RETHINKING PROGRAM SEARCH FOR AUTOML: A REACHABILITY ANALYSIS OF LLM-EDITED TRAINING PROGRAMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Do the code edits an LLM agent makes to a training program reach anything a hyperparameter optimizer could not? End scores cannot tell a new mechanism from a renamed knob. We make the question executable: relative to a frozen program, its configuration space, and an extension declared in advance, we sort every edit into four reachability classes by comparing parent and child forward outputs at fixed configurations. Only edits outside the extension can credit program search, and only against a tuner given the same extension. We audit 467 edits a Claude agent proposed to LoRA adapter code and run controlled searches on Pythia-160M and 410M over ARC-Easy, RACE, and OpenBookQA. Of 332 primary edits, 76% are hyperparameter-level: 56% reconstruct exactly inside the extension and 20% rescale one coordinate. At an equal budget of 492 evaluations, neither our controller nor OpenEvolve or ShinkaEvolve separated on validation from TPE on the unedited program. A best-child acceptance rule overstates progress: in our controller's archives, 38 of 51 admitted programs entered because the best score of a short local tuning run beat the parent's by more than one point, below single-run noise, not because the edit helped at matched configurations. Given a wider surface, the agent produced uncovered programs in every seed, yet none showed an established gain over a tuner given the extension; on OpenBookQA that tuner matched the agent's sealed-test gain. Program search should be credited only for uncovered edits, against a coverage-matched control.

1 INTRODUCTION

LLM agents have moved from tuning machine-learning pipelines to writing them. MLE-bench and MLAgentBench score agents on end-to-end ML tasks (Chan et al., 2025; Huang et al., 2024). FunSearch, AlphaEvolve, OpenEvolve, and ShinkaEvolve let an LLM mutate a program and keep what scores well (Romera-Paredes et al., 2024; Novikov et al., 2025; Sharma, 2025; Lange et al., 2026), and "self-improving" systems apply the same loop to training code itself. The shared premise: a trainable system exposes a program P and a configuration λ; classical AutoML fixes P and searches λ in a declared domain Λ_0 (Bergstra et al., 2011; Falkner et al., 2018); editing P should reach what no setting of λ can. Whether that holds for the edits agents actually produce has not been tested under a controlled budget; we test it for single-shot proposals confined to an adapter surface.

The question is hard to answer from end scores because three effects can be mistaken for program-search gains. An edit that renames a scaling rule may only reparameterize an existing knob, so its gain was available to any optimizer on Λ_0. An edit that helps at one configuration may hurt at another, so comparing parent and child at a single configuration measures an interaction rather than an effect. And a search that scores a child many times reports the maximum of noisy draws, so archives fill with winners that fail a held-out test.

Contributions. *(1) A controlled comparison.* To our knowledge this is the first comparison of LLM program search with hyperparameter search that combines a matched evaluation budget with configuration-matched attribution: the same budget and inner optimizer for every arm, paired seeds, parent and child trained at identical configurations (a *matched panel*), and a sealed test opened once. *(2) Reachability classes.* Relative to a frozen (P_0, Λ_0) and a finite extension Λ_+ frozen

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.

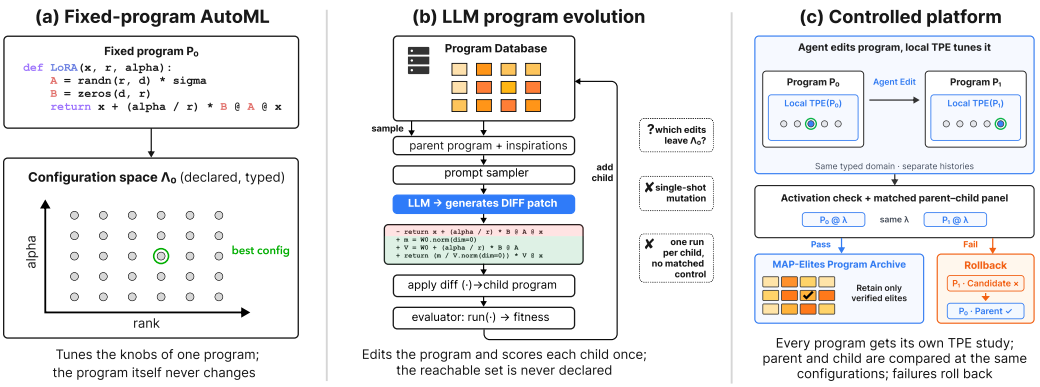


Figure 1: **Three ways to search a trainable system. (a)** Fixed-program AutoML tunes a declared, typed configuration space Λ_0 of one program P_0. **(b)** LLM program evolution edits the program and scores each child with one run; nothing declares which edits leave Λ_0, so a "new mechanism" may be an existing hyperparameter under another name. **(c)** The controlled platform gives every program a local TPE study over the same typed domain, checks that the edit is active, compares parent and child at the same configurations, retains verified elites, and rolls back otherwise.

before any witness runs, every realized edit falls in one of four classes (Section 3): R0 same-point equivalent; R1 reparameterizable by a declared map g on a recorded subdomain of Λ_0; R2 covered by Λ_+; R3 uncovered. Classes are decided by executable *witnesses* that compare forward outputs of the two programs at fixed configurations we call *anchors*; proposal prose plays no role. Only R3 gains can be attributed to program search beyond a declared space. *(3) An audited corpus and platform.* SELFTUNE (Section 4; Figure 1) implements the contract above and checks that an edit is behaviorally active before spending GPU time; with it we audit every structural edit the agent proposed in our registered runs.

For the agents and surfaces we tested, the findings reverse the default expectation. A strict majority of edits reconstruct exactly inside the frozen extension: the agent was, for the most part, tuning hyperparameters in code. Under equal budgets no controller separated on validation from persistent TPE (TPE run on the frozen program for the whole budget), and a half-point sealed-test lead for our controller was not established over 16 seeds (Figure 2). Most archive admissions rest on a child's best local score rather than on matched evidence, and edit effects change sign across matched configurations, which is why end-score comparisons overstate program search. Given a wider surface the agent did leave the extension in every seed, yet those R3 programs showed no established advantage over a tuner given the witnessed extension. Program search can still add value, and the protocol shows where that value must be demonstrated: in the R3 class, against a coverage-matched control, with matched attribution and held-out confirmation.

2 RELATED WORK

LLM-driven AutoML and program evolution. The first wave of LLM-for-AutoML work kept the program fixed: AutoML-GPT, MLCopilot, and AgentHPO propose hyperparameters from task descriptions and past runs (Zhang et al., 2023c; 2024; Liu et al., 2024b), and LLMs have been benchmarked directly as HPO proposers or embedded in Bayesian optimization (Zhang et al., 2023a; Liu et al., 2024c). The second wave edits the program itself. FunSearch and AlphaEvolve couple LLM proposals with evaluation (Romera-Paredes et al., 2024; Novikov et al., 2025); AIDE searches the space of ML solution code (Jiang et al., 2025), and AutoML-Agent and DS-Agent write full pipelines (Trirat et al., 2025; Guo et al., 2024); LLM-driven NAS writes model code under evolutionary or quality-diversity search (Chen et al., 2023; Nasir et al., 2024; Zheng et al., 2023); OpenEvolve, ShinkaEvolve, and autoresearch expose practical harnesses (Sharma, 2025; Lange et al., 2026; Karpathy, 2026); self-improving agent systems turn the loop on the agent's own code (Zelikman et al., 2024; Hu et al., 2025; Zhang et al., 2025; 2026a); and research-agent systems and benchmarks supply executable ML tasks (Nathani et al., 2025; Lu et al., 2024; Toledo et al., 2025). Both waves are judged by end score against a fixed configuration or a short sweep, so code versus configuration is never measured. Concurrent analyses reach a similar verdict for LLMs as

configuration proposers: on fixed spaces classical optimizers win (Ferreira et al., 2026), a budget-matched study traces an LLM advisor's edge to its default configuration (Rodrigues et al., 2026), and simple baselines match code-evolution pipelines (Gideoni et al., 2026). None asks whether the edits themselves leave the declared space, which is our question (Appendix Table 4). Reachability is decided per realized edit, independently of its controller, so it applies to FunSearch- or AIDE-style search as well; Finding 1 speaks only for the controllers we ran.

Hyperparameter optimization and search-space design. TPE models conditional spaces (Bergstra et al., 2011) and Optuna supplies our pinned implementation (Akiba et al., 2019); multi-fidelity and prior-guided tuners (Falkner et al., 2018; Mallik et al., 2023) and adapter-specific rank allocation (Zhang et al., 2023b) would strengthen the fixed-program control, which we keep plain on purpose (Appendix A). NAS learned that headline gains often vanish against random search and tuned fixed-space baselines (Yu et al., 2020; Li & Talwalkar, 2020; Yang et al., 2020): the declared space determines what any optimizer can find. Reachability applies that lesson to code edits: an edit a conditional optimizer could express is a search-space contribution.

Execution-grounded verification and the testbed. Our platform uses realized diffs, activation tests, and matched executions as measurement infrastructure with MAP-Elites retention (Mouret & Clune, 2015); its memory ablation follows agents that retain execution experience across episodes (Shinn et al., 2023; Zhao et al., 2024; Wang et al., 2024; Ouyang et al., 2026; Zhang et al., 2026b; Yang et al., 2026). The testbed edits LoRA-family adapters (Hu et al., 2022; Liu et al., 2024a; Kalajdzievski, 2023) on Pythia (Biderman et al., 2023) and ARC (Clark et al., 2018).

