HEURITRAIN: SEARCH ORGANIZATION AND VERIFICATION IN AUTONOMOUS POST-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Autonomous post-training agents seek to improve models by iteratively proposing training changes, evaluating outcomes, and using feedback to guide further experiments. However, accumulated results do not by themselves determine which configurations to extend or which interventions to test. We address this challenge by formulating autonomous post-training as sequential search over complete Data–Recipe configurations, governed by an explicit experimental state. We introduce HeuriTrain, a harnessed framework that enforces task contracts and records experiments, and Factorized Experimental State (FES), which organizes Data and Recipe exploration through separate parent pools, histories, and edit permissions. Evaluations on IFEval, Countdown, and a deterministic Splay control show that policy rankings vary across settings. FES achieves a mean advantage of 1.73 percentage points over Global Archive in eight paired IFEval searches (95% CI $[-0.25, 3.10]$) and the highest mean performance among the compared policies across 18 matched Splay blocks. Further analyses reveal interactions between edits and evaluation and execution shifts that can exceed policy gaps, motivating verification alongside search. Code is available at https://anonymous.4open.science/r/heuritrain-review-code-D059.

1 INTRODUCTION

Autonomous research agents turn goals into sequences of proposed, executed, and revised experiments (Jiang et al., 2025; Yamada et al., 2025; Karpathy, 2026). Post-training makes this loop costly: each trial combines training data with a training recipe, and its outcome is observed only after training and evaluating the resulting model (Ouyang et al., 2022; Shao et al., 2024; Tie et al., 2025). The researcher must choose its next experiment from limited feedback within a finite compute budget.

Yet a measured score does not settle that choice. A greedy loop continues the leading configuration, whereas archive-based search can revisit another measured parent (Karpathy, 2026; Zhang et al., 2026a). Earlier outcomes, including unsuccessful trials, can inform later proposals (Yu et al., 2026; Ning et al., 2026a); the result used as evidence need not be the configuration inherited for training. Because Data and Recipe choices can interact, a promising edit need not transfer across complete configurations. A selected score also warrants verification beyond its discovery run (Ning et al., 2026b). The next decision therefore depends on parent eligibility, visible evidence, permitted edits, and remaining opportunities.

Existing systems already let agents plan subsequent trials from feedback and explore post-training pipelines (Yano et al., 2025; Yu et al., 2026; Rank et al., 2026; Ning et al., 2026a). AEVO casts evolution as an interactive environment whose state is the accumulated evolution context and lets a meta-agent edit the search mechanism (Zhang et al., 2026b); we hold mechanisms fixed and compare how organizations of that state shape post-training experiments whose evaluations cost GPU-hours and vary across executions. We study two linked questions within a defined research task: *How does the organization of eligible parents, visible evidence, permitted edits, and remaining opportunities shape the experiments an agent proposes? Which observed differences support choosing one search policy over another?*

To study these questions, we propose HeuriTrain, a framework for autonomous post-training under a human-defined task contract. The contract fixes the starting weights, objective, editable Data and Recipe surfaces, evaluator, and budget. Within these bounds, a search policy selects a measured

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.

