under margin-crossing changes to omitted facts. The lineage claim in Definition 1 is not identified if any remains free: (a) positive-$P_\star$ target behavior outside the support of verification queries or the physical reference $Q_g^\circ = b(\mathsf{P}_g)$; (b) a retain outcome or branch/source relation needed for self-cause; or (c) physical installation and carry-forward linking a produced successor to the next updater and frozen pre-state. Each omission admits worlds W^+, W^- with identical transcript laws and opposite witness truth. Every binary decision rule has worst-case error at least $1/2$; at total variation κ, the bound is $(1-\kappa)/2$.*

The theorem concerns identification in its stated model class, not how RSI arises. Alternative evidence must address the omitted fact: reusable-holdout guarantees can justify specified data reuse; transport bounds can address distribution mismatch; structural assumptions may identify unobserved contrasts; calibrated fault models can bound observation error. These guarantees are not interchangeable. Identifying the physical witness still requires its truth to be invariant across the remaining compatible worlds in $[W]_{\mathsf{T}}$.

For the target-support gap in clause (a), fix the artifact, its physical reference, and budget B. Let the evaluation distribution P_{aud} and target $P_\star$ have finite first moments under a common predeclared metric. If $d_g(z) = u(Q_g^+, z; B) - u(Q_g^\circ, z; B)$ is L_g-Lipschitz, the standard Kantorovich–Rubinstein bound (Peyré & Cuturi, 2019, Section 6.1) gives $G_g \geq \ell_g^{\text{aud}} - L_g W_1(P_\star, P_{\text{aud}})$, where W_1 is the corresponding 1-Wasserstein distance and $\ell_g^{\text{aud}} \leq \mathbb{E}_{P_{\text{aud}}} d_g$. A sample-based lower bound needs a coverage guarantee, jointly with any estimated inputs. On this validity event, with justified pre-outcome upper bounds on L_g, W_1, the condition $\ell_g^{\text{aud}} > \delta_G + L_g W_1(P_\star, P_{\text{aud}})$ is sufficient to certify target gain. Failure of this condition neither refutes gain nor proves nonidentification. The transport bound does not identify the physical reference, provenance, or installation relations.

Corollary 1 (An abstention requirement for missing evidence). *Let P_+ and P_- be transcript laws for opposite witness truths, with $\mathrm{TV}(P_+, P_-) \leq \kappa$ as in Theorem 1. Consider a decision rule returning $\{\text{verified}, \text{refuted}, \text{not identified}\}$. If $P_-(\text{verified}) \leq \alpha$ and $P_+(\text{refuted}) \leq \alpha$, then under either law*

$$P_\pm(\text{not identified}) \geq \max\{0, 1 - 2\alpha - \kappa\}. \tag{6}$$

In particular, additional observations that preserve identical transcript laws cannot turn an omitted causal relation into a reliable affirmative or negative conclusion.

Proof. Total variation bounds the difference in the probability of every event. Under P_+, the verified probability is at most $P_-(\text{verified}) + \kappa \leq \alpha + \kappa$, and the refuted probability is at most α. Their sum is at most $2\alpha + \kappa$. Exchange the laws for the same bound under P_-. Randomized decision rules are covered by adjoining their common independent randomization to the transcript. □

Corollary 2 (Recursive assurance tax). *For an H-stage path, let $C_g, \widetilde{C}_g$ be its physical and recorded stage-pass indicators and $Y = \prod_g C_g$, $\widetilde{Y} = \prod_g \widetilde{C}_g$. If $\Pr(\widetilde{C}_g = 1, C_g = 0) \leq \zeta_g$, then $\Pr(\widetilde{Y} = 1, Y = 0) \leq \min\{1, \sum_g \zeta_g\}$, sharply without cross-stage restrictions. Equal allocation of a whole-lineage false-pass budget ε requires $\zeta_g \leq \varepsilon/H$. If stage-$g$ confirmation is disclosed before Ψ_{g+1} freezes, the unrestricted class also requires at least H sequential outcome-blind confirmation epochs, unless a declared reusable-holdout restriction is imposed.*

Freshness is structural; fidelity is calibrated. For one shared iid injected-fault law, $0/n$ false passes gives exact upper $1 - \alpha^{1/n}$ at confidence $1 - \alpha$, so equal budget requires $n \geq \log \alpha / \log(1 - \varepsilon/H)$. At $H = 12$ and $\varepsilon = \alpha = .05$, this is $n \geq 718$ fault trials before effect estimation. This is an analytic sample-size illustration, not a completed calibration experiment. The number does not transport to stage-specific or adversarial faults; shared use additionally requires the same conditional fault law at each stage.

Let $Y_i = W_H(h_i)$ be the physical whole-history witness and $\widetilde{Y}_i = \widetilde{W}_H(h_i)$ its recorded pass label. A recorded nonpass may mean contrary evidence or *not identified*; it does not by itself establish $Y_i = 0$.

Index lineages in immutable launch order and analyze only the longest resolved prefix. A predeclared deadline resolves timeout or instrument failure as $\widetilde{Y}_i = 0$ while retaining *not identified* and without imputing arm outcomes. For the explicit confidence sequence below, each lineage is adjudicated

before the next is launched. Here $\mathcal{F}_{i-1}$ contains all earlier labels and pre-launch design and calibration information, $\mathcal{P}_i = \mathcal{F}_{i-1}$, and $\widetilde{Y}_i$ is $\mathcal{F}_i$-measurable. Immutable ordering alone is insufficient for overlapping launches; those require a separately justified delayed-feedback analysis.

Before outcomes, specify predictable, $\mathcal{P}_i$-measurable $\zeta_i \geq 0$ from independent calibration or an explicit semantic model. On a single event $\mathcal{E}_{\text{sem}}$ of probability at least $1 - \alpha_{\text{sem}}$, every adjudicated launch satisfies

$$\Pr(\widetilde{Y}_i = 1, Y_i = 0 \mid \mathcal{P}_i) \leq \zeta_i,$$

where ζ_i combines fresh-test error, replay error, and calibrated record/provenance error. If the bound is posited rather than estimated, $\alpha_{\text{sem}} = 0$ only conditional on that semantic model; the guarantee is model-relative, not empirically calibrated. A missing calibration is not assigned zero.

Theorem 2 (Fail-closed adaptive-history bound)**.** *Let L_N be any simultaneous lower confidence sequence satisfying $L_N \leq N^{-1} \sum_{i \leq N} \mathbb{E}(\widetilde{Y}_i \mid \mathcal{P}_i)$ for every resolved launch-order prefix $N \geq 1$ with probability at least $1 - \alpha_{\text{stat}}$. Then, with probability at least $1 - \alpha_{\text{stat}} - \alpha_{\text{sem}}$,*

$$\frac{1}{N} \sum_{i \leq N} \mathbb{E}(Y_i \mid \mathcal{P}_i) \geq L_N - \frac{1}{N} \sum_{i \leq N} \zeta_i \qquad \textit{for every } N. \tag{7}$$

The bound certifies that the mean conditional witness probability of the declared adaptive history generator exceeds π_0 when its right-hand side exceeds π_0. A lower bound below π_0 does not rule out a larger true probability.

One explicit conservative construction for sequential bounded labels is

$$L_N = \max\left\{0, \frac{1}{N} \sum_{i=1}^{N} \widetilde{Y}_i - \sqrt{\frac{\log\{\pi^2 N^2/(6\alpha_{\text{stat}})\}}{2N}}\right\}. \tag{8}$$

Conditional Hoeffding bounds allocate $6\alpha_{\text{stat}}/(\pi^2 N^2)$ to prefix N; their sum is α_{stat}. Appendix D gives the argument. We do not evaluate this bound on the public-release analyses, which supply neither joint witness labels nor calibrated ζ_i.