3 Problem Setup and Reachability Classes

Program states and inner trials. An executable workspace is $x = (P, \lambda)$, where P is a training program and $\lambda \in \Lambda(P)$ is a typed configuration valid for that program. At outer round r a controller selects a parent program and an action $u_r \in \mathcal{A}_{\text{prog}}(P_{p_r}) \cup \{\textsc{Keep}, \textsc{Rollback}\}$, producing P_r; an inner policy π_{cfg} then allocates typed configurations $\lambda_{r,k}$ to P_r from a configuration history owned by that program, and the evaluator $\mathcal{E}(P_r, \lambda_{r,k})$ returns validation utility, charged cost, and the realized diff. The corpus and equal-budget studies use the strict case $\Lambda(P) = \Lambda_0$ for every program: a six-variable typed LoRA domain (learning rate, rank, α, dropout, target modules, initializer), which an edit may reweight but cannot add to, remove from, or resize. The coverage-matched study instead gives one tuner the frozen extension Λ_+ defined below.

Reachability classes. Fix the evaluator $\mathcal{E}$, registered randomness ξ, and an outcome $Y(P, \lambda; \xi)$ specified before classification (a forward trace, a checkpoint, or a scalar score; we state which one each witness uses). Let $\Delta(P_0)$ denote the realized child. Relative to (P_0, Λ_0) and a finite extension $\Lambda_+ \supset \Lambda_0$ frozen before any classification, we apply the conditions below in order; the first that holds assigns the class, and a witness that cannot execute returns $\perp$ instead of a class.

$$\text{R0}: \ \forall \lambda \in \Lambda_0 : \ Y(\Delta(P_0), \lambda; \xi) = Y(P_0, \lambda; \xi), \tag{1}$$

$$\text{R1}(D): \ \exists \,\text{declared } g : \Lambda_0 \to \bar{\Lambda}, \ D := g^{-1}(\Lambda_0) : \quad \forall \lambda \in D : \ Y(\Delta(P_0), \lambda; \xi) = Y(P_0, g(\lambda); \xi), \tag{2}$$

$$\text{R2}: \ \exists \, \lambda_+ \in \Lambda_+ : \ Y(\Delta(P_0), \lambda; \xi) = Y(\mathcal{C}(\lambda_+), \lambda; \xi) \text{ for the registered } \lambda, \tag{3}$$

$$\text{R3}: \ \text{none of the above, for a witness that executed.} \tag{4}$$

Here $\bar{\Lambda}$ is the ambient typed space of the six coordinates and $\mathcal{C}$ is the executable constructor that realizes a point of Λ_+ as a program. R0 captures shadowed flags, dead code, and defaults overwritten by the runtime. $\text{R1}(D)$ captures reparameterizations on a nonempty D, with $\text{R1} := \text{R1}(\Lambda_0)$. The map g must be declared before it is tested, and the witness records the subdomain D on which g stays inside Λ_0. If $D \neq \Lambda_0$ the witness returns $\text{R1}(D)$ (*partial*-R1) and stops before Equation 3 is tested: outside D the edit selects a value beyond the frozen range of one coordinate, a range extension rather than a new mechanism, so such edits are hyperparameter-level but count neither as covered by Λ_+ nor as R3. R3 is the only class whose gains can be attributed to program search beyond a declared hyperparameter space; it is a statement about what Λ_+ did *not* enumerate, not a claim of non-enumerability. Every condition is decided at the registered anchors defined below.

How Λ_+ was built and frozen. Λ_+ is a conditional space, one active mechanism family per point with its own typed coordinates on top of Λ_0, realized as a program by a constructor $\mathcal{C}$. Its catalog of candidate families and constructors was assembled *after* the corpus existed, from the subtypes the agent had produced, then frozen before any witness ran or any coverage-matched optimizer was launched. Coverage is retrospective but not automatic: a family receives credit only if a candidate with executable source passes an exact reconstruction witness against the frozen constructor; of the eight subtypes with executable source, five did, one is a declared reparameterization (R1), and two stay unclassified (Table 3). Because the catalog was enumerated from the agent's own edits, the corpus coverage of Finding 2 is an upper bound on what a designer would have declared a priori; the prospective study freezes its catalog before search (Section 5).

Witnesses. Classes are decided by executable witnesses, not by proposal text. An *anchor* is a configuration $\lambda \in \Lambda_0$ registered before any result; we use four, crossing the two learning rates, two rank–α pairs, and the four placements of Λ_0 (Appendix Table 5); the same four serve every matched panel. A witness compares forward outputs of the two programs on one registered token batch at the four anchors in bf16 with a nonzero deterministic adapter state installed through a declared parameter map, and accepts equality at $\texttt{atol}= 10^{-3}$ while recording the maximum deviation. For a subtype s the witness outcome is $\mathcal{W}(s) \in \{\mathrm{R0},\ \mathrm{R1}(D),\ \mathrm{R2},\ \mathrm{R3},\ \perp\}$, where $\perp$ (*unclassified*) is returned when the subtype has no executable source, because its runs logged only a code excerpt, or when the agent's implementations of it return different classes. We never classify from excerpts or proposal text; $\perp$ edits count as not covered, which can only lower the covered share.

Attribution and the joint gate. For a realized transition the platform records

$$\begin{aligned} \Delta_r &= \mathrm{Diff}(P_{p_r}, P_r), \qquad a_r = \mathrm{Activate}(P_{p_r}, P_r), \\ \ell_r &= \mathrm{Verify}(P_{p_r}, P_r) \in \{\textsc{Positive}, \textsc{Negative}, \textsc{Heterogeneous}, \textsc{Uncertain}\}, \end{aligned} \tag{5}$$

where a_r is the activation check and ℓ_r summarizes the matched panel, the child evaluated at the four anchor configurations of the parent: POSITIVE when the median anchor delta exceeds the registered noise floor and no anchor falls below -2 points, NEGATIVE when the median is below the negative floor or any anchor falls below -2 points, HETEROGENEOUS when anchors otherwise disagree in sign, and UNCERTAIN otherwise. A candidate enters the run-local archive only after

$$g_r = \mathbf{1}[u_r = \textsc{Keep} \ \vee \ \ell_r \in \mathcal{L}_{\mathrm{eligible}} \ \vee \ Q_r - Q_{p_r} > \delta], \tag{6}$$

with $\mathcal{L}_{\mathrm{eligible}} = \{\textsc{Positive}, \textsc{Heterogeneous}\}$, where HETEROGENEOUS is retained because such an edit may still pay off under local configuration search, and Q_r the best score of the child's fixed-budget local study; the third disjunct, with δ one accuracy point, is the *optimized-quality rescue*.

4 SELFTUNE: A CONTROLLED PLATFORM FOR NESTED SEARCH

SELFTUNE is the harness that optimizes, attributes, and retains every edited program, including those from external controllers, under identical rules; it is primarily a measurement instrument, and its own controller is one arm of Finding 1.

Program-local inner search. Every program owns a typed TPE history, budget ledger, checkpoint, and rollback target, so no TPE model is fit to configurations whose semantics an edit changed. Each candidate's fixed local budget $B = B_{\mathrm{match}} + B_{\mathrm{inner}}$ first isolates the edit at parent configurations, then lets program-local TPE test its interaction with the space; the gate (Equation 6) applies after both.

Activation before evaluation. A source diff does not establish that an edit changes the program the evaluator runs. Before any training is charged, the platform runs parent and child adapter code through one training step of a small surrogate model at each anchor and rejects candidates whose outputs, gradients, and parameters all match (Appendix B.3); this is why R0 is empty.

Retention, rollback, and external controllers. The controller proposes from a snapshot of a MAP-Elites archive (Mouret & Clune, 2015); a candidate that fails the gate rolls back and cannot update the archive; deployment prefers the persistent fixed-program incumbent on ties. Verified past transitions can be reused as a proposal memory (*verified operator memory*, VOM), an ablated component,

Table 1: **Evaluated systems.** ✓ active, × fixed or absent. All rows share the evaluator, inner TPE, budget, and failure rule; paired attribution trains parent and child at the same configurations.

Reported instance	Edit P	Search λ	Retain programs	Paired edit attrib.	Role
TPE (typed LoRA) (Bergstra et al., 2011; Akiba et al., 2019)	×	✓	×	×	fixed-program control
OpenEvolve 0.3.2 + local TPE (aligned) (Sharma, 2025)	✓	✓	✓	×	aligned baseline
ShinkaEvolve + local TPE (aligned) (Lange et al., 2026)	✓	✓	✓	×	aligned baseline
SELFTUNE (archive + local TPE, ± memory)	✓	✓	✓	✓	this paper

not a claimed advantage. OpenEvolve and ShinkaEvolve run as outer controllers over the same inner optimizer, evaluator, ledger, and failure rule; only SELFTUNE attributes edits at matched configurations (Table 1).

5 EXPERIMENTAL PROTOCOL

Table 2 lists the studies; seeds, prompts, and artifact hashes are in the registry (Appendix B), and every contrast is within one study. *Persistent TPE* (TPE-Base in the coverage-matched study) runs TPE on the frozen program for the whole budget. Search and selection read validation data only: 570 ARC-Easy questions (acc_{norm}), 4,887 RACE questions in the measurement study, and a search split in the coverage-matched study, whose confirmation and test splits open once in that order (Appendix B.9). A failed run is charged its reserved budget and consumes its slot; proposer calls are not charged (Appendix B.11).

Equal-budget study. All arms share a 12-evaluation TPE initialization on P_0. SELFTUNE spends 252 evaluations on the persistent backbone and 240 on ten generations of two candidates, each receiving four matched parent configurations and eight candidate-local TPE evaluations; the sampler fits after four startup points, so half of each local study is model-based, and the anchors start each candidate at its parent's tuned configurations. Twelve evaluations per candidate is the arm's own division of its 240: tuning each candidate longer means proposing fewer. The gate (Equation 6) admits positive or heterogeneous matched evidence, or an optimized gain above one point, mirroring best-child-score acceptance but below single-run noise. OpenEvolve 0.3.2 and ShinkaEvolve reuse the same backbone and spend 240 evaluations on 20 candidates (Table 1). A registered extension added eight seeds to the main contrast, declaring in advance that the pooled result supersedes the first eight (Appendix B.6).