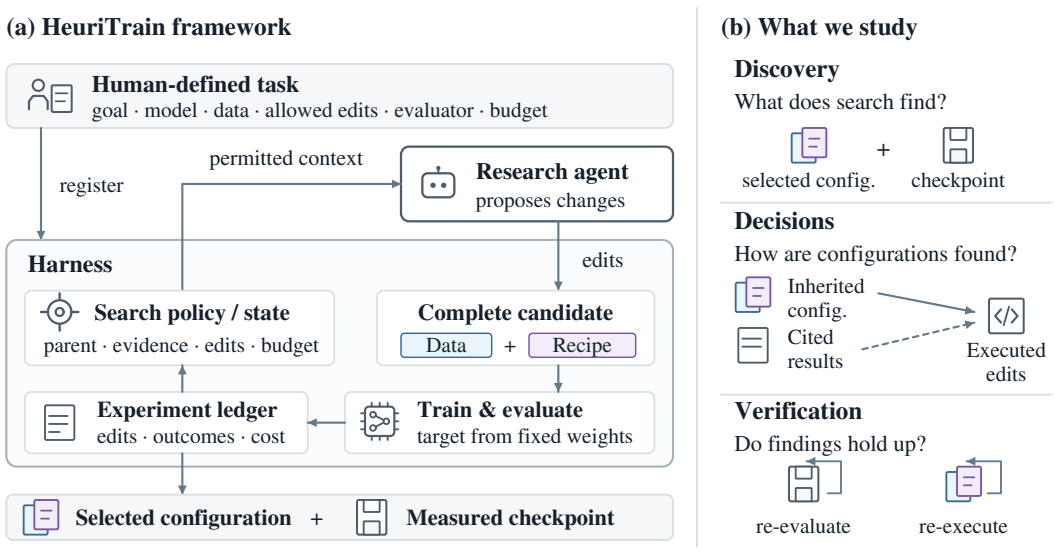

Figure 1: HeuriTrain overview. (a) A research agent and the Harness conduct budgeted experiments within a human-defined task, returning the best measured Data–Recipe configuration and its checkpoint. (b) Our study examines discovery, decisions, and verification.

parent and determines which prior experiments the research agent can consult. The agent uses this context to propose an edit; the Harness trains and evaluates the resulting complete configuration from the registered weights. It records the context, rationale, executed changes, and outcome in an experimental ledger, from which the policy constructs the next request.

We evaluate HeuriTrain on two post-training tasks, IFEval (Zhou et al., 2023) and Countdown (Pan et al., 2025), and a deterministic CPU task, Adversarial Splay. These studies demonstrate that HeuriTrain supports autonomous improvement of training configurations while making the underlying decisions traceable and search policies systematically comparable. The deterministic control further shows that organizing experimental state can improve proposal quality, establishing it as a consequential design choice for autonomous research agents.

In summary, our contributions are threefold. **(1)** We introduce HeuriTrain, a framework that turns autonomous post-training into auditable experiments over complete Data–Recipe configurations, linking each agent decision to its available evidence, executed intervention, and measured outcome. **(2)** We formulate search policies through explicit experimental state and introduce FES alongside Global Greedy and Global Archive baselines. FES finishes above Archive in 7 of 8 paired IFEval searches, with a mean advantage of 1.73 pp (95% CI $[-0.25, 3.10]$). **(3)** We provide an empirical study across two post-training tasks and 18 matched deterministic-control blocks that separates policy effects from initialization, evaluation, and execution effects.

2 HeuriTrain Framework

Overview. HeuriTrain enables a research agent to improve a target policy through budgeted post-training experiments. We define the objective and search space (Section 2.1), the registered environment and experiment loop (Section 2.2), and the policies organizing successive trials (Section 2.3). Figure 1 connects this workflow to discovery, decision analysis, and verification.

2.1 Post-Training Objective and Search Space

Post-training objective. Post-training adapts an existing policy model to improve its behavior on a target task. Given an initial policy π_{θ_0} and a limited experimental budget B, our goal is to find an adapted policy whose responses receive higher reward from a task-specific verifier. Let $\mathcal{T}$ denote the target task distribution and $V(z, a)$ the verifier reward for response a to input z. We express policy

quality as

$$J(\pi_\theta) = \mathbb{E}_{z\sim\mathcal{T},\, a\sim\pi_\theta(\cdot|z)}\left[V(z,a)\right], \tag{1}$$

with responses generated under the registered decoding procedure. This verifier defines the outcome used to assess a trained policy; a training algorithm may use a separate optimization reward.

From policy improvement to configuration search. Post-training from a fixed initial policy requires deciding **what data to learn from** and **how to update the model**. We organize these decisions into two editable surfaces: DATA specifies the selection and preparation of training examples, and RECIPE specifies the training settings and code. Together, they form a complete configuration $x \in \mathcal{X}$ that defines an executable training experiment.