External calibration. Let $q_i = \Pr(\widetilde{Y}_i = 1 \mid \mathcal{P}_i)$. If a compatible world can set physical $Y_i = 0$ on every recorded-pass event without changing the ledger and its calibration records, the sharp false-pass bound is q_i itself. An in-band channel then supplies no calibration beyond treating every reported pass as false. A trusted observation, intervention, or fault model must cross the system boundary.

Standard binary mixture boundaries provide anytime L_N (Howard et al., 2021). Here we apply them to the same-history estimand and account separately for errors in the records. Applying Corollary 2 conditionally on $\mathcal{P}_i$ gives a conservative whole-history ζ_i from conditionally valid stagewise error bounds. Shared calibration can reduce cost only if its fault law applies across those stages. Reusing rollout outputs may reduce the number of physical jobs, but more data cannot identify a relation that the experiment never measures.

Evaluation design should establish the required causal relations and assess measurement fidelity before estimating full-history success rates. Section 3 asks which barrier released evidence reaches, not whether public systems "really are" RSI; every inspected release stops before a powered full-witness test becomes interpretable.

C Operational assumptions

The following conditions specify a prospective implementation for the physical claim in Definition 1. We use disjoint confirmation data here; a design that reuses disclosed data would need its own adaptive-analysis guarantee. The boundary challenge below is an optional extension for claims across multiple intervention boundaries. None of these conditions is asserted to hold for the public records in Section 3. The reporting checklist and evidence-card schema accompany the supplement.

Assumption 1 (Freeze and fork)**.** Proposal and selection finish before the installation gate. The commit and retain states are complete snapshots or manifests, implement the same interface, and receive matched tasks, resource access and limits, nonintervened model and serving versions, messages, and

failure rules. A model change inside the declared mechanism belongs to the intervention; it cannot be introduced as an unrecorded external change. No post-freeze edit or asymmetric retry occurs before scoring.

Assumption 2 (Target and replay)**.** Target tasks and replay problems $x \sim \mathcal{R}$ are sampled only after the compared mechanisms freeze. Selection data, target data, and successor replay data are disjoint; every mechanism contrast uses the same declared replay population, complete parent-state law, pre-outcome reference rule b, and matched exogenous-randomness law. At stage g, the receipt binds $Q_g^\circ = b(\mathsf{P}_g)$ before outcomes and both Q_g°, Q_g^+ are scored on the same confirmation block. Once a target outcome is disclosed to an updater, it is development data for every later stage and is not reused as fresh confirmation. A failure produced by a frozen system arm receives its predeclared numeric utility. A failure of the confirmation instrument receives no imputed arm outcome and closes the verification history as nonpassing and *not identified*. Retries, repair, and exclusion rules are fixed before outcomes; complete-case conditioning is forbidden.

Assumption 3 (Source and identity)**.** The ledger binds each installed state and its complete parent set to the generating transcripts and allowed external inputs, records every initiated branch, and tests whether the committed successor is the next physical updater. It assigns each initiated lineage an immutable launch index and adjudication deadline; analysis cannot skip an unresolved earlier launch, and a deadline resolves recorded nonpass. The physical relation $\mathrm{InstallEdge}(\widetilde{\psi}_{g+1}, \psi_{g+1}^{\mathrm{exec}}) = 1$ is primitive; matching state roots are fallible recorded evidence for it. Record integrity, replay fidelity, provenance, and noninterference receive distinct error budgets. A hash alone establishes only byte identity.

Assumption 4 (Outcome-blind boundary challenge)**.** Before outcomes, the authors and an independent challenger receive the interface, state schema, and intervention API. A common frozen inventory lists every mutable causal parent of proposal, scoring, selection, materialization, and installation. Each proposed factorization must meet four admission conditions:

(i) Every inventory item is assigned exactly once to the artifact, ordinary adaptive state, successor mechanism, or frozen external schedule.
(ii) The factorization preserves the declared no-swap execution and score semantics.
(iii) The compared states implement the same frozen fork interface and failure policy.
(iv) The claimed mechanism swap can be executed with every input named by the factorization, rather than reconstructed from logs.

A versioned record documents the published rule, parent inventory, challenger registry, assignment maps, admission records, and reasons. File checksums allow subsequent changes to be detected. Every proposal satisfying all four conditions is admitted without author veto; failures and disagreements remain in the ledger, and the finite, nonempty admitted set is frozen before outcomes. If no admissible assignment is available, the verification procedure returns *not identified*. A conforming record alone does not establish inventory completeness, execution equivalence, or physical fork fidelity. Different admitted boundaries need not replace the same physical components; claims of equivalent representations additionally require intervention transport (Appendix C.3). These records document the procedure; they do not establish that the admitted set covers every scientifically reasonable system boundary.

Let $Y_i^{\mathrm{sys}} = \min_f Y_{i,f}$ and $\widetilde{Y}_i^{\mathrm{sys}} = \min_f \widetilde{Y}_{i,f}$ over admitted factorizations. A false recorded system pass contains a false factorization pass, so a conditional union bound gives $\zeta_i^{\mathrm{sys}} \le \sum_f \zeta_{i,f}$ in Theorem 2. Here the sum is over the frozen finite challenged set, with jointly valid calibration events. The outer process stays binary; calibration and fork costs scale with the challenged set.

The verification procedure withholds confirmation when required evidence is unavailable. If a faithful full-state fork is impossible, the experiment may still report target gain or a bundled policy contrast, but not the controlled installation effect I_g. If the successor cannot be linked to the next physical updater, it may report a useful self-produced artifact, but not closure.

C.1 Simultaneous estimation of the three effects

This procedure specifies how one could test the witness in a prospective experiment. It is not an experiment performed on the public releases. Assume the target utility is prespecified and lies in $[0, 1]$;

an unbounded utility requires a separately justified tail model or a predeclared bounded estimand. Clipping after inspecting results changes the estimand and is not permitted.

Freeze a horizon H and either one declared boundary ($F = 1$) or a finite, nonempty challenged set of F factorizations. Let $\mathcal{J}$ be a prespecified list of at most K distinct scalar contrasts. In the general case $K = 3FH$ is a valid upper bound. With direct continuation under an unchanged contract, $K = F(2H+1)$ suffices: H target contrasts and $H+1$ distinct mechanism contrasts per factorization. The shared contrast $R_g = I_{g+1}$ is tested once against $\max(\delta_R, \delta_I)$ when both stage requirements apply. A shared test requires identical physical comparands and the same target, replay law, and resource budget, not merely adjacent generation labels.

For comparison j, both comparands, its sample size n_j, and all failure rules freeze before its confirmation block is drawn. They may depend on previously disclosed development information. Let $\mathcal{D}_j$ denote that information, including the frozen comparands. Each elementary observation is a paired difference in $[-1, 1]$:

$$D_{gi}^{G} = u(Q_g^{+}, z_i; B) - u(Q_g^{\circ}, z_i; B), \tag{9}$$

$$D_i^{a,b} = u(\mathrm{Out}\{\mathcal{T}(x_i, \psi_a; Z_i)\}, z_i; B) - u(\mathrm{Out}\{\mathcal{T}(x_i, \psi_b; Z_i)\}, z_i; B)\,. \tag{10}$$

Here $z_i \sim P_\star$, $x_i \sim \mathcal{R}$, and $Z_i \sim \mathcal{Z}$ are fresh conditional on $\mathcal{D}_j$, with the declared within-pair coupling and independent draws across i. In Equation 10, the target task is independent of both replay executions and is withheld until their output artifacts freeze. The two executions receive identical complete nonintervened inputs and use the same exogenous random object with the proper marginal law in each arm. This is stronger than merely supplying equal seed integers to uncontrolled services. The expectations are respectively G_g and $m_\Theta(\psi_a) - m_\Theta(\psi_b)$; the reference term cancels. If target tasks are batched within a replay, their bounded mean is one replay-level observation. Shared outputs do not turn the tasks in that batch into independent mechanism replications.