Coverage-matched study. Agent-Open edits a structured MLP-site program, a wider surface than the corpus (Appendix B.1), with a different proposer, prompt, task, and scale. It freezes four proposals per seed and gives program-local TPE to the two with the best matched means. A registered amendment written after all three-arm results, the sealed test included, added TPE-Covered^{+} on OpenBookQA: TPE on Λ_{+} plus the residual-block family the agent had produced, enumerated from frozen programs without reading outcomes. It is a post-hoc exploratory arm, and it made the OpenBookQA test split two-pass.

Decision rule. A contrast is *established* only when its paired bootstrap interval excludes zero and its two-sided exact sign-flip $p < 0.05$. With eight seeds the exact test reaches $p = 0.0078$ only for a unanimous $8/0$ split and $p \geq 0.07$ otherwise, so the interval carries the size. A non-established contrast is *compatible with zero* when its interval is narrow around zero and *unresolved* when it is wide and one-sided, a shortage of power (8–17 seeds for a one-point effect, Appendix Table 8), not evidence of parity.

Table 2: **Registered studies.** Budgets are configuration evaluations (two training epochs each) or GPU-hours per arm and seed. Sealed test: how many times the test split is scored, always after selection locked.

Study (Finding)	Model, task	Seeds	Budget/seed	Arms or units	Proposer; sealed test
Equal budget (1, 3)	Pythia-160M, ARC-Easy	8+8	492 evals	persistent TPE; SELFTUNE with and without memory; OpenEvolve, ShinkaEvolve (validation only)	opus-4-8; once (new seeds scored separately)
Edit corpus (2)	160M–6.9B, ARC-Easy	–	–	332 primary edits, 14 subtypes; 135 appendix edits	opus-4-8; none
Model scale (2)	160M, 410M, 1B, 6.9B	4	108 evals	complete SELFTUNE; no fixed-program control above 160M	opus-4-8; none
Measurement (4)	Pythia-410M, RACE	6	4 anchors	P_0, two R2 edits, one agent edit outside Λ_+; same-seed replays	fixed edits; none
Coverage-matched (5)	Pythia-410M, RACE; OpenBookQA	8	12 GPU-h; 6 GPU-h	TPE-Base (Λ_0), TPE-Covered (Λ_+), Agent-Open; TPE-Covered$^+$ (post hoc)	opus-5; once (OpenBookQA: second pass for post-hoc arm)

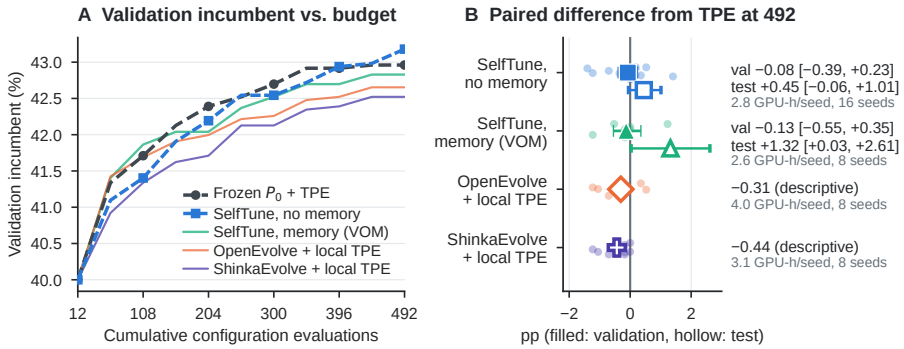


Figure 2: **Under equal budgets, no controller separates on validation from a tuner that never edits code. (A)** Mean validation incumbent over the first eight seeds. **(B)** Paired difference from persistent TPE (points: seeds; bars: 95% bootstrap intervals; filled: validation; hollow: sealed test, or the descriptive external contrasts), with mean training GPU-hours per seed (TPE: 2.6).

6 RESULTS

6.1 FINDING 1: NO CONTROLLER SEPARATES FROM PERSISTENT TPE AT EQUAL BUDGET

On validation, SELFTUNE without memory ends within a tenth of a point of persistent TPE over 16 paired seeds (-0.08, $p = 0.71$; six wins, six losses), a contrast compatible with zero (Figure 2). The memory arm is 0.13 points behind over eight seeds, and OpenEvolve and ShinkaEvolve finish at 42.65% and 42.52% against TPE's 42.97%. On the sealed test SELFTUNE leads by half a point ($+0.45$, $p = 0.15$) and the memory arm by $+1.32$ ($p = 0.125$). Neither lead is established: both intervals admit a one-point gain, but adding eight seeds moved the main lead from $+0.95$ toward zero (Appendix Table 7).

6.2 FINDING 2: MOST WITNESSED EDITS ARE HYPERPARAMETER-LEVEL

The primary corpus holds 332 edits in 14 mechanism subtypes (Table 3). Five subtypes pass every reconstruction witness with zero deviation and hold 187 edits (56%, R2); rank-stabilized scaling is the declared map $g(r, \alpha) = (r, \alpha\sqrt{r})$, which stays in the α grid at 7 of 20 points (65 edits, partial-R1). Together, 76% of edits are hyperparameter-level, and the share holds at every proposing scale (Figure 3A). R0 is empty because the activation check rejects inert edits, and no witnessed subtype reached R3; the remaining 24% are unclassified and counted as not covered: 13% come from runs that logged only a code excerpt, and 11% from subtypes whose sources disagree, through numerically different implementations or a source that implements another mechanism. Witnesses run 53 complete workspaces from the appendix cohort and reach primary edits through their runtime subtype label (Appendix B.5). The two published variants, AdaLoRA-style rank gating and rsLoRA, alone account for 39% of edits.

Table 3: Witness decisions for each mechanism subtype in the primary corpus (four anchors, `atol`$= 10^{-3}$, bf16, nonzero adapter state). "Pass" counts passing over tested complete-workspace candidates from the appendix cohort; unclassified ($\perp$) edits count as not covered, which can only lower the R2 share.

Mechanism subtype	Edits	Pass	Decision $\mathcal{W}(s)$
Rank-stabilized scaling	65	10/10	R1(D), $\lvert D\rvert = 7/20$ points of Λ_0
Rank gate	63	13/13	**R2**
Output gate	56	9/9	**R2**
RMS normalization	30	4/7	$\perp$ (divergent implementations)
Depth-aware scaling	28	6/6	**R2**
Bottleneck dropout	23	3/3	**R2**
Bottleneck activation	17	4/4	**R2**
Custom initialization	8	0/1	$\perp$ (source is another mechanism)
Six further subtypes	42	none	$\perp$ (excerpt only)
R2 (covered by Λ_+)	**187**	**35/35**	**56.3% of all edits, 74% of classified; partial-R1 19.6%, $\perp$ 24.1%**

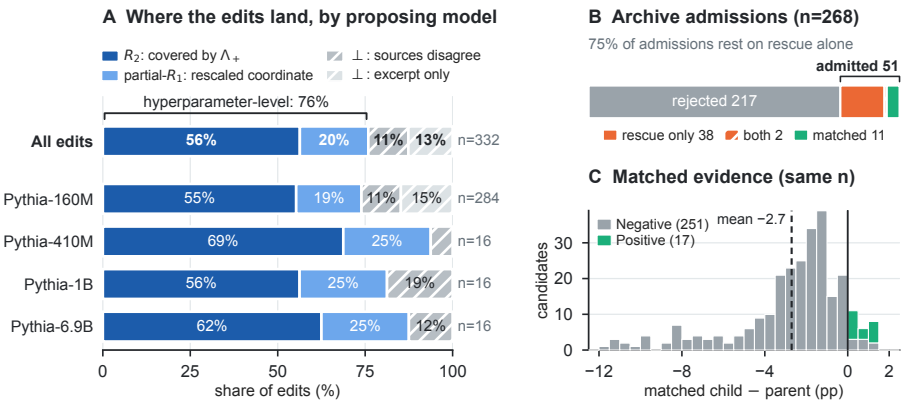


Figure 3: **Most edits are hyperparameter-level, and most archive admissions rest on rescue. (A)** Witness class of the 332 primary edits, pooled and by the model at which the agent proposed them. $\perp$ (*unclassified*): the run logged only a code excerpt, which cannot be executed (42 edits), or its executable sources disagree: numerically different implementations (30) or a source that implements another mechanism (8). Both count as not covered, which can only lower the R2 share. **(B)** The 268 structural candidates of the 492-evaluation study by archive outcome and the gate disjunct under which the 51 admitted ones entered. **(C)** Matched-panel effect (mean over four anchors) of the same candidates; 17 carry a POSITIVE label.

The scale study behind the per-model rows (Figure 4) ran complete SELFTUNE from 160M to 6.9B under one validation-only contract: continued search after the shared initialization added 1.23–2.50 points (14/16 seeds), and an edited program was the final incumbent in 6/16 seeds; without a fixed-program control above 160M, that gain is realized discovery, not a program-search effect.

6.3 FINDING 3: MOST ARCHIVE ADMISSIONS REST ON THE BEST CHILD SCORE ALONE

Of 268 structural candidates with complete panels in the 492-evaluation study, 17 received a positive matched label and 51 entered the archive; 38 of those entered on optimized-quality rescue alone (Figure 3B). In a predecessor 252-evaluation study, a -0.20 validation gap to TPE became -1.19 on the sealed test ($0/1/7$, $p = 0.016$; Appendix B.7).