HeuriTrain operates over this configuration space. A *search policy* q organizes successive experiments: the research agent proposes changes, and the Harness trains and evaluates the resulting target policies π_θ, providing feedback for subsequent proposals. At the budget limit, the Harness returns a **selected checkpoint** together with its **complete Data–Recipe configuration**. The checkpoint provides the best measured policy discovered during search, while the configuration preserves the procedure for re-execution and further refinement, allowing its repeatability to be assessed.

2.2 HARNESSED EXPERIMENT LOOP

The research agent chooses training interventions within a registered task; the Harness validates and executes its proposals and records their outcomes (Figure 1(a)). The resulting checkpoints, configurations, and records support the analyses summarized in Figure 1(b).

Task registration. A human makes the post-training objective executable by registering

$$\mathcal{E} = (\theta_0, \mathcal{D}, \mathcal{X}, \mathrm{Train}, \mathrm{Eval}, \mathrm{Render}, B),$$

which specifies starting weights, training and evaluation assets, legal configurations, training and evaluation procedures, the agent's information interface, and budgets in proposal opportunities and GPU-hours. Before search, the research agent constructs a complete root configuration x_0 within the task contract, which the Harness trains and evaluates from the registered starting weights θ_0. Task-specific runners accommodate different training backends and rollout systems; the contract identifies fixed infrastructure and the editable Data and Recipe components. This bounds the agent's autonomy around choosing experiments within a defined environment.

Proposal and execution. For each iteration, the Harness implements the search policy's rules to select a measured parent and render its complete configuration, permitted history, editable files, and remaining budget. The agent decides the concrete intervention and returns an edit and, where requested, a rationale. The Harness checks the submission against the registered edit scope and configuration rules, rejecting invalid edits and duplicates before training. An accepted edit is applied to the parent to form a complete candidate x_t, which is trained from θ_0 and scored at its final checkpoint. Candidates therefore inherit configurations while starting from the same registered weights. The evaluator remains outside the editable scope, and rendered requests omit development answers.

Feedback and accounting. Let $\mathcal{L}_t$ denote the experiment ledger available before iteration t. Each attempt adds its parent, rendered request, any accompanying rationale, submitted and executed changes, execution status, and any resulting checkpoint, score, and cost. Invalid edits, duplicates, and failed executions are recorded and charged under the contract. The ledger supplies evidence for subsequent proposals and connects what the agent could inspect, what it inherited, and what it actually tested. This supports analyzing decisions through both their stated rationale and their measured consequences (Section 3.3).

Selection and verification. At the budget limit, the Harness returns the best measured complete configuration and its associated final checkpoint:

$$\hat{x}_B = x_{\hat{t}}, \quad \hat{\theta}_B = \theta_{\hat{t}}^{\mathrm{final}}, \qquad \hat{t} \in \arg\max_{t\in\mathcal{I}_B} y_t, \tag{2}$$

where y_t is the evaluator score and $\mathcal{I}_B$ indexes measured entries within budget, including the root. Selection compares final checkpoints across candidates. The saved checkpoint supports re-evaluation,

while its complete configuration supports re-execution to assess whether the discovery performance persists. Combining separately tested edits creates a new candidate requiring its own evaluation.

Algorithm 1 in Appendix B.1 summarizes the loop. The Harness enforces the contract through request construction and submission validation; process-level isolation depends on the runner. Section 3.1 specifies study settings, and Appendix A.1 documents task artifacts and runner implementations.

2.3 Experimental State and Search Policies

The research agent conducts the search under a prescribed *search policy* q. This policy governs which configurations the agent may extend, which earlier results it receives, and which components it may edit; the agent chooses the specific intervention within those constraints. The Harness implements these rules through the requests and validation described above.