Proposition 1 (Simultaneous effect certificate)**.** *Under the preceding conditions, write $\theta_j = \mathbb{E}(D_{ji} \mid \mathcal{D}_j)$ for each frozen contrast and define*

$$\widehat{\theta}_j = \frac{1}{n_j}\sum_{i=1}^{n_j} D_{ji}, \qquad \ell_j = \widehat{\theta}_j - \sqrt{\frac{2\log(K/\alpha_{\mathrm{eff}})}{n_j}}. \tag{11}$$

Then $\Pr\{\ell_j \le \theta_j \text{ for every measured } j\} \ge 1 - \alpha_{\mathrm{eff}}$. Consequently, passing only when every required ℓ_j exceeds its predeclared margin controls false confirmation of the effect conjunction by α_{eff}. No independence between contrasts, stages, or factorizations is required. This statement concerns the measured effect conjunction under faithful execution; it does not certify provenance, installation, or the validity of the target utility.

Proof. Conditional on $\mathcal{D}_j$, D_{ji} are independent, bounded in $[-1, 1]$, and have common mean θ_j. Hoeffding's inequality (Hoeffding, 1963) gives $\Pr(\widehat{\theta}_j - \theta_j \ge r \mid \mathcal{D}_j) \le \exp(-n_j r^2/2)$. The displayed radius makes this probability at most α_{eff}/K. Integrating the conditional inequality and union-bounding over the at most K prespecified comparisons proves simultaneous coverage, including comparisons chosen using earlier disclosed blocks. On that event, every passing lower bound is below its true effect, which proves the second claim. Unmeasured required comparisons cannot pass. □

A lineage may stop early as nonpassing. Increasing n_j after seeing a block's results, however, requires an appropriate confidence sequence rather than the fixed-sample bound. A reused $R_g = I_{g+1}$ estimate is one test of the same already-frozen pair; if feedback changes either object, that is a new comparison requiring new confirmation and an allocated error budget. A frozen system arm's crash or malformed output receives its prespecified utility and remains in the denominator. An instrument failure supplies no valid paired observation: the verification procedure returns *not identified*, without deleting that pair and reporting a complete-case pass. Recorded provenance and replay errors must be bounded separately before Proposition 1 can be combined with Theorem 2.

For planning, assume that every adaptively selected required contrast exceeds its margin by at least a design alternative $\Delta > 0$ almost surely, and that faithful measurement returns all n paired observations for every block almost surely. These are assumptions about the complete protocol, not

conditioning on the observed subset of runs without failures. Conditional Hoeffding bounds for the opposite tail show that

$$n > \frac{2}{\Delta^2}\left\{\sqrt{\log(K/\alpha_{\text{eff}})} + \sqrt{\log(K/\beta)}\right\}^2 \tag{12}$$

is sufficient for all effect tests to pass with probability at least $1-\beta$ under these assumptions. Random instrument failures require a separate failure-probability budget; discarding them does not preserve this power guarantee. This is a conservative planning bound, not a minimax sample-complexity claim. The alternative Δ is a planning assumption, not an observed effect or a guarantee that an improving system exists.

C.2 Finite-horizon feasibility and a mathematical witness

Proposition 2 (Finite-horizon gain budget)**.** *Let $u \in [0,1]$, fix Θ, and suppose direct continuation holds for the entire H-stage witness. Its nonnegative mechanism margins must satisfy*

$$\delta_I + (H-1)\max(\delta_I, \delta_R) + \delta_R < 1. \tag{13}$$

If additionally every artifact reference is the preceding installed artifact, then $H\delta_G < 1$ is necessary. These bounds are relative to a fixed bounded target and common replay distribution; they do not apply to a changing target or to arbitrarily rescaled utility.

Proof. Write $\psi_0, \ldots, \psi_{H+1}$ for the directly continuing mechanisms, so $I_g = m_\Theta(\psi_g) - m_\Theta(\psi_{g-1})$ and $R_g = m_\Theta(\psi_{g+1}) - m_\Theta(\psi_g)$ for $g = 1, \ldots, H$. The first of the $H+1$ distinct increments exceeds δ_I, the last exceeds δ_R, and each of the $H-1$ internal increments exceeds their maximum. Their sum telescopes to $m_\Theta(\psi_{H+1}) - m_\Theta(\psi_0)$. Since the arm-invariant reference cancels and each output's expected utility lies in $[0,1]$, this difference is at most one. Strict threshold inequalities give Equation 13. The artifact gains likewise telescope to $V_\star(Q_H) - V_\star(Q_0) \le 1$ under the additional reference condition. □

This check can reject an impossible claim contract before replay. Equal mechanism margins require $(H+1)\delta < 1$. Under the proposition's assumptions, indefinite direct continuation is excluded if either mechanism margin is fixed and strictly positive. With both mechanism margins zero, this bound is compatible with infinitely many positive, summable competence increments; it does not establish a physically attainable infinite lineage. Prespecified changing-target claims require a separate analysis.

A constructive example, not an empirical result. The fixed-boundary contract is nonempty. Fix any finite $H \ge 2$ and let $\eta = 1/(H+3)$. A task artifact is $q \in [0,1]$, with deterministic target utility $u(q, z; B) = q$. Let the replay input contain an arbitrary parent artifact q with $b(x) = q$, and fix any replay distribution with $\bar{q} = \mathbb{E}_{\mathcal{R}} q$. Mechanism C_k, for $0 \le k \le H+1$, returns artifact $a_k = (k+1)\eta$ and constructs successor mechanism C_{k+1} by incrementing its own stored program constant; after this finite range the values may be capped at one. A declared installation operation executes the constructed successor. All executable update instructions and that constant belong to the mechanism slot in this example; other mutable state is empty. The finite budget is chosen in advance to permit these bounded operations.

Start from artifact $Q_0 = 0$ and mechanism C_0. Stage g uses the output of C_{g-1}: $Q_g^+ = g\eta$, $\psi_g^- = C_{g-1}$, and $\psi_g^+ = C_g$. The latter executes, produces C_{g+1}, and that successor is installed and executed on the same constructed lineage, including the terminal successor C_{H+1}. Therefore

$$m_\Theta(C_k) = (k+1)\eta - \bar{q}, \qquad G_g = I_g = R_g = \eta \quad (1 \le g \le H). \tag{14}$$

Any predeclared margins strictly below η satisfy the finite witness. These are exact consequences of specified mathematical rules, not measured AI performance, a new dataset, or evidence that a deployed system realizes RSI. The example is deliberately programmed; it establishes attainability of the stated fixed-boundary predicate, not autonomous discovery, unbounded improvement, or substantive intelligence.

Why this does not remove boundary dependence. Represent the example by a fixed interpreter U with the program and constant in ordinary state. Transporting the original program replacement still changes that state and preserves its effect. Replacing only U by itself instead gives $I_g = R_g = 0$: this is a different physical treatment, not loss of a causal effect through relabeling (Appendix C.3). If this different intervention boundary passes the mechanical admission tests, the stronger challenged-set witness fails despite the original fixed-boundary witness. Excluding it requires a scientifically justified scope restriction fixed before outcomes. A fixed-boundary witness and a witness robust across all declared treatments remain distinct claims; the latter need not be attainable for every admitted set.