6.4 FINDING 4: EDIT EFFECTS VARY IN SIGN ACROSS ANCHORS, ALIGNED WITH PLACEMENT

Before the deterministic execution contract, same-seed replays on RACE differed by 2.6 points (SD); after it they are exact (Figure 5). Over six seeds and four anchors, the three measured edits have mean effects of -0.35 to -0.48 points, but each helps at anchors 1–2, which place the adapter

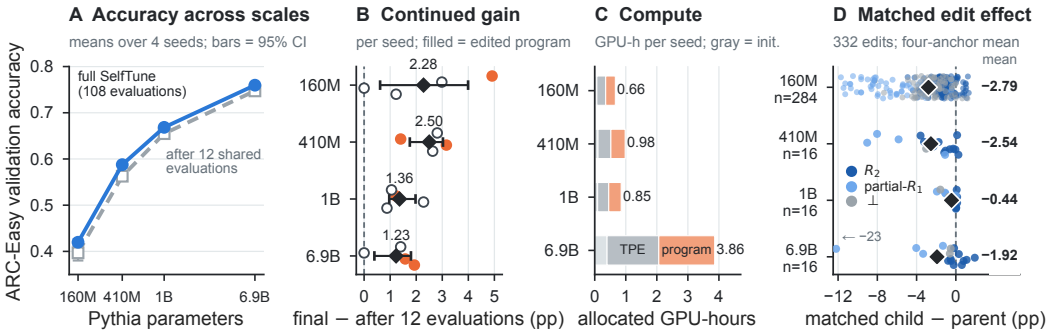


Figure 4: **Complete SelfTune across model scales.** One validation-only contract at every scale: four seeds, a shared 12-evaluation TPE initialization on P_0, two outer generations, 108 evaluations per seed. **(A)** Mean best value after the shared initialization and after the full budget, with 95% seed-bootstrap intervals. **(B)** Per-seed improvement over the shared initialization; a filled point means the edited program, not the persistent-TPE backbone, was the final incumbent. **(C)** Mean allocated GPU-hours per seed (wall time times the 0.125 reserved GPU fraction): initialization (gray), persistent TPE, and program search. **(D)** Matched-panel effect of every primary edit by proposing model, colored by witness class (Figure 3A); diamonds are means, listed at right (Appendix Table 6); one 6.9B edit at -23 points is drawn at the axis edge. All 1728 evaluations completed; because safe selection retains the stronger incumbent, Panel B reports realized discovery, not a causal component effect.

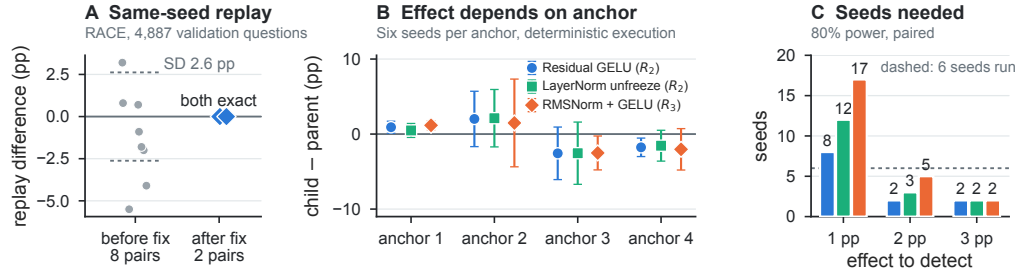


Figure 5: **The measurement floor of a matched panel. (A)** Same seed, configuration, and program run twice on RACE, before and after the deterministic contract. **(B)** Six-seed matched effects of three edits at four anchors (mean ± SD); every edit helps at anchors 1–2 and hurts at 3–4. **(C)** Seeds needed to detect a given effect.

narrowly, and hurts at 3–4, which place it broadly; residual GELU moves from $+2.02$ to -2.57. An edit's sign thus depends on the configuration it is scored at, a placement interaction that a single-configuration comparison cannot see.

6.5 Finding 5: Uncovered Programs Showed No Established Gain over Covered Tuning

Λ_+ contains no MLP-site primitive, so every program that places one is R3. The agent left Λ_+ in every seed: 29 of 32 proposals per task returned a program with an RMSNorm residual block, and the edited program became the incumbent in four RACE and all eight OpenBookQA seeds. Figure 6 gives every contrast. On RACE, Agent-Open trails TPE-Base at every stage, established only on the search split. On OpenBookQA it leads TPE-Base by $+1.33$ on the sealed test, an unresolved contrast, but TPE-Covered moves by the same amount, so the coverage-matched contrast is -0.12 with an interval four points wide. The post-hoc TPE-Covered$^+$ sampled the agent's family in 72 of 435 trials and selected it in one seed (Appendix B.9).

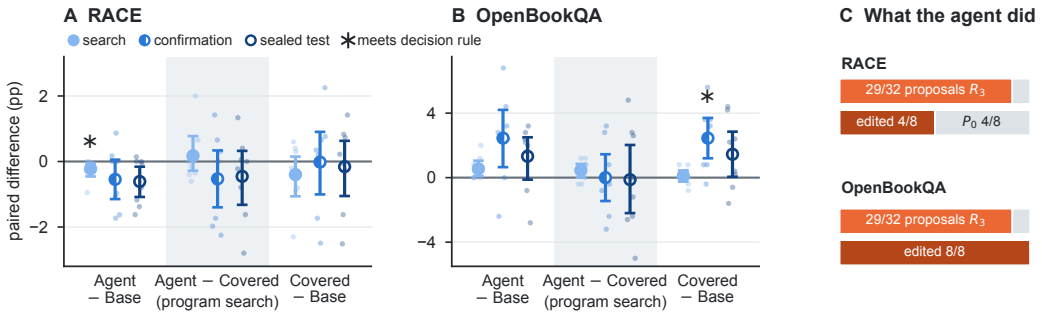


Figure 6: **Coverage-matched study: no uncovered-program gain survives coverage.** The shaded middle group, Agent − Covered, isolates program search: it compares the agent with a tuner given the same extension Λ_+. **(A, B)** Each group shows the paired difference over eight seeds on the search split, the confirmation split, and the sealed test (mean and 95% bootstrap interval; faint points are seeds). **(C)** The agent left Λ_+ in 29 of 32 proposals per task, and its edited program won the seed in 4 of 8 RACE and 8 of 8 OpenBookQA seeds; light gray marks a KEEP proposal or a seed that selected P_0. RACE arms receive 12 GPU-hours per seed and OpenBookQA arms 6, about 50 trials per arm and seed on both tasks because an OpenBookQA trial is cheaper. $*$: interval excludes zero and exact sign-flip $p < 0.05$. Exact values and the post-hoc TPE-Covered$^+$ arm: Appendix Table 9.

7 DISCUSSION

Why the edits reduce to hyperparameters. The editable files act only on the inputs (x, W_0, A, B) the adapter already sees (Appendix B.1); proposals gravitated to published variants designed as knobs, rsLoRA (Kalajdzievski, 2023) and AdaLoRA-style gating (Zhang et al., 2023b); and rescue rewards any child whose local study scores well, which a small conditional family satisfies without leaving Λ_+.

What the protocol adds. These null results are actionable. First, the edits an agent proposes concentrate on a few published mechanisms, so a small conditional extension covers most of its search: five families of Λ_+ reconstruct 56% of all edits, and on OpenBookQA a tuner given Λ_+ gained $+1.45$ points over TPE-Base on the sealed test, matching the agent's $+1.33$ (Finding 5). This suggests a division of labor in which the agent proposes mechanism families, a conditional optimizer exploits them, and the witness decides when a family has been absorbed into the declared space, turning an R3 discovery into a search-space contribution. Second, the platform makes the test cheap to repeat: the activation check rejects inert edits before any GPU time, the matched panel scores parent and child at the same four registered anchors, and the deterministic contract makes replays exact, so a one-point effect needs 8–17 seeds rather than being lost in 2.6-point replay noise (Finding 4). Third, the 467-edit corpus, with subtype labels, matched panels, and hashed artifacts for every started run, lets a new edit be witnessed against the same anchors and Λ_+.

Implications for evaluating program search. The protocol turns three reporting habits into checks any program-search study can run. Report an edit's class, not only its score: a few forward passes per anchor reclassify three quarters of our structural edits as knob changes. Give the fixed-program baseline the declared extension: on OpenBookQA that alone reproduces the agent's sealed-test gain. Attribute at matched configurations under deterministic execution before trusting an archive: best-child scores admitted 38 of 51 elites without admissible matched evidence.

8 CONCLUSION

For the single-shot adapter edits studied here, program search did not separate from a tuner that never edits code: most edits reduced to hyperparameter choices, and the R3 programs the agent did produce showed no established advantage over a tuner given a declared space. Reachability makes "does code beat knobs" testable: witness R3 against a frozen Λ_+, give the control that Λ_+, compare at matched anchors, and confirm on a held-out split. Limitations are listed in Appendix A.

AI USE STATEMENT

Generative AI tools were used to assist with research-method and experiment design, software implementation, experiment orchestration and monitoring, source and literature review, result interpretation, mathematical presentation, and drafting and editing the manuscript. We did not use generative AI to create synthetic datasets, clean or reformat datasets, or analyze qualitative data; those activities are not part of this study. The authors independently verified every definition, derivation, and empirical claim. AI-assisted code is subject to automated tests, frozen protocol checks, and artifact audits. Automated audits check every quantitative claim against the registered run artifacts and every literature citation against its primary source. Test outcomes are not supplied to the search or used as model-development feedback. The authors reviewed all AI-assisted work and take responsibility for the final content, including text, claims, code, and other artifacts produced with AI assistance.