To express these choices explicitly, we define the *experimental state* constructed before iteration t:

$$S_t = (\mathcal{P}_t, \mathcal{M}_t, \mathcal{U}_t, b_t), \tag{3}$$

where $\mathcal{P}_t$ contains eligible parents, $\mathcal{M}_t$ contains visible experimental records, $\mathcal{U}_t$ specifies editable files, and b_t records remaining opportunities. The policy constructs this state from the ledger $\mathcal{L}_t$ of past attempts. We compare global search with an alternative organized by surface.

Global Greedy. The agent extends the highest-scoring measured configuration, following the greedy continuation strategy of `autoresearch` (Karpathy, 2026). The agent may modify Data, Recipe, or both, with global history providing evidence beyond its selected parent.

Global Archive. The agent receives a parent sampled uniformly from all measured configurations, with the same global history and edit scope as Greedy. Following the archive idea of the Darwin Gödel Machine (Zhang et al., 2026a), this keeps lower-scoring configurations available for exploration.

Factorized Experimental State (FES). The global policies share evidence across Data and Recipe, although these surfaces involve heterogeneous interventions whose effects can interact. FES asks whether organizing experimental context by surface better supports the agent's next decision. It maintains separate Data and Recipe states. A schedule selects the active surface, whose state supplies the parent, visible history, and editable file; the other surface remains at root. Each candidate still defines a complete training experiment evaluated by the Harness.

We compare these policies as bundles of parent selection, evidence, edit scope, and allocation, examining the configurations discovered and how agents use prior results to reach them.

3 Experiments

3.1 Experimental Setup

Tasks. We evaluate HeuriTrain on two post-training tasks and a deterministic control. IFEval (Zhou et al., 2023) evaluates instruction following after LoRA SFT of Qwen2.5-3B-Instruct; Countdown (Pan et al., 2025) evaluates arithmetic reasoning after full-parameter GRPO of Qwen2.5-3B. Both expose data selection and training recipes as editable surfaces. Adversarial Splay evaluates search without execution variability: the agent modifies even- or odd-position keys in a 4,096-access sequence to maximize splay-tree rotations.

Implementation details. The research agent chooses the initial Data–Recipe configuration within the registered constraints, without a prescribed initialization strategy. This agent-generated configuration establishes the trained root for subsequent search. Matched policy comparisons reuse the same measured root, so their differences arise after this shared initialization. We use Claude Opus for paired IFEval and Countdown global searches, and Claude Sonnet for Adversarial Splay (Anthropic, 2026), keeping the researcher fixed within each matched comparison. Coding-agent runs use Claude Code (Anthropic, 2025), while paired IFEval uses fresh, tool-free calls. Countdown candidates receive 200 GRPO updates; IFEval follows the proposed epoch count. Task instructions, agent prompts, editable files, and execution settings are provided in Appendix A.1. Comparison-specific budgets and repetitions are described alongside the corresponding results.

Metrics. We report strict prompt accuracy on 541 IFEval prompts, strict Pass@1 on 2,000 Countdown problems, and total rotations for Splay; higher scores are better throughout. IFEval requires every instruction in a prompt to pass. Countdown requires the last answer block to reach the target using every supplied number exactly once, and this evaluator is distinct from the GRPO training reward. Search performance is the best measured score within budget, including the root. We also report gains over the root and GPU-hours, with paired differences and confidence intervals for matched comparisons.

3.2 Main Results

Overall performance. Tables 1 and 2 show that HeuriTrain supports autonomous improvement of post-training configurations across both tasks. In Table 1, all three search policies improve their trained roots, with gains of 4.60–16.10 percentage points (pp). Global Greedy achieves the highest reported IFEval endpoint, while FES achieves the highest Countdown endpoint, showing improvements under different search organizations across SFT and GRPO. The paired IFEval study further compares FES and Global Archive under a shared root and budget. FES returns higher endpoints in seven of eight searches, with a mean advantage of 1.73 pp (95% CI $[-0.25, 3.10]$). Together, these results demonstrate that the agent–Harness loop can translate Data–Recipe edits into measured gains over the registered training configuration, with FES achieving higher endpoints in most matched comparisons.