C.3 Physical interventions and their representations

Changing coordinates for one physical treatment differs from changing the treatment. Exact causal abstractions preserve corresponding interventions (Rubenstein et al., 2017; Beckers & Halpern, 2019; Geiger et al., 2025); here we specialize that established framework to our mechanism contrasts rather than propose a new general abstraction theory.

Transporting one experiment. Let $\mathcal{X}$ contain complete physical states, $J_p : \mathcal{X} \to \mathcal{X}$ be measurable and replace the candidate program while retaining other inputs, and $K(x, dy)$ describe one budget-limited update. The common replay law is ν and $v(y) \in [0, 1]$ is the measurable output artifact's target utility. Write $\mu(p) = \int\int v(y)K(J_p x, dy)\nu(dx)$; the shared reference cancels from $I(p, q) = \mu(p) - \mu(q)$. For a measurable bijection τ with measurable inverse, transport the experiment by

$$J'_p = \tau J_p \tau^{-1}, \qquad K'(\tau x, \cdot) = \tau_{\#} K(x, \cdot), \qquad \nu' = \tau_{\#}\nu, \qquad v'(\tau y) = v(y), \tag{15}$$

where $\tau_{\#}$ denotes pushforward. Changing variables gives $\mu'(p) = \mu(p)$ and $I'(p, q) = I(p, q)$: each mean depends only on its arm's marginal law. For paired replays, let $\Gamma_{p,q}(x; dy_p, dy_q)$ have marginals $K(J_p x, \cdot)$ and $K(J_q x, \cdot)$. Transporting $\Gamma_{p,q}$ by $\tau \times \tau$ at corresponding replay states preserves the paired-difference distribution; matching arm marginals alone need not preserve it.

For multistage experiments, transporting the initial law, conditional transition kernels, and any history-dependent intervention rule preserves finite path laws. Under these conditions, jointly transporting successor identities and physical source, installation, and execution events, together with target and reference utilities, is sufficient to preserve the full witness. Equality of scalar effects alone does not provide this guarantee. Resource limits and access permissions must agree between representations; a slower emulator is not automatically a fixed-budget recoding.

Which recodings remain single-slot swaps? Assume nonempty product domains $\mathcal{C} \times \mathcal{M}$ and $\mathcal{C}' \times \mathcal{M}'$, where c contains all nonmechanism inputs. Every mechanism is installable in every context in these declared domains. Define $J_m(c, p) = (c, m)$ and $J'_{m'}(c', p') = (c', m')$.

Proposition 3 (Preservation of the single-slot intervention family)**.** *A bijection $\tau : \mathcal{C}\times\mathcal{M} \to \mathcal{C}'\times\mathcal{M}'$ satisfies $\tau J_m = J'_{h(m)}\tau$ for every m, with a context-independent map h, if and only if*

$$\tau(c, m) = (a(c), h(m)) \tag{16}$$

for bijections $a : \mathcal{C} \to \mathcal{C}'$ and $h : \mathcal{M} \to \mathcal{M}'$.

Proof. Writing $\tau = (\tau_C, \tau_M)$, the commuting equality at (c, p) gives $(\tau_C(c, m), \tau_M(c, m)) = (\tau_C(c, p), h(m))$ for every c, p, m. Thus τ_C depends only on c and τ_M only on m. Bijectivity of τ implies bijectivity of both component maps. Conversely, substitution of the product map proves the equality. □

This elementary classification applies to the full specified family of hard swaps, not to arbitrary partial domains or all causal abstractions. A mixed recoding may preserve the same experiment through Equation 15; its intervention can then modify several coordinates rather than remain a hard replacement of one slot (Geiger et al., 2025).

For example, placing a program in state under an unchanged interpreter gives $\tau(c, m) = ((c, m), U)$. The original replacement becomes $J'_p((c, m), U) = ((c, p), U)$, which still changes the program and preserves its effect. Replacing only U by itself is the identity operation and has zero effect, but it is a different treatment. Relabeling code as state does not erase an existing physical contrast.

The additional successor relation. Equal current competence alone does not establish successor correspondence. For deterministic successor constructors F, F', the context and mechanism maps a, h preserve this correspondence when $h(F(c, m)) = F'(a(c), h(m))$; stochastic constructors use the corresponding joint successor-and-lineage kernel. Combined with competence transport, this is sufficient to preserve R_g, but is not necessary for equality of scalar R_g: distinct successors can have equal competence. Physical installation and continuation remain separate requirements for the complete witness.

Proposition 4 (Immediate artifact responses do not identify successor gain)**.** *Consider an oracle returning only the immediate artifact for a replay input and an opaque current-mechanism label. It reveals neither program contents nor produced-successor identities and does not permit continuation using an unobserved successor. Two finite deterministic systems can agree on every adaptive sequence of such queries while their successor gains have opposite signs at corresponding installed mechanisms.*

Proof. Let labels be $m \in \{0, 1, 2\}$. Both systems return artifact $q = m/2$ for every replay input, with utility q and zero reference utility, so their competence maps equal $m_\Theta(m) = m/2$. The first system constructs successor $F_+(m) = \min\{m + 1, 2\}$; the second constructs $F_-(m) = \max\{m - 1, 0\}$. At $m = 1$, their successor gains are respectively $1/2$ and $-1/2$. Every allowed oracle response agrees; induction gives identical adaptive query transcripts. The differing successor constructors belong to the two systems' complete mechanisms. Exposing those programs or successor identities would distinguish the systems and is outside this observation class. □

This is an exact mathematical counterexample, not an experiment. Full causal abstraction can include the successor variable; artifact-only equivalence omits information needed to identify its gain.

Scope of the boundary challenge. The frozen challenged set can include different physical treatments, whose effects need not agree. We retain $W_H^{\mathrm{sys}} = \min_f W_{H,f}$ as the deliberately stronger requirement that every admitted boundary passes. This is robustness across the declared treatments, not invariance under all recodings. Finite checks of the commuting relations cover only their declared domain unless a structural proof extends them; valid record transformations alone do not establish physical provenance or execution.

D Proofs

Missing-edge theorem. Start from a positive-witness world W^+. For (a), either change the selected artifact on a positive-$P_\star$, zero-query-probability set until G_g crosses its margin, or, absent a reference receipt, hold scores and parent IDs fixed while changing the physical artifact realizing $b(\mathsf{P}_g)$; nonzero query probability gives the total-variation relaxation. For (b), replace the unobserved retain-arm conditional output law by the commit-arm law on the declared replay population, making $I_g = 0$, or detach the tested artifact from the installed mechanism while holding bytes and scores fixed. For (c), change the latent installation relation or restart the next pre-state from another branch while holding artifacts, scores, and identifiers fixed. Each construction changes one conjunct without changing the transcript law. A rule declaring with probability p errs by $1 - p$ in W^+ and p in W^-, giving $1/2$ for identical laws and $(1 - \kappa)/2$ at total variation κ. □

Recursive assurance corollary. If the whole recorded path passes while the physical path fails, some stage lies in $\{\widetilde{C}_g = 1, C_g = 0\}$. The union bound gives the display; disjoint stage-failure events attain it up to one. The outcome-blind epoch claim applies the interpolation construction preceding Theorem 1 after each disclosure: absent a reusable-holdout restriction, the disclosed block places no positive distribution-free lower bound on the feedback-dependent successor. Repeating over H stages yields at least H sequential epochs. This is a lower bound, not a complete job count: the terminal

successor also needs confirmation after freezing, and adjacent $R_g = I_{g+1}$ descriptions must not be counted as independent tests. The zero-failure calibration follows by solving $1 - \alpha^{1/n} \le \varepsilon/H$. □