ETHICS STATEMENT

The work uses publicly documented model and benchmark resources and does not involve human subjects. Its main research-integrity risks are benchmark leakage, selective reporting, and compute use. We mitigate these risks with validation-only search, a sealed final test, frozen protocols, explicit negative and invalidated results, failure-aware accounting, and source-hashed artifacts. We report the computational scope and do not infer broad safety or capability conclusions from the narrow benchmark.

REPRODUCIBILITY STATEMENT

Section 5 specifies the evaluation budgets, seeds, data boundaries, and failure rules. The experiment repository records protocol hashes, seed-level outcomes, the harness, hashed manifests, journals, resolved configurations, and analysis code. Section 6 reports every sealed-test outcome separately from the validation-only comparisons.

REFERENCES

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2623–2631. ACM, 2019. doi: 10.1145/3292500.3330701. URL https://doi.org/10.1145/3292500.3330701.

James S. Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. URL https://proceedings.neurips.cc/paper_files/paper/2011/hash/86e8f7ab32cfd12577bc2619bc635690-Abstract.html.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, Usvsn Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar Van Der Wal. Pythia: A suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 2397–2430. PMLR, 2023. URL https://proceedings.mlr.press/v202/biderman23a.html.

Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, Aleksander Madry, and Lilian Weng. MLE-bench: Evaluating machine learning agents on machine learning engineering. In *International Conference on Learning Representations*, volume 2025, pp. 50466–50494, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/7e3767db483c942b883eb4f8cfb74e31-Paper-Conference.pdf.

Angelica Chen, David M. Dohan, and David R. So. EvoPrompting: Language models for code-level neural architecture search. In *Advances in Neural Information Processing Systems*, 2023.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. arXiv:1803.05457, 2018. URL https://arxiv.org/abs/1803.05457. Preprint.

Stefan Falkner, Aaron Klein, and Frank Hutter. BOHB: Robust and efficient hyperparameter optimization at scale. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1437–1446. PMLR, 2018. URL https://proceedings.mlr.press/v80/falkner18a.html.

Fabio Ferreira, Lucca Wobbe, Arjun Krishnakumar, Frank Hutter, and Arber Zela. Can LLMs beat classical hyperparameter optimization algorithms? A study on autoresearch. *arXiv preprint arXiv:2603.24647*, 2026.

Yonatan Gideoni, Sebastian Risi, and Yarin Gal. Simple baselines are competitive with code evolution. *arXiv preprint arXiv:2602.16805*, 2026.

Siyuan Guo, Cheng Deng, Ying Wen, Hechang Chen, Yi Chang, and Jun Wang. DS-Agent: Automated data science by empowering large language models with case-based reasoning. In *International Conference on Machine Learning*, 2024.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*. OpenReview.net, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations*, volume 2025, pp. 21344–21377, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/36b7acf6f6010652b3f2a433774a66fe-Paper-Conference.pdf.

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. MLAgentBench: Evaluating language agents on machine learning experimentation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 20271–20309. PMLR, 2024. URL https://proceedings.mlr.press/v235/huang24y.html.

Zhengyao Jiang, Dominik Schmidt, Dhruv Srikanth, Dixing Xu, Ian Kaplan, Deniss Jacenko, and Yuxiang Wu. AIDE: AI-driven exploration in the space of code. *arXiv preprint arXiv:2502.13138*, 2025.

Damjan Kalajdzievski. A rank stabilization scaling factor for fine-tuning with LoRA. arXiv:2312.03732, 2023. URL https://arxiv.org/abs/2312.03732. Preprint.

Andrej Karpathy. autoresearch: Ai agents running research on single-gpu nanochat training automatically. GitHub software repository, 2026. URL https://github.com/karpathy/autoresearch. Accessed 2026-08-15.

Robert Lange, Yuki Imajuku, and Edoardo Cetin. Shinkaevolve: Towards open-ended and sample-efficient program evolution. In *International Conference on Learning Representations*, volume 2026, pp. 74026–74078, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/7886b9bafe76c52fd568db10ff9772df-Paper-Conference.pdf.

Liam Li and Ameet Talwalkar. Random search and reproducibility for neural architecture search. In *Uncertainty in Artificial Intelligence*, pp. 367–377, 2020.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. DoRA: Weight-decomposed low-rank adaptation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32100–32121. PMLR, 2024a. URL https://proceedings.mlr.press/v235/liu24bn.html.

Siyi Liu, Chen Gao, and Yong Li. Large language model agent for hyper-parameter optimization. *arXiv preprint arXiv:2402.01881*, 2024b.

Tennison Liu, Nicolás Astorga, Nabeel Seedat, and Mihaela van der Schaar. Large language models to enhance Bayesian optimization. In *International Conference on Learning Representations*, 2024c.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The AI scientist: Towards fully automated open-ended scientific discovery. arXiv:2408.06292, 2024. URL https://arxiv.org/abs/2408.06292. Preprint.

Neeratyoy Mallik, Edward Bergman, Carl Hvarfner, Danny Stoll, Maciej Janowski, Marius Lindauer, Luigi Nardi, and Frank Hutter. PriorBand: Practical hyperparameter optimization in the age of deep learning. *arXiv preprint arXiv:2306.12370*, 2023. URL https://arxiv.org/html/2306.12370v2.

Jean-Baptiste Mouret and Jeff Clune. Illuminating search spaces by mapping elites. arXiv:1504.04909, 2015. URL https://arxiv.org/abs/1504.04909. Preprint.

Muhammad U. Nasir, Sam Earle, Christopher Cleghorn, Steven James, and Julian Togelius. LLMatic: Neural architecture search via large language models and quality diversity optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, 2024.

Deepak Nathani, Lovish Madaan, Nicholas Roberts, Nikolay Bashlykov, Ajay Menon, Vincent Moens, Mikhail Plekhanov, Amar Budhiraja, Despoina Magka, Vladislav Vorotilov, Gaurav Chaurasia, Dieuwke Hupkes, Ricardo Silveira Cabral, Tatiana Shavrina, Jakob N. Foerster, Yoram Bachrach, William Yang Wang, and Roberta Raileanu. MLGym: A new framework and benchmark for advancing AI research agents. In *Second Conference on Language Modeling*. OpenReview.net, 2025. URL https://openreview.net/forum?id=ryTr83DxRq.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. Alphaevolve: A coding agent for scientific and algorithmic discovery. arXiv:2506.13131, 2025. URL https://arxiv.org/abs/2506.13131. Preprint.

Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. ReasoningBank: Scaling agent self-evolving with reasoning memory. In *International Conference on Learning Representations*, volume 2026, pp. 94327–94354, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/980ea04d23d1f6908964eba2a74afe45-Paper-Conference.pdf.

Carson Rodrigues, Oysturn Vas, Isaiah Abner DCosta, and Nithish Kumar Prabhakaran. When is an LLM worth it for hyperparameter optimization? A budget-matched study on tabular data finds the warm-start is a default configuration, not the model. *arXiv preprint arXiv:2606.21641*, 2026.

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(7995):468–475, 2024. doi: 10.1038/s41586-023-06924-6. URL https://www.nature.com/articles/s41586-023-06924-6.

Asankhaya Sharma. Openevolve: An open-source evolutionary coding agent. GitHub software repository, 2025. URL https://github.com/algorithmicsuperintelligence/openevolve.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652. Curran Associates, Inc., 2023. doi: 10.52202/075280-0377. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Edan Toledo, Karen Hambardzumyan, Martin Josifoski, Rishi Hazra, Nicolas Baldwin, Alexis Audran-Reiss, Michael Kuchnik, Despoina Magka, Minqi Jiang, Alisia Lupidi, Andrei Lupu, Roberta Raileanu, Tatiana Shavrina, Kelvin Niu, Jean-Christophe Gagnon-Audet, Michael Shvartsman, Shagun Sodhani, Alexander Miller, Abhishek Charnalia, Derek Dunfield, Carole-Jean Wu, Pontus Lars Erik Saito Stenetorp, Nicola Cancedda, Jakob Foerster, and Yoram Bachrach. AI research agents for machine learning: Search, exploration, and generalization in MLE-bench. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 35309–35348. Curran Associates, Inc., 2025. doi: 10.52202/085713-1186. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/328b81881da145412f2bc56c998dfb6a-Paper-Conference.pdf.

Patara Trirat, Wonyong Jeong, and Sung Ju Hwang. AutoML-Agent: A multi-agent LLM framework for full-pipeline AutoML. In *International Conference on Machine Learning*, 2025.

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*, 2024. URL https://openreview.net/forum?id=ehfRiF0R3a.

Antoine Yang, Pedro M. Esperança, and Fabio Maria Carlucci. NAS evaluation is frustratingly hard. In *International Conference on Learning Representations*, 2020.

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. arXiv:2607.28568, 2026. URL https://arxiv.org/abs/2607.28568. Preprint.

Kaicheng Yu, Christian Sciuto, Martin Jaggi, Claudiu Musat, and Mathieu Salzmann. Evaluating the search phase of neural architecture search. In *International Conference on Learning Representations*, 2020.

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*. OpenReview.net, 2024. URL https://openreview.net/forum?id=46Zgqo4QIU.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin gödel machine: Open-ended evolution of self-improving agents. In *International Conference on Learning Representations*, volume 2026, pp. 104223–104294, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/aa5f5e6eb6f613ec412f1d948dfa21a5-Paper-Conference.pdf.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, XiongHui 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*, volume 2025, pp. 34040–34077, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/5492ecbce4439401798dcd2c90be94cd-Paper-Conference.pdf.

Lei Zhang, Yuge Zhang, Kan Ren, Dongsheng Li, and Yuqing Yang. MLCopilot: Unleashing the power of large language models in solving machine learning tasks. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics*, 2024.