Table 1: **Post-training results on IFEval and Countdown.** Score denotes strict prompt accuracy for IFEval and strict Pass@1 for Countdown, both in percent; (+Δ) denotes the gain over each trained root in percentage points. **Bold** and underlining mark the best and second-best search results within each task. All searches use Claude Opus 5 as the research agent.

	Countdown			IFEval		
Method	Score ↑	Evaluated	GPU-h	Score ↑	Evaluated	GPU-h
Unadapted policy	0.85	—	—	47.32	—	—
Trained root	41.00	—	—	36.41	—	—
Global Greedy	<u>49.15</u> (+8.15)	18	42.33	**49.35** (+12.94)	20	8.50
Global Archive	45.60 (+4.60)	17	40.17	<u>45.66</u> (+9.25)	21	8.78
FES	**57.10** (+16.10)	17	42.90	43.07 (+6.66)	18	8.58

Table 1 shows that agent-generated initialization contributes differently across the two tasks. On Countdown, the trained root raises strict Pass@1 from 0.85% to 41.00%, and search adds another 4.60–16.10 pp, further improving an already effective training configuration. On IFEval, the initial configurations reduce accuracy below the unadapted model's 47.32%. Subsequent search improves every root: Global Greedy reaches 49.35%, while Archive and FES recover part of the initial loss but remain below the unadapted reference. These results distinguish the contribution of root construction from that of subsequent search: Countdown gains build on a strong trained root, whereas IFEval search first needs to recover performance lost during initialization.

Paired policy comparison. Under a shared starting point and budget, FES more frequently discovers higher-scoring configurations than Global Archive. We compare eight seed-matched pairs from the same IFEval root, with 24 proposal opportunities and a 14 GPU-hour cap per search, keeping training and shuffle seeds fixed. FES achieves higher endpoints in seven pairs and raises mean accuracy from Archive's 45.52% to 47.25%, a paired advantage of 1.73 pp (95% CI $[-0.25, 3.10]$; Table 2). The first completed pair favors Archive by 4.62 pp, illustrating why the repeated comparison matters: the predominant direction across searches differs from that of the initial pair.

This advantage translates into more complete recovery from the agent-generated initialization. Starting from a root 12.75 pp below the unadapted model, FES returns configurations that nearly recover the entire deficit on average and exceed the unadapted reference in five of eight searches. Archive also improves the shared root, but its mean endpoint remains 1.80 pp below that reference. Thus, within the same search budget, FES more often finds configurations that restore or improve upon the initial

policy's performance, connecting the choice of experimental-state organization to the quality of the returned model.

Table 2: **Paired search results on IFEval.** Scores denote strict prompt accuracy (%), reported as mean ± sample SD across eight seed-matched searches per policy. Gains are measured over the shared trained root in percentage points (pp). The paired difference is FES minus Global Archive, with a 95% percentile-bootstrap confidence interval obtained by resampling matched search pairs.

Policy	Strict accuracy (%) Mean ± SD	Mean gain over root (pp)
Global Archive	45.52 ± 0.81	$+10.95$
FES	47.25 ± 2.26	$+12.68$
Paired difference: $+1.73$ pp; 95% CI: $[-0.25, 3.10]$ pp		
FES finishes higher in $7/8$ matched pairs.		

3.3 Agent Behavior and Candidate Selection

Search organization shapes refinement. Figure 2 illustrates how the agent develops interventions along two complementary dimensions: the examples used for training and the way model parameters are updated. In the Global Greedy path, broad data filtering gives way to targeted checks on response structure, alongside adjustments to learning rate, gradient accumulation, and LoRA capacity. Global Archive permits the same edit scope while keeping previously measured configurations available as alternative parents. FES organizes refinement into separate branches: Data proposals target source selection and constraint coverage, while Recipe proposals adjust trainable modules, capacity, and update intensity. These paths make the practical distinction concrete: global search can revise interacting choices together, whereas FES tests each surface against a fixed background.