Explicit confidence sequence. Let $\mu_i = \mathbb{E}(\widetilde{Y}_i \mid \mathcal{F}_{i-1})$. Because $\widetilde{Y}_i \in [0,1]$, conditional Hoeffding's lemma gives $\mathbb{E}[\exp\{\lambda(\widetilde{Y}_i - \mu_i)\} \mid \mathcal{F}_{i-1}] \le \exp(\lambda^2/8)$. Iteration and optimization over λ imply

$$\Pr\left(\sum_{i=1}^{N}(\widetilde{Y}_i - \mu_i) \ge N r_N\right) \le e^{-2N r_N^2}, \qquad r_N = \sqrt{\frac{\log\{\pi^2 N^2/(6\alpha_{\text{stat}})\}}{2N}}.$$

A union bound over all positive integers N proves Equation 8, since $\sum_{N\ge1} N^{-2} = \pi^2/6$. Clipping the lower bound at zero is valid because each $\mu_i \ge 0$. This establishes simultaneous prefix coverage without independence of successive histories, subject to the stated filtration. □

Adaptive-history theorem. On $\mathcal{E}_{\text{sem}}$, each adjudicated launch satisfies

$$\mathbb{E}(\widetilde{Y}_i \mid \mathcal{P}_i) \le \mathbb{E}(Y_i \mid \mathcal{P}_i) + \Pr(\widetilde{Y}_i = 1, Y_i = 0 \mid \mathcal{P}_i) \le \mathbb{E}(Y_i \mid \mathcal{P}_i) + \zeta_i.$$

Sum on the simultaneous L_N event; intersecting it with $\mathcal{E}_{\text{sem}}$ and union-bounding proves Equation 7 for every prefix and optional stop. The in-band consequence is sharp because false-pass probability never exceeds pass probability, while a compatible closure world can set $Y_i = 0$ on every recorded pass without changing the recorded law. □

E Additional empirical evidence

We first give additional comparisons, then follow individual task records, and finally describe the source checks and sensitivity analyses. All observations come from public releases. No released task agent or task grader was rerun.

E.1 Additional comparisons

Recorded updates. The Hyperagents analysis retains all 500 noninitial attempts, including 51 without resolved parent–child scores. Of 41 direct `meta_agent.py` edits, 40 have linked parent records, patches, and scores; ten improve on exposed validation. Thirty-nine of these 40 also change the task agent. Run 2/generation 59 is the exception: it edits the meta agent without editing the task agent and raises validation from .51 to .53. That observation is worth retaining, although it supplies no matched installation comparison. Table 4 follows a separate, previously documented two-transition example within the same 500 attempts, outside the final winner's path $0 \to 2 \to 7 \to 17$.

Table 4 Two Hyperagents transitions from run 3. Both modify the task and meta agents; the scores were also used during selection.

Record	$11 \to 16$	$16 \to 24$	Interpretation
Validation score	.60 → .50	.50 → .60	Exposed score changes
Validation samples	10 → 100	100 → 10	Common tasks unverified
Files changed	Task agent and meta agent		Bundled edits
Source checks	Record and patch metadata		Recorded parent chain
Fresh target/reference	Absent	Absent	G_g not identified
Matched install/retain	Absent	Absent	I_g not identified
Successor replay/install	Absent	Absent	R_g not identified

Evaluation counts change in 48 of the 449 resolved comparisons: 38 go from 100 to 10 samples, and ten from 10 to 100. Of those 48, 35 improve, 11 regress, and two tie. The other pairs use $100 \to 100$ (359) or $10 \to 10$ (42) samples. Equal counts do not establish identical tasks; changed counts alone do not establish bias.

Praxist provides another view of evaluation reuse. Rocket records 12 committed boundaries, 11 next-generation agendas, and 665 evaluation summaries; 664 reuse the same nominal, near-OOD, and hard-OOD banks, while one records a protocol failure. The run ends by SIGTERM after 12 of 30

planned generations. These records show feedback inheritance under reused evaluations, rather than fresh confirmation or a matched complete-updater fork (Li et al., 2026a; Sapient, 2026).

Selection and transfer in other releases. Table 5 separates the units of each comparison. It includes both positive and negative differences; the rows are not a common performance ranking. The small AIDE^2 comparison is reported as inconclusive by its authors, and the inspected paper archive contains no raw run table or analysis script. EvoAgentBench entries are published means, not significance tests. Additional release inventories are retained in the reproduction package rather than treated as further experiments.

Table 5 Additional comparisons. Each row retains its own denominator and evaluation setting.

Source	Comparison	Recorded result
AIDE^2 (Srikanth et al., 2026)	Discovered/reference outer agents; same initial inner agent; three 50-step runs per arm	Mean final grade .780 / .782; plotted bands show seed ranges
AI4AI-Bench (Chi et al., 2026b)	Best versus final checkpoint, among completed multi-checkpoint cases	Best precedes final in 163/265 cases (61.5%); 290 pages inspected
EvoAgentBench (Gao et al., 2026)	Memento, ReasoningBank, GEPA minus Vanilla, matched on model, domain, harness	54 positive, one zero, 17 negative differences
EvoAgentBench	Same model, domain, and method across OpenClaw/Nanobot	Opposite signs in 14/36 matched settings
PostTrainBench (Rank et al., 2026)	Four selected viewer payloads; 2,643 events	Scores 1.00 / .50 flagged; .97 / 0 have neither integrity flag

E.2 Events from individual tasks

Figures 6 and 7 follow two function-calling runs and one software-repair task. These cases were selected retrospectively to inspect how an outcome arose, not to estimate the prevalence of a failure mode.

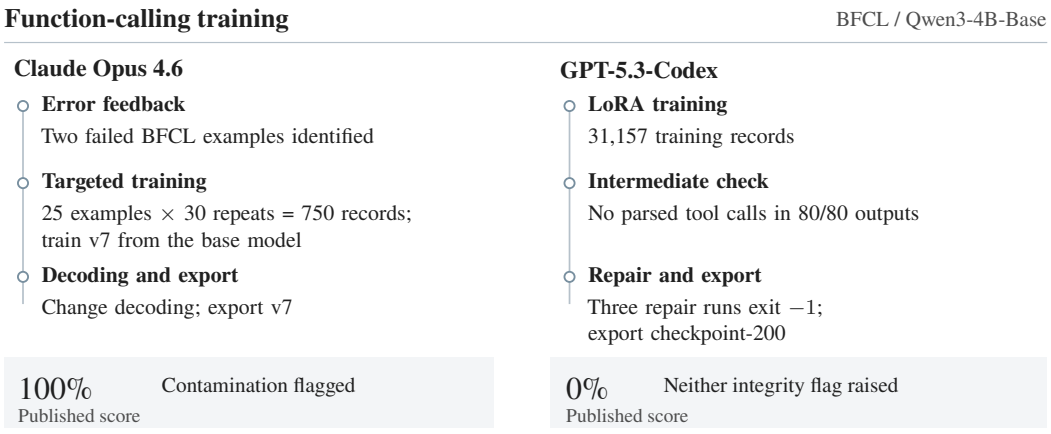

Figure 6: **Selected BFCL training paths** (aisa-group, 2026a). These independent runs each have a nominal 10-hour budget. Scores and integrity flags are publisher-reported; an unflagged run is not established as clean. Connections show event order, not elapsed time.