Michael R. Zhang, Nishkrit Desai, Juhan Bae, Jonathan Lorraine, and Jimmy Ba. Using large language models for hyperparameter optimization. *arXiv preprint arXiv:2312.04528*, 2023a.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. AdaLoRA: Adaptive budget allocation for parameter-efficient fine-tuning. In *International Conference on Learning Representations*, 2023b.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, Urmish Thakker, James Y Zou, and Kunle Olukotun. Agentic context engineering: Evolving contexts for self-improving language models. In *International Conference on Learning Representations*, volume 2026, pp. 86069–86100, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/8a94ff6f922d995d7d3f4ebf4143e442-Paper-Conference.pdf.

Shujian Zhang, Chengyue Gong, Lemeng Wu, Xingchao Liu, and Mingyuan Zhou. AutoML-GPT: Automatic machine learning with GPT. *arXiv preprint arXiv:2305.02499*, 2023c.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17):19632–19642, 2024. doi: 10.1609/aaai.v38i17.29936. URL https://ojs.aaai.org/index.php/AAAI/article/view/29936.

Mingkai Zheng, Xiu Su, Shan You, Fei Wang, Chen Qian, Chang Xu, and Samuel Albanie. Can GPT-4 perform neural architecture search? *arXiv preprint arXiv:2304.10970*, 2023.

A Limitations

The complete-controller study covers ARC-Easy at 160M; the prospective studies cover RACE and OpenBookQA at 410M. Every surface is a LoRA-family adapter or one structured MLP-site program on multiple-choice QA; whether the class shares transfer to multi-component pipelines (Chan et al., 2025; Huang et al., 2024) is open, and each task needs its own Λ_0, Λ_+, and constructor. The adapter surface is the case most favorable to program search: Λ_0 has six coordinates, so most of what an edit can do lies outside it. All but the extended main contrast use eight seeds, so exact tests establish only unanimous splits and bootstrap intervals are anti-conservative; intervals span about ± 0.5 points on ARC-Easy and ± 2 on OpenBookQA, and about twenty registered contrasts carry no multiplicity correction. The fixed-program control is plain TPE, against which a stronger tuner (Falkner et al., 2018; Mallik et al., 2023) could only widen the gaps. Six of 14 subtypes, holding 13% of edits, were logged only as code excerpts and could not be witnessed. Witnesses take the forward trace as the outcome Y at an installed adapter state, so they decide what a program computes, not how it trains: an edit that changes only the backward pass, optimizer state, or initialization passes the activation check, which also compares gradients and updated parameters, but is forward-identical to its parent, so a forward witness would return R0. No subtype with executable source is of that kind; the three initialization subtypes (17 edits, all $\perp$) would need a checkpoint outcome Y, which the definitions admit. The equal-budget, corpus, and scale studies predate the deterministic execution contract, so their matched labels carry replay noise, and the placement interaction of Finding 4 is not powered at six seeds per anchor.

What the nulls do not show. *Power:* eight seeds establish only unanimous contrasts, so the unresolved sealed-test leads ($+0.45$ to $+1.33$ points) admit a one- to two-point gain; the one contrast extended to 16 seeds moved toward zero. *Under-tuning:* a candidate gets 12 evaluations, so a gain that needs configuration co-adaptation could be missed, but candidates start at the parent's tuned anchors, and the rescue rule already credits gains seen only under local search, yet the arm admitting most children by rescue lost ground on its sealed test (Finding 3).

Other scope limits. We test one memory design fed by noisy labels; reliable-label memory and a registered loss-level R3 positive control remain untested.

B Registered Study Details

This appendix gives the details a reader needs to judge or reproduce the studies: what the agent could edit and was told, how classes were witnessed, how each study was run, and which runs failed. Every number resolves to a registered artifact; the artifact index (Appendix B.12) names them, and the experiment registry holds their paths and content hashes. Superseded, cancelled, and incomplete runs stay in the record but enter no confirmatory claim.

B.1 Editable Surface, Prompt, and Configuration Domain

What the agent could change. Each candidate workspace exposes a custom LoRA implementation in three files, `lora_modules/config.py`, `layer.py`, and `model.py`. The training script, evaluator, data pipeline, task definitions, and both search-space files are protected and restored from source before execution. The candidate contract shipped in every workspace reads, in part:

> Implement one localized, reversible structural change to the custom LoRA adapter … Preserve the frozen base model, data, objective, optimizer, schedule, precision, batching, and target task. Implement only the single adapter mechanism named in the proposal. Keep its rollback local and explicit; do not bundle ordinary hyperparameter tuning with a structural change. … Do not run shell commands, Python imports, smoke checks, training, or evaluation. The outer activation gate owns executable checks.

The locus of nearly every edit is the adapter forward, `result + lora_B(lora_A(dropout(x))) * scaling`, and the parameters that feed it.

Proposal prompt. The outer agent receives the selected archive parent, its completed inner trials, the remaining budget, and (when enabled) verified transition records. The instruction that governs what counts as a proposal reads, in part:

> Make at most one bounded structural change that the frozen inner search space cannot express. The inner TPE policy already controls learning rate, rank, alpha, input dropout, target modules, and initializer. Do not hard-code or duplicate those choices. … Interpret the two evidence channels separately: `matched_delta` estimates how the program edit transfers across identical configurations; `optimized_delta` measures whether program-local TPE finds a better joint result under the same program budget. A matched-negative transition may remain useful when its optimized delta clears the registered margin. Do not describe that case as a positive structural effect. … If no listed mechanism is safe, leave the files unchanged so the harness records KEEP.

The full prompts, the candidate contract, and the deterministic subtype classifier are part of the experiment record.

The configuration domain Λ_0. Six typed variables: learning rate (log-uniform on $[10^{-5}, 10^{-3}]$), rank $r \in \{4, 8, 16, 32\}$, $\alpha \in \{8, 16, 32, 64, 128\}$, dropout uniform on $[0, 0.2]$, target placement in {attention QKV, QKV+output, all core, all linear}, and initializer in {default, Gaussian}. Every program shares this domain; an edit may change what a point means but cannot add or remove a coordinate. Λ_+ adds one active mechanism family per point with its own typed coordinates: the gate parameterization for rank and output gates, the depth-scale initialization and trainability flag, the bottleneck dropout probability, and the bottleneck activation name.

B.2 Positioning Against Prior Evaluations

Table 4 places this study among systems and evaluations that search programs or configurations. Rows other than ours are characterized from the cited papers; apart from the controllers of Table 1, none was run here. Reachability is decided per realized edit, so the protocol applies to any row that edits code.

Table 4: What each evaluation measures. ✓ part of the method or study design; × not; – not applicable. Columns: the method edits program code; it searches a declared configuration space; the study compares against a fixed-program tuner at a matched budget, meaning the same number of configuration evaluations or GPU-hours (no row, ours included, charges LLM calls); edits are attributed by training parent and child at the same configurations; edits are classified against a declared space. OpenEvolve and ShinkaEvolve have no configuration search of their own; our aligned versions add local TPE (Table 1).

System or study	Edits code	Searches config.	Tuner at matched budget	Matched attribution	Classifies vs. space
Program search					
FunSearch (Romera-Paredes et al., 2024)	✓	×	×	×	×
AlphaEvolve (Novikov et al., 2025)	✓	×	×	×	×
AIDE (Jiang et al., 2025)	✓	×	×	×	×
OpenEvolve, ShinkaEvolve (Sharma, 2025; Lange et al., 2026)	✓	×	×	×	×
autoresearch (Karpathy, 2026)	✓	×	×	×	×
LLM-driven NAS (Chen et al., 2023; Nasir et al., 2024)	✓	×	×	×	×
AutoML-Agent, DS-Agent (Trirat et al., 2025; Guo et al., 2024)	✓	×	×	×	×
Configuration search					
TPE, BOHB, PriorBand (Bergstra et al., 2011; Falkner et al., 2018; Mallik et al., 2023)	×	✓	–	×	×
LLMs as HPO proposers (Zhang et al., 2023a)	×	✓	✓	×	×
LLM-enhanced BO (Liu et al., 2024c)	×	✓	✓	×	×
NAS search-phase evaluations (Yu et al., 2020; Li & Talwalkar, 2020)	×	✓	✓	×	×
Concurrent analyses					
LLM vs. classical HPO on autoresearch (Ferreira et al., 2026)	✓	✓	✓	×	×
Budget-matched LLM advisor (Rodrigues et al., 2026)	×	✓	✓	×	×
Simple baselines for code evolution (Gideoni et al., 2026)	✓	×	×	×	×
This paper	✓	✓	✓	✓	✓

B.3 Agent Harness

Two harnesses produced the edits in this paper. Both keep the proposer away from data, evaluation, and execution: the agent writes code or a program description, and a separate process decides whether, how, and on what the result runs.

Equal-budget, corpus, and scale studies: proposer. Each decision calls `claude-opus-4-8` through the Claude Agent SDK with at most 20 turns, the candidate workspace as working directory, and the tools Read, Edit, MultiEdit, Write, Glob, and Grep; no shell tool is granted, and a harness check refuses to start if one is. The prompt carries a typed packet (the selected parent, its completed inner trials with their configurations and scores, the remaining budget, and, in the memory arm, at most four retrieved transition records), marked as data rather than instructions. Transient API errors are retried up to four times.

Workspace. Every candidate starts from a copy of a base workspace whose files are pinned by content hash. Only `lora_modules/config.py`, `layer.py`, and `model.py` are editable; the trainer, evaluator, task definitions, preflight, and search spaces are restored from source after the agent returns, and a candidate whose file set or protected-file hashes differ is rejected. The edit is the unified diff over the three editable files, identified by a hash of parent and child file hashes; an unchanged workspace is recorded as KEEP. The prompt forbids running code or reading data, the evaluator, or other workspaces; this is enforced by the tool list and the prompt rather than by filesystem or network isolation. A tool audit over the execution traces flags calls outside the granted

list: none occurred in the 189 traces of the seed extension, and two attempted shell calls occurred in the predecessor 252-evaluation study.