Feedback guides which configuration to continue. The unsuccessful proposals in Figure 2 show how exploration proceeds while the best measured configuration is retained. FES continues testing edits after either branch reaches its incumbent, and regressing candidates leave the best-so-far curve unchanged. A separate recorded Greedy iteration shows how such outcomes enter later decisions: the agent inherits the $263/541$ incumbent, cites a previous $226/541$ failure when motivating a quadratic update-scale proxy, and submits a Recipe-only edit that scores $249/541$. The Harness retains $263/541$. The cited result shapes the stated hypothesis, while the inherited configuration supplies the executable starting point. This distinction allows unsuccessful experiments to inform subsequent proposals without discarding earlier improvements.

Useful edits must be evaluated together. The separate Data and Recipe branches raise a natural question: can their improvements be combined? An executed IFEval crossing shows why the combination requires its own measurement. From a parent scoring $259/541$, a Data edit reaches 267, a Recipe edit reaches 266, and their combination scores 248 (Table 6). The three children are proposed together and each is evaluated once. In these executions, the Data edit improves the score under the parent's Recipe but lowers it under the edited Recipe. An edit's measured benefit therefore belongs to the complete configuration in which it was tested; combining favorable components creates a new candidate for selection. The four measurements quantify this dependence: the Data edit contributes $+8$ prompts under the original Recipe and -18 under the revised Recipe, a -26-prompt interaction contrast. This crossing provides a concrete reason to preserve each edit's tested counterpart when deciding whether to reuse it.

The selection criterion determines what is retained. Once candidates have been evaluated, the selection signal determines which improvement the search preserves. Among Countdown's 17 Archive candidates and root, selecting by mean training reward over the final 50 updates returns a checkpoint scoring 12.70% strict Pass@1. Selecting by the registered evaluator instead returns 45.60%, a 32.90 pp difference within the same candidate pool (Appendix C.5). The search has already generated the stronger checkpoint, but retaining it depends on using the task objective for final selection. Evaluation thus contributes directly to the quality of the returned policy.