Function calling. Feedback in the Claude run identifies BAM→JPY and lambda $t \to x$ errors, followed by targeted training examples. Because the run also changes decoding, the final score cannot be attributed to the added data alone. The Codex run's final local 20-example check is zero, as is its published score; the 80 intermediate outputs are a separate check. The four-case selection in

Table 5 estimates neither prevalence nor verifier accuracy, and an unflagged run is not independently established as clean.

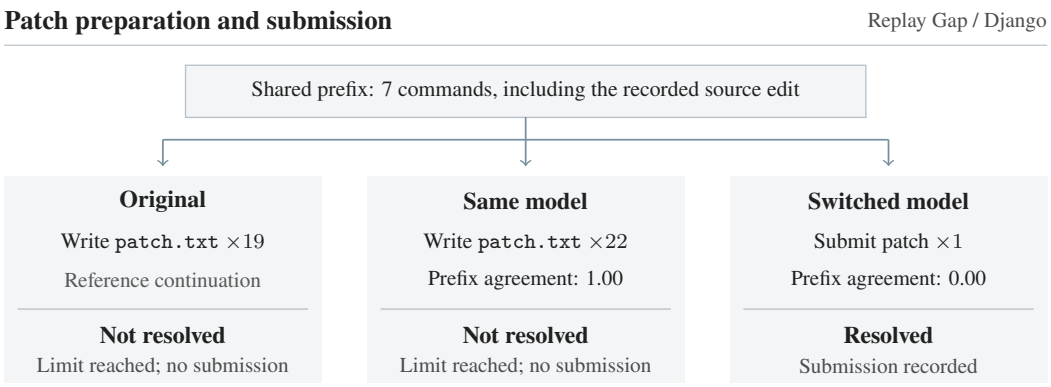

Figure 7: **Recorded continuations of one Django task** (Gonuguntla, 2026b). The three paths belong to `django__django-11163`; prefix agreement compares commands after their shared prefix. Outcomes are release labels. The external model switch does not isolate a causal effect or constitute a self-produced update.

Software repair. The Django task requires an empty dictionary for `fields=[]`. Before the fork, `if fields and ...` becomes `if fields is not None and ...`; the records share this edit and four later patch-writing commands. Original and same-model continuations reach a limit without submitting. The switched continuation submits immediately; its full patch also changes `BaseModelForm`. This is a concrete example of agreement and task completion diverging, not an isolated causal effect of changing models. The source event indices and JSONL lines are retained with the case extraction.

E.3 Sources and comparison rules

Retained sources. Hyperagents contributes 2,525 generation records, including the 500 main-condition attempts, checked against an 11,200-row source index (Zhang et al., 2026a). AutoResearch uses Evo's six-table derivative and the original export (EvoHQ, 2026; Prime Intellect, 2026); Evo's proposal associations are heuristic/model-assisted rather than outcome evidence. Praxist is represented by a retained source-review summary, with the original ZIP external. ResearchArena supplies monitor outputs for one CUDA workload (aisa-group, 2026b). Replay Gap supplies 179 bases and 717 branches across 56 tasks, including 358 switch/control pairs (Gonuguntla, 2026b). Versions, extraction rules, and longer release inventories are in the reproduction package; file checks establish consistency with those records, not independent verification of upstream execution.

AutoResearch joins. Exact matching on (agent, wave/version, run ID) joins 10,380 Evo attempts to 10,428 official export rows: 10,301 unique matches, 68 unmatched keys, and 11 ambiguous keys. Among unique matches, 10,293 finite losses agree, six differ, and two are missing. The original export predates the Evo freeze. Table 6 keeps the same proposal-linked denominator before and after recovery.

Table 6 Recovered AutoResearch source links. Counts require a unique exact match; the unmatched proposal-linked attempt remains in the denominator. A listed script path is not proof of execution.

Source field	Attempts / 5,597
Evo recovered-source catalog	0
Official launched-script metadata	5,596
Official logged-source metadata	5,593

Launched-script coverage is 216/216 in novelty, 2,494/2,494 in v2, and 2,886/2,887 in v3, all Codex runs. The export omits 57 inventory rows with missing configuration paths and separately excludes

missing logs. We retrieve the first unique proposal-linked run lexicographically in each wave: three scripts and three source snapshots, all matching publisher sizes and digests. This selected-object check is not a random sample or a check of every listed path; it neither validates the proposal associations nor supplies an installation intervention.

Replay Gap matching. All same-task bases yield 2,642 associations: 717 recorded parents and 1,925 deliberately reassociated wrong-run candidates. The reference label is publisher-recorded identity, not verified physical ancestry; reassociations are diagnostic controls, not observed linkage errors. Content matching uses messages before the kth assistant message and compares commands or ordered (role, content, actions), excluding IDs, timestamps, token counts, and response metadata. Exact matching rejects 48 recorded parents because branches add boundary user messages: 43 add one, three add two, and two add four. The exploratory prefix-extension rule requires the base pre-fork transcript to be an exact prefix of the branch pre-fork transcript. It retains all parents but accepts six wrong candidates, all at fork step 1 without a preceding action. Removing messages in 717 deletion controls preserves identifier matches but makes content checks unavailable.

Table 7 Replay Gap task outcomes across 717 branches, including one unmatched control.

Post-fork prefix agreement	Resolved	Not resolved	Total
$c = 1$	2	241	243
$c < 1$	3	471	474

Full prefix agreement permits extra branch commands. All five successes in Table 7 occur in two tasks. The 358 matched pairs contain 353 joint failures, three switch-only successes, two control-only successes, and no joint successes. For paired comparisons, early means the first available fork within a task/run/arm; $c = L_{\text{prefix}}/L_{\text{base}}$. Early larger/smaller switches average 3.24%/8.05%, versus 27.29%/73.83% for controls. Paired differences are -24.05/-65.78 points, with task-cluster 95% intervals $[-31.59, -17.02]$/$[-75.24, -56.25]$. The percentile bootstrap uses 20,000 draws (seeds starting at 20261023), keeping a task's pairs together. Equal-task weighting and all four early/late strata remain in the released analysis. Tokens, steps, and wall time are available for 356/358 pairs, but are usage quantities rather than matched available resources or efficiency effects.

E.4 Sensitivity and interpretation

The 17-family comparison. The families in Figure 4 and their original annotations are unchanged. The source index includes AI4AI-Bench, RSIBench-Data, Frontier-Eng, and FML-bench (Chi et al., 2026b; Meng et al., 2026; Chi et al., 2026a; Zou et al., 2026); PostTrainBench, EvoAgentBench, and Terminal-Bench (Rank et al., 2026; Gao et al., 2026; Laude Institute, 2026); SEA-Eval, SE-Bench, Meta-Agent Challenge, and ResearchArena (Jiang et al., 2026; Yuan et al., 2026; Lu et al., 2026; Libon et al., 2026). The other six families appear in Table 9. Trace data were fixed on August 25, 2026, and the registry/source cutoff on August 26; neither was prospective preregistration.

On September 23, two isolated model sessions recoded the same 12 fields without prior labels, manuscript claims, aggregates, or each other's output. Thirteen GitHub archives were retrieved; two pinned Hugging Face datasets were inaccessible. Table 8 separates disagreements from unavailable entries (U); code 0 instead means support not found in inspected sources. There is no independent human validation or adjudication. Agreement therefore concerns repeatability of this coding process, not its accuracy.