Decision loop.

1. Select a parent uniformly among occupied archive cells (mechanism family $\times$ edit scope); two candidates per generation over ten generations propose from the same frozen snapshot.
2. Call the proposer; classify the added diff into a mechanism subtype with deterministic keyword rules.
3. Activation check: in a fresh process, install parent and child adapter code in a small surrogate model and, at each of the four anchors, compare hashes of outputs, gradients, and parameters after one forward, backward, and SGD step; also reject changed defaults that the runtime configuration overwrites. An inert, failed, or duplicate (same parent and subtype) proposal is retried once; two failures become KEEP, whose budget goes to parent-local TPE.
4. Matched panel: train the child at the parent's four best configurations.
5. Candidate-local TPE: eight trials in two blocks of four (Optuna TPE, multivariate, constant liar, four startup points), in a study owned by that program.
6. Apply the gate of Equation 6 with a one-point rescue margin; admit to the archive or roll back.

Evaluation. Each trial fine-tunes the adapter for two epochs in bf16 (batch 8, gradient accumulation 4, sequence length 256) on the 2,251-example ARC-Easy training split and scores the 570 validation questions with character-normalized multiple-choice accuracy. Trials run under Ray Tune with a FIFO scheduler at one eighth of a GPU each; a failed trial is charged its full reserved budget and recorded as failed.

Coverage-matched study: proposer. Agent-Open calls `claude-opus-5` once per proposal through the Claude CLI with no tools, a 300-second limit, and one retry. Its packet holds the task, seed, proposal index, the parent P_0, example programs, and the shared space; it contains no outcomes. Four proposals per seed are generated and frozen before any child is trained.

Program language. The response must be one literal `PROGRAM = {...}` assignment under 32 KB with no imports or executable code; an interpreter builds the tensor graph from it. A program places up to two blocks at MLP input or output sites of chosen layers, each a topologically ordered graph of at most 16 nodes over linear, GELU, SiLU, tanh, sigmoid, RMSNorm, dropout, add, and multiply, joined to the residual stream at hidden width; it may also unfreeze layer norms or the last two MLPs and set the training recipe (epochs, warmup, schedule, loss, curriculum, sequence length). Programs that fail to parse or validate, or that repeat an earlier program, are logged and discarded.

Allocation and execution. Each valid proposal receives the four-anchor matched panel; the two with the highest matched mean then share program-local TPE for the rest of the budget. Four P_0 anchor runs per seed are shared by all arms and charged to none. Arms run on whole GPUs under a deterministic contract (fixed cuBLAS workspace, deterministic algorithms, cuDNN benchmarking off) with per-trial time caps; a failed trial is charged the larger of its cap and its actual time. Selection reads only the search split; confirmation and sealed-test splits are opened once each, in that order, after all selections freeze.

B.4 ANCHORS AND THE WITNESS PROTOCOL

Table 5: The four registered anchors, shared by all witnesses and matched panels. Dropout is 0.05 and initialization is the default at every anchor.

Anchor	lr	r	α	placement
1	10^{-4}	8	16	attention QKV
2	3×10^{-4}	32	64	QKV + output
3	10^{-4}	32	64	all core
4	3×10^{-4}	8	16	all linear

The four anchors (Table 5) were registered before any result. They cross the two learning rates and the two rank–α pairs of Λ_0 with its four placements, so anchors 1–2 place the adapter narrowly and 3–4 broadly while learning rate and rank vary within each pair. The same four configurations serve every reachability witness, every matched panel, and the activation check.

A witness runs one registered token batch through parent and child in bf16 with a nonzero deterministic adapter state installed through a declared parameter map, and accepts equality at $\texttt{atol} = 10^{-3}$ while recording the maximum deviation. An R1 map must be declared before execution. A scalar validation score never establishes R0 or R1, and the run-time classifier's scalar or file-path fallbacks are never used for classification. If candidates within one subtype receive different classes, the subtype is unclassified and earns no coverage credit. Because dropout is inactive in an eval-mode forward, the bottleneck-dropout witness adds a train-mode pass with a reset forward RNG that reconstructs the source probabilities 0.05 and 0.10; the output-gate witness applies the sigmoid after the bf16 cast, as the child source does.

B.5 Corpus Construction and Witness Results

Cohorts. The frozen partition holds 332 primary edits and 135 appendix edits. The primary cohort comprises 268 edits from the completed equal-budget run and 64 from the completed four-scale study. The appendix cohort holds 53 edits from a cancelled allocation-control run and 82 from an incomplete attribution-first run; their prompt and run-status differences keep them out of the primary distribution; 13 records lacking matched evidence are excluded. Mechanism subtypes were assigned at run time by a deterministic classifier before any outcome was observed. Primary records retain a source excerpt, not a complete diff, so reachability attaches to a mechanism subtype rather than to an individual primary edit. The 53 appendix records that retain complete workspaces come from the same agent and editable surface; their lineage audit resolves every child source, and they supply the executable witness candidates without entering the primary denominator. The adjudicated table has 14 subtypes; the three largest hold 184 of 332 edits.

Results by subtype. Rank-stabilized scaling declares the map $g(r, \alpha) = (r, \alpha\sqrt{r})$; all ten candidates match P_0 at the mapped point at all four anchors with zero logit difference, and the map stays inside the frozen α grid at 7 of 20 (r, α) points, so the subtype is partial-R1. Rank gate, output gate, depth-aware scaling, bottleneck dropout, and bottleneck activation pass 13/13, 9/9, 6/6, 3/3, and 4/4 candidates, every one with zero maximum deviation; these five R2 subtypes hold 187 of 332 primary edits (56.3%). RMS normalization splits into three numerical paths: the canonical reconstruction matches the four candidates that normalize in fp32, cast to bf16, and then apply the gain, but not the two that apply the gain in fp32 before casting or the one that casts the normalization factor before multiplying (maximum deviation 0.549), so the subtype is unclassified. The sole retained source assigned to custom initialization implements a depth-aware trainable scalar instead; we withhold that subtype rather than reassign it after seeing its source.

By model scale. Table 6 splits the primary corpus by the model at which the agent proposed each edit, using the same adjudicated subtypes and witness results; matched outcomes use each edit's four-anchor panel. The scale study contributes 16 edits per scale, so those rows are descriptive. Figure 4 reports the study's accuracy trajectories; it has no fixed-program control above 160M and is therefore not evidence about program search.

Table 6: Primary-corpus edits by proposing model. "All-positive" counts edits whose four matched deltas are all positive, a stricter condition than the POSITIVE label. A matched delta trains the child at the parent's four best configurations and subtracts the parent's score from the trial that selected them; these studies predate the deterministic contract, so the parent side carries selection bias under 2.6-point replay noise, and the mean is a gate signal rather than an effect size. The registered-anchor study of Finding 4 measures effects of -0.35 to -0.48 points.

Model	Edits	R2	partial-R1	Unclassified	Mean matched Δ (pts)	All-positive
Pythia-160M	284	**157 (55%)**	53 (19%)	74 (26%)	$\mathbf{-2.79}$	1
Pythia-410M	16	**11 (69%)**	4 (25%)	1 (6%)	$\mathbf{-2.54}$	0
Pythia-1B	16	**9 (56%)**	4 (25%)	3 (19%)	$\mathbf{-0.44}$	0
Pythia-6.9B	16	**10 (62%)**	4 (25%)	2 (12%)	$\mathbf{-1.92}$	2

B.6 EQUAL-BUDGET CONTROLLER STUDY

The study uses eight seeds on Pythia-160M and ARC-Easy. Every arm starts from the same 12-evaluation TPE initialization and ends at 492 configuration evaluations; search and selection read validation data only.

Seed extension. After the eight-seed analysis and its sealed test, a registered amendment added seeds 52009–52016 for persistent TPE and the no-memory arm, the paper's main contrast, under the identical protocol, prompts, gate, proposer model (`claude-opus-4-8`), and nondeterministic execution contract; the memory arm was not extended. The amendment fixed the analysis before any new GPU work: the pooled 16-seed result supersedes the 8-seed result regardless of direction, the test split is scored a second time for the 16 new checkpoints only after their selection locks, and the pooled tests are reported without adjustment for sequential extension. Table 7 gives the three cohorts. All 5,856 physical evaluations completed with no failed trial; the launcher failed three times before any GPU work and the terminal stage once at its checkpoint-reconstruction gate, each recorded and resumed without replaying a trial.

Table 7: The main contrast, SELFTUNE without memory minus persistent TPE, in percentage points, by seed cohort. Intervals are paired 95% bootstrap intervals; p is the two-sided exact sign-flip test.

Cohort	Validation	Sealed test
First 8 seeds	$+0.22$ $[-0.15, +0.64]$; .44; $5/1/2$	$+0.95$ $[+0.08, +1.84]$; .125; $4/2/2$
New 8 seeds	-0.37 $[-0.79, -0.02]$; .19; $1/3/4$	-0.06 $[-0.40, +0.27]$; .81; $2/3/3$
Pooled 16	-0.08 $[-0.39, +0.23]$; .71; $6/4/6$	$+0.45$ $[-0.06, +1.01]$; .15; $6/5/5$

Table 1 summarizes how the external controllers were aligned. Their endpoints carry seed-bootstrap intervals around each arm mean, but the frozen analysis defines paired contrasts only for the two SELFTUNE arms against persistent TPE, so the external comparisons are descriptive.