Future replay changes from seven explicit original codes to zero explicit, eight partial, and nine absent in each recoding. None of the three codings has a complete explicit six-field design or eight-field public-evidence join; accepting all U entries still leaves the latter at zero. Allowing partial support instead joins DGM, PAST-Bench, AgentStream, SEAGym, and FinEvo-Bench in both recodings, with BenchTrace unresolved in B. Removing rejected-branch accounting yields three strict six-field joins in the original coding. Its explicit eight-field join stays empty after any one or two omissions, whereas some partial-support rows need only one. These checks show dependence on definitions and interpretation, not that no evaluated system could exhibit RSI.

Table 8 Coding agreement. Overall comparisons cover 204 cells; each field covers 17 families. U in either coding, including U/U, is unavailable rather than agreement. A/B agree on 171/187 jointly resolved cells.

Comparison / field	Agree	Differ	U
A / B	171	16	17
A / original	92	101	11
B / original	102	88	14
Field-level agreement between A and B			
Fresh target	12	5	0
Matched no-update	15	2	0
Future replay	17	0	0
Rejected candidates	15	2	0
Next updater	16	1	0
Linked recurrence	16	1	0
Public tasks	16	0	1
Evaluation endpoints	13	3	1
Raw interactions	12	0	5
Candidate archive	13	0	4
Released forks	14	1	2
Lineage IDs	12	1	4

Table 9 Source distinctions behind coding differences. These observations explain disagreements without resolving them by fiat.

Source	Distinction relevant to the code
PAST-Bench (Xue et al., 2026)	Comparison outputs are specified, but generated traces are excluded from commits: a specified comparison is not a released execution.
SEAGym (Zheng et al., 2026)	Checkpoint evaluation makes no new updates; generated run outputs are excluded from version control.
DGM (Zhang et al., 2025a)	Inherited patches record ancestry; the no-self-improvement ablation does not supply a matched retain/install pair.
AgentStream; FinEvo-Bench (Yan et al., 2026; Deng et al., 2026)	Retained-state versus reset comparisons need not isolate installation of a particular complete updater.
BenchTrace (Huang et al., 2026)	Reflexion runners restore prior snapshots. Substituting A's zero for B's next-updater or two-stage code makes its permissive join nonpassing rather than unresolved.

ResearchArena. The retained source has 2,040 valid measurements on 205 runs and 444 complete run–monitor combinations. Requiring every monitor's four access conditions, without inspecting outcomes, gives 58 CUDA runs: seven baselines, 37 embedded-side-task assignments, and 14 independent-side-task assignments (1,392 measurements). They share one workload; the other 147 runs and exclusions remain listed in the release.

AUC counts how often an assigned run scores above a baseline, with half credit for ties. The 357 full-cohort pairs share runs, and the target is assignment discrimination, not successful sabotage. Artifact access also changes tools and environment access. The two result files per run (116 total) match retained source hashes, but only one run has `side_task_success`. In seven of 15 embedded3 runs, `landmine_detected` is false while `kl_landmine` is true; those assays measure different properties. Baselines have no side-task assessment, so absent or negative flags cannot supply clean ground truth.

Paired-run resampling keeps all conditions and monitors together. Score-shift summaries use 20,000 draws per stratum; AUC resampling preserves the 7/37/14 counts. For 144 contrasts, the 95th-percentile maximum absolute centered-deviation band has radius .3265 AUC units. Every full-cohort CoT-after-artifact interval includes zero. Deleting one baseline can reverse that contrast for GPT-5.4, GPT-5.5, and Kimi. Same-generator checks are possible only for Opus 4.8 (3 baseline/9 assigned), GPT-5.4 (1/10), and GPT-5.5 (3/12). These are conditional checks under within-stratum run exchangeability, not finite-sample or cross-task guarantees; they do not establish that CoT harms detection in a wider population.

F Further connections and applications



Figure 8: **Connections across the improvement literature.** Branches group related questions rather than stages of RSI or levels of evidence; a paper may span several branches. The cited systems change different objects, from prompts and model parameters to search policies and update procedures. Their evaluation must identify which object improved.

The following comparisons place the criterion alongside training methods, adaptive agents, and scientific and engineering applications. They discuss reported methods and findings, not additional trace extractions or changes to the frozen 17-family corpus.

F.1 Learning rules, adaptation, and agent evaluation

Learning an update rule. Learned optimizers make the update procedure an explicit learning target: Andrychowicz et al. (2016) train recurrent optimizers across optimization problems, whereas MAML learns an initialization that supports rapid gradient-based adaptation (Finn et al., 2017). Meta-learning representations for continual learning targets interference during subsequent updates (Javed & White, 2019). Lee et al. (2024) instead combine learned representations with sequential Bayesian updates, keeping the neural representation fixed during continual learning. These examples separate an improved starting point, an improved representation, and an improved update rule. All can help future learning, but they are different experimental treatments.

Retaining what was learned. Continual learning asks whether acquiring new capabilities preserves earlier ones. GEM introduces evaluation measures for transfer and forgetting alongside a constrained-update method (Lopez-Paz & Ranzato, 2017). Deep Generative Replay synthesizes earlier training examples (Shin et al., 2017); CLEAR combines replay-based reinforcement learning and behavioral cloning with learning from current experience (Rolnick et al., 2019). Prompt-based approaches offer another route: L2P learns a prompt pool for sequential visual tasks (Wang et al., 2022), and CODA-Prompt learns input-conditioned combinations of prompt components (Smith et al., 2023). This literature motivates reporting retained capability alongside new-task gains. It also clarifies why accumulating useful state need not change the rule that updates that state.

Which part of a prompt is optimized? AutoPrompt searches discrete tokens with gradient guidance (Shin et al., 2020). Prefix-tuning and prompt tuning instead learn continuous vectors while keeping the language model fixed (Li & Liang, 2021; Lester et al., 2021); P-Tuning v2 extends deep prompt tuning across model scales and language-understanding tasks (Liu et al., 2022). Evolvable text prompts are a different object again. EvoPrompt combines language-model proposals with evolutionary search (Guo et al., 2024), while PromptAgent uses Monte Carlo Tree Search and error feedback (Wang et al., 2024b). A useful comparison should therefore specify both the mutable prompt representation and the procedure used to search it, rather than treating all prompt improvement as the same intervention.

Program optimization and query budgets. MIPRO jointly searches instructions and demonstrations for modules in a language-model program, and studies how to improve the proposal process itself (Opsahl-Ong et al., 2024). FedOne addresses a complementary constraint: the query cost of federated, black-box discrete prompt learning (Wang et al., 2025). These settings make different resources visible. End-task accuracy at a fixed budget measures the complete optimization procedure; attributing that accuracy to an improved proposal policy additionally requires a comparison that isolates the policy change. Neither the number of prompt revisions nor the number of model calls supplies that comparison on its own.

Model training and optimizer design. InstructGPT uses demonstrations and preference feedback to train model behavior (Ouyang et al., 2022), while DPO fits a policy directly from preference pairs without a separately trained reward model (Rafailov et al., 2023). QLoRA reduces adaptation memory by training low-rank adapters through a frozen quantized backbone (Dettmers et al., 2023). These methods alter the objective, feedback, or cost of training. More directly relevant to optimizer improvement, Alfano et al. (2025) search a family of preference-optimization objectives on controlled tasks and use the resulting insights for language-model alignment. Evaluating an improved training algorithm is a meaningful endpoint even when that algorithm does not generate and install its own replacement.

Self-generated training and correction. SPIN contrasts model-generated responses with an existing demonstration distribution across training iterations (Chen et al., 2024b). SCoRe trains multi-turn correction through reinforcement learning on the model's own trajectories (Kumar et al., 2025).

AgentRefine uses synthesized environments and refinement trajectories to improve agent generalization (Fu et al., 2025). Each provides a concrete account of how experience changes later behavior. The relevant distinctions are the source of supervision, the part of the system that changes, and transfer to new tasks; the presence of a feedback loop alone does not answer all three.

Agent composition and skill acquisition. MetaGPT organizes collaboration through role-specific operating procedures (Hong et al., 2024); ChatDev structures communication across software-development stages (Qian et al., 2024); AgentVerse studies collaboration among specialized agents (Chen et al., 2024a). CycleQD works at the model level, combining quality-diversity search, model merging, and mutation to acquire multiple agent skills (Kuroki et al., 2025). These are distinct ways to improve a deployed system. To attribute a gain to a model or updater, the comparison must account for changes in coordination, available skills, and inference resources as well.

Checking task completion. WebArena supplies executable web tasks evaluated by functional completion (Zhou et al., 2024). OSWorld extends execution-based evaluation to tasks across real desktop applications (Xie et al., 2024). Such environments help distinguish a plausible action sequence from a completed task. They also supply potential outcome measures for adaptation studies, but an agent's task success is not by itself a measurement of its ability to improve future agents. Our criterion concerns that additional connection, not a replacement for task-level evaluation.

F.2 Recursive systems and applications

Learning how to train. Absolute Zero trains a single model in both task-proposer and solver roles, using code execution to validate generated reasoning tasks and their answers (Zhao et al., 2025). SEAL learns to generate finetuning data and update directives, with the adapted model's downstream performance providing the reinforcement-learning signal (Zweiger et al., 2025). Both make the production of learning experiences part of what is learned. In our formulation, this motivates including the data-generation or self-edit policy in the replacement boundary. The empirical question then concerns the effect of installing that learned policy on future updates, not only the performance of a model it has already trained. This is a distinction between measured effects, not between gradient-based and code-based improvement.

Execution-grounded research. Si et al. (2026) turn pre-training and post-training problems into executable environments and study both evolutionary search and reinforcement learning from experimental feedback. Their analysis distinguishes average reward gains from improvements in the best discovered solution. Frontis-MA1 trains program-evolution operators and composes them in OpenMLE's search framework; its held-out transfer study separately changes the trained model and the framework (Yang et al., 2026). These component comparisons help locate the source of a gain. Our criterion asks a further, sequential question: whether a self-produced replacement improves future model production and produces a more competent successor that is then installed and run. Component transfer and that connected sequence are different experimental targets.

Learning from experience and judgments. RoboCat uses trained robot policies to collect experience for later training iterations (Bousmalis et al., 2023). RISE learns multi-turn response correction from trajectories, comparing against single-turn strategies under equal inference computation (Qu et al., 2024). Self-Rewarding Language Models instead use the model's own reward judgments in iterative preference training and report improvements in both instruction following and judging (Yuan et al., 2024). The latter changes a source of future training feedback, not merely task outputs. These examples locate different possible treatments: the experience-generating policy, response-correction policy, or preference judge. The treatment and subsequent-learning outcome must be specified before comparing complete update procedures.

Adaptive memory and context. Dynamic Cheatsheet maintains reusable strategies and code at inference time without weight updates or explicit ground-truth feedback (Suzgun et al., 2026). A-Mem organizes linked memory notes and updates existing memories as new information arrives (Xu et al., 2025b). ACE uses generation, reflection, and curation to revise contextual playbooks for agent and financial tasks (Zhang et al., 2026c). These are substantive forms of adaptation. Fixed model weights do not imply a fixed complete mechanism when prompts or memory affect future updates.

Conversely, a richer memory does not by itself show that the procedure constructing memories has improved. The declared replacement boundary determines which of these effects is being tested.

Prompts, workflows, and program search. GEPA uses reflection on execution trajectories and Pareto-based candidate selection to optimize prompts, while AFlow searches code-represented workflows with Monte Carlo Tree Search and execution feedback (Agrawal et al., 2026; Zhang et al., 2025b). TextGrad propagates language-model feedback through computation graphs to optimize system components and outputs (Yuksekgonul et al., 2025). metaTextGrad goes a level further, optimizing the prompts and structure of language-model optimizers (Xu et al., 2025a). ShinkaEvolve adapts model selection and accumulates search guidance in a meta-scratchpad (Lange et al., 2026). These are changes to search behavior, not just a growing collection of candidates. AlphaEvolve also evolves meta-prompts and reports deployed improvements to Gemini training kernels (Novikov et al., 2025). The replaced object may therefore be a candidate program, the policy that searches for programs, or infrastructure used to train later models. Our criterion distinguishes an effective optimized search procedure from the repeated production, installation, and execution of more competent successors.

Scientific outputs and research procedures. FunSearch pairs a frozen language model with program evaluation to discover cap-set constructions and bin-packing heuristics (Romera-Paredes et al., 2024). Co-Scientist combines hypothesis tournaments with scientist-guided laboratory validation (Gottweis et al., 2026). The Virtual Lab coordinates scientist agents, with human feedback, to develop a nanobody-design pipeline whose outputs receive experimental testing (Swanson et al., 2025). These studies illustrate different ways to check a discovered artifact: mathematical or executable criteria in one case, physical experiments in the others. Such validation is valuable even when the discovery procedure is not itself the experimental treatment. Testing whether a revised procedure makes subsequent discoveries more reliably requires a separate comparison.

Software repair and kernel optimization. SWE-agent examines agent–computer interfaces for repository repair (Yang et al., 2024b); Agentless uses a prescribed sequence of localization, repair, and patch validation (Xia et al., 2025). Together they motivate treating the interaction interface and workflow as part of the evaluated system rather than attributing all gains to the language model. KernelBench measures functional correctness as well as execution speed and studies refinement with execution and profiling feedback (Ouyang et al., 2025). CUDA-L1 learns a CUDA optimizer through contrastive reinforcement learning (Li et al., 2026b). Here, a faster generated kernel and a better trained optimizer are distinct, measurable achievements. Evidence of either should not be conflated with a self-produced optimizer improving the construction of its own successor.

Changing the improvement mechanism. SICA selects an agent from its performance archive to modify its own implementation and reuses selected agents in subsequent improvement steps (Robeyns et al., 2025). MetaSkill-Evolve explicitly separates fast task-skill updates from slower updates to the policies governing its improvement pipeline, and reports held-out comparisons with single-level evolution (Wang et al., 2026c). Such methods are closer to the meta loop than output refinement alone. Reusing a changed improver is an important execution link; comparing complete search procedures can also establish a useful system-level advantage. Neither measurement is interchangeable with the stage-specific effect of installing a particular frozen updater on matched future inputs. Our sufficient criterion keeps these questions separate without treating an unmeasured effect as absent.

Search budgets and generalization. Gideoni et al. (2026) find simple search baselines competitive with more elaborate code-evolution methods across mathematical discovery, agent design, and machine-learning competitions. Wang et al. (2026b) compare harness evolution with task-level search under comparable feedback and inference budgets, then assess evolved harnesses on disjoint tasks. Their reported gains are sensitive to these evaluation choices. These studies motivate two controls that serve different purposes: budget matching tests whether extra search explains a gain, whereas held-out evaluation tests transfer beyond the search tasks. Neither control alone identifies improvement in the process that constructs future updaters.

Changing evaluators. The Red Queen Gödel Machine co-evolves agents and evaluators, holding its evaluation criterion fixed within an epoch while allowing utility changes at epoch boundaries

(Iacob et al., 2026). This is a substantive alternative to a globally fixed objective. Our fixed-target results apply within a declared comparison, not automatically across changing utilities: cross-epoch improvement needs an explicit common target or a justified correspondence between the old and new objectives. We do not infer such a correspondence from increasing scores under successive evaluators.