The memory ablation. Verified operator memory is a deterministic index of past transitions, each stored with its realized diff, mechanism subtype, matched-panel label (POSITIVE, NEGATIVE, HETEROGENEOUS, UNCERTAIN), and per-anchor deltas. At proposal time the index returns records compatible with the current task, model, search space, and evaluator, ranked to surface at least one positive and one negative record first, and the proposer receives them as advisory context with the instruction to treat mixed evidence as uncertain. The memory alters neither the gate, the budget, nor the inner optimizer; it changes only what the agent is told. It was active: by the end of each seed it held 60–71 records, of which 96% (516 of 538) carried NEGATIVE labels and 22 were POSITIVE. A memory that mostly says "this failed" can narrow proposals but cannot point to a gain, which is why Finding 1 treats it as an ablation and Finding 4 explains why the few positive labels were unreliable.

B.7 ARCHIVE ROUTING AND THE PREDECESSOR SEALED TEST

The 492-evaluation ledger contains 268 structural candidates and 51 archive admissions. Forty admissions satisfy optimized-quality rescue, 38 of them without admissible matched evidence; 13 have admissible matched evidence, two of which also satisfy rescue. Four further POSITIVE candidates passed the gate but did not displace the elite of their archive cell.

The predecessor sealed test cited in Finding 3 belongs to an earlier 252-evaluation controller with verified operator memory. Its selection was frozen before the single test pass. Against persistent TPE the test contrast is -1.19 points (95% bootstrap interval $[-1.79, -0.58]$, W/T/L $0/1/7$, two-sided exact sign-flip $p = 0.016$); against its own no-memory arm the validation gap of $+0.57$ became -0.72. This establishes non-transfer for that earlier controller; it is not a randomized ablation of the later rescue rule.

B.8 DETERMINISM AND MATCHED-PANEL VARIANCE

Before the deterministic execution contract, eight same-seed replay pairs on RACE differed by 2.6 points (SD). Under the contract, two replay pairs have identical adapter hashes, document sequences, predictions, and validation accuracy. The variance extension then ran six seeds at the four anchors, so each edit has 24 parent–child pairs; the seed is the independent unit and the anchors a fixed block

(Table 8). The study makes no efficacy claim, and its seed counts are development estimates rather than a prospective power analysis.

Table 8: Six-seed matched-panel variance. Effects and standard deviations are percentage points. The last column is the development estimate of seeds needed to target a one-point effect.

Edit	Mean	Pair SD	Seed-mean SD	Seeds
Residual GELU	-0.35	3.13	1.00	8
LayerNorm unfreezing	-0.38	3.38	1.20	12
RMSNorm + GELU	-0.48	3.70	1.46	17

B.9 Coverage-Matched Studies

Table 9 lists every registered contrast of the RACE and OpenBookQA three-arm studies at all three stages. Selection used the search split only; the confirmation split and the sealed test were each opened once after selection froze. RACE uses a 2,000-question search partition of whole articles, a 2,887-question confirmation partition, and a 4,934-question sealed test; OpenBookQA uses 250, 250, and 500 questions. Agent-Open edits placements of eight primitive operations (linear, activations, RMSNorm, add, multiply) at MLP input or output sites, the training recipe, and trainability. All arms share the same four P_0 anchor observations per seed before their budgets begin. TPE-Covered$^+$ enumerates the 18 residual-block variants in the frozen Agent-Open programs (all 29 OpenBookQA programs reconstruct); seeds, anchors, budget, and partitions are those of the original arms, which were not rerun.

Table 9: All registered three-arm contrasts over eight paired seeds. Entries are mean percentage-point difference, paired 95% bootstrap interval, two-sided exact sign-flip p, and W/T/L. A–B denotes Agent-Open minus TPE-Base; A–C denotes Agent-Open minus TPE-Covered; C–B denotes TPE-Covered minus TPE-Base; C$^+$ denotes the amended arm TPE-Covered$^+$.

Stage	A–B	A–C	C–B
RACE			
Search	-0.22 $[-0.46, -0.06]$; .031; 0/2/6	$+0.18$ $[-0.28, +0.77]$; .719; 3/1/4	-0.40 $[-1.06, +0.15]$; .297; 3/0/5
Confirmation	-0.55 $[-1.15, +0.05]$; .156; 2/2/4	-0.53 $[-1.39, +0.33]$; .328; 3/1/4	-0.02 $[-1.01, +0.90]$; .977; 5/0/3
Sealed test	-0.61 $[-1.08, -0.16]$; .094; 1/2/5	-0.45 $[-1.32, +0.33]$; .375; 3/1/4	-0.16 $[-1.06, +0.63]$; .734; 4/0/4
OpenBookQA			
Search	$+0.55$ $[+0.10, +1.05]$; .125; 4/4/0	$+0.45$ $[+0.00, +0.85]$; .141; 5/1/2	$+0.10$ $[-0.25, +0.45]$; .812; 3/3/2
Confirmation	$+2.45$ $[+0.65, +4.20]$; .062; 6/1/1	$+0.00$ $[-1.45, +1.45]$; 1.00; 3/1/4	$+2.45$ $[+1.20, +3.70]$; .016; 7/0/1
Sealed test	$+1.33$ $[-0.12, +2.50]$; .125; 6/0/2	-0.12 $[-2.20, +2.03]$; .953; 3/1/4	$+1.45$ $[+0.05, +2.85]$; .109; 6/0/2

Stage	A–C$^+$	C$^+$–B	C$^+$–C
OpenBookQA, amended arm TPE-Covered$^+$			
Search	$+0.15$ $[-0.25, +0.60]$; .672; 4/0/4	$+0.40$ $[-0.05, +0.90]$; .250; 5/1/2	$+0.30$ $[-0.25, +0.80]$; .438; 4/3/1
Confirmation	$+1.25$ $[+0.05, +2.50]$; .125; 6/1/1	$+1.20$ $[-0.50, +2.80]$; .258; 5/0/3	-1.25 $[-2.65, +0.05]$; .164; 2/0/6
Sealed test	$+0.83$ $[-0.90, +2.78]$; .477; 4/0/4	$+0.50$ $[-1.40, +2.35]$; .648; 5/0/3	-0.95 $[-2.78, +1.15]$; .430; 2/0/6

The amended TPE-Covered$^+$ arm. The family added to Λ_+ is a closed world built only from the frozen Agent-Open programs: layer subset, MLP site, block operations, linear widths and zero-initialization, dropout probabilities, recipe, and trainability, 18 semantic variants in all. Every OpenBookQA proposal reconstructs exactly as a point of this family under the four-anchor witness. The arm ran under the original protocol with the original three arms' terminal results reused, not rerun; its selections were locked before the confirmation split and the sealed test were opened. Across its 435 settled trials TPE sampled the new family 72 times, with the same mean search accuracy as the rest (42.2% versus 42.3%), and one seed selected a residual-block program.

Pilots. A two-arm operability pilot (Agent-Open against persistent TPE, six GPU-hours per arm and seed, Pythia-410M, RACE) preceded the three-arm design. All eight Agent-Open proposals were inconclusive on the matched panel and rolled back, so both arms deployed P_0; the $+1.14$-point arm difference (W/T/L $5/0/3$, $p = 0.109$) reflects different P_0 initialization policies, not a program-search gain, and no sealed-test access was spent. A separate 992-evaluation TPE pilot found fixed-program TPE stalling before one third of its trace on SciQ in three of four seeds while still improving after 60% of the trace on RACE in two of four, so any fixed takeover point from TPE to program search would be task-specific.

B.10 Incomplete and Failed Studies

Every started formal experiment is reported here regardless of outcome. In the RACE three-arm study, 44 of 1,194 trials failed and were charged their full reserved cap: 24 exceeded the frozen time cap, 12 were interrupted by driver restarts, and 8 were refused by the source-binding guard. Completed prefixes were reused, settled trials were never rerun, and interrupted trials stayed charged. The OpenBookQA study had 4 interrupted trials, and the TPE-Covered^{+} arm lost its driver once to a terminated session, charging 4 in-flight trials their reserved cap before resuming from completed prefixes. One of the OpenBookQA study's confirmation cells needed a documented precision fix (BF16 autocast for a program with unfrozen LayerNorm) before it could be scored; no checkpoint, split, or selection changed.

The adaptive TPE pilot was planned for 1,296 evaluations and interrupted after 992; its traces are development evidence and cannot establish TPE saturation or a Λ_0 ceiling. A first prospective two-arm study on 410M and RACE failed its feasibility contract before producing efficacy evidence because its 600-second evaluation cap was shorter than a full RACE evaluation; the redesigned study is the one in Section 5. These runs consumed compute and remain auditable, but none enters a treatment contrast. The four-scale study completed but has no equal-budget fixed-program control above 160M; it contributes 64 primary-corpus edits and identifies no program-search effect.

B.11 Hardware and Cost Accounting

All studies run on four-GPU NVIDIA GB300 nodes (284 GB per GPU, aarch64) with PyTorch 2.12, CUDA 13.0, and bf16. The equal-budget and model-scale studies share each GPU among eight trials and charge wall time times $1/8$; the measurement and coverage-matched studies run one trial per GPU and charge exclusive GPU-seconds including setup, evaluation, and failures.

B.12 Artifact Index

Table 10 names the registered artifacts behind the main-text numbers. Paths and content hashes are in the experiment registry.

Table 10: Registered artifacts referenced in this paper.

Label	Content
A1	Equal-budget paired analysis, SELFTUNE vs. TPE
A2	External-controller endpoints
A3	Adjudicated subtype counts
A4	Rank-stabilized scaling mapping witness
A5	Reachability aggregate
A6	Same-seed replay audit
A7	Matched-panel variance analysis
A8	Coverage-matched protocols, confirmations, and sealed tests (RACE, OpenBookQA)
A9	492-evaluation held-out lock, analysis, and terminal audit