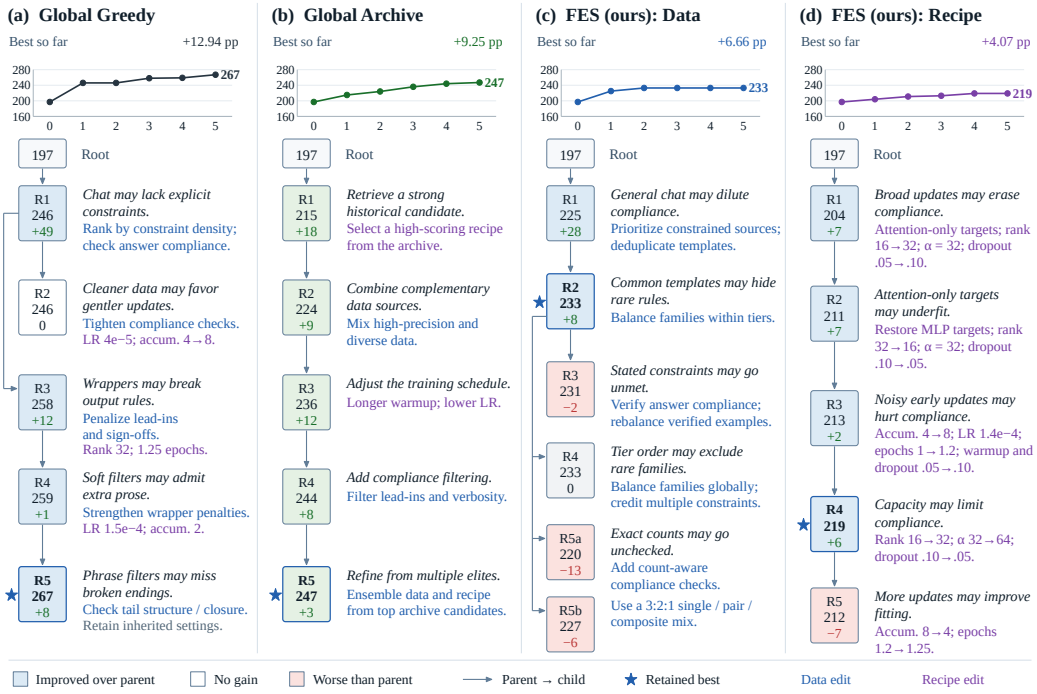

Figure 2: **Case study of IFEval search organization.** Panels show Global Greedy, Global Archive, and the Data and Recipe branches of FES. Nodes report strict-correct prompt counts and changes from the displayed parent; arrows indicate configuration inheritance. Curves show best-so-far counts, stars mark retained candidates, and colors distinguish Data and Recipe edits. Headline gains are percentage-point improvements over the root.

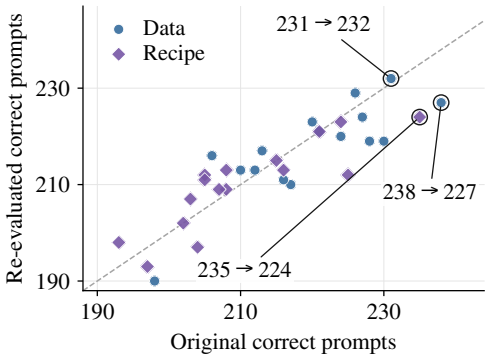

Figure 3: **IFEval checkpoint re-evaluation.** Axes report strict prompt counts. Circles denote Data candidates and diamonds Recipe candidates; the dashed line marks unchanged scores.

Verification checks the selected output. Re-evaluation examines whether the selected checkpoint retains its ranking under a revised evaluation procedure. We re-evaluate 32 IFEval adapters under a revised wrapper and runtime, holding adapter weights and recorded decoding settings fixed. Scores change by -13 to $+10$ correct prompts (Figure 3). The original Data winner falls from 238 to 227, while another adapter rises from 231 to 232 and overtakes it. The selection changes without further training, making checkpoint re-evaluation a direct check on which candidate to retain (Appendix **??**).

Re-executing configurations. Three Countdown execution pairs matched retrospectively by recorded settings score $16.00/20.70\%$, $26.00/44.50\%$, and $4.20/45.60\%$, giving gaps of 4.70, 18.50, and 41.40 pp. All six executions exceed the unadapted model's 0.85%. The pairs match on recorded data, launcher, and evaluation settings; rollout seeds, base-weight bytes, and physical execution conditions were not fully pinned. Training improves on the unadapted reference in each execution, while the performance associated with a recorded configuration changes substantially across executions.

These checks close different parts of the selection process. The Countdown reward comparison asks which checkpoint to retain from a fixed candidate pool; IFEval re-evaluation checks that choice using the saved weights; configuration re-execution tests the procedure kept for future training. They assess the selected checkpoint and its training procedure separately. We next examine search organization and edit composition under deterministic scoring.

3.4 Deterministic Control

FES finds stronger candidates under deterministic scoring. Adversarial Splay tests whether the policies differ when each configuration receives a deterministic score. Across 18 matched blocks, all policies start from the same root and receive six proposal opportunities. FES finishes above each global baseline in 14 blocks, with mean advantages of 283.8 rotations over Global Greedy and 413.9 over Global Archive (Table 3). Both paired 95% confidence intervals lie above zero. FES therefore produces stronger candidates within this short search budget. These results compare the complete policy bundles, which jointly determine parent selection, visible history, edit scope, and opportunity allocation.

Table 3: **Search results on Adversarial Splay.** Scores are mean rotation counts across 18 matched blocks; gains are relative to the shared root. Paired differences report FES minus the comparison policy, with 95% bootstrap confidence intervals obtained by resampling matched blocks. W/T/L counts FES wins, ties, and losses.

Policy	Mean rotations ↑	Gain over root	FES difference [95% CI]	FES W / T / L
FES	67,597.6	+589.6	—	—
Global Greedy	67,313.8	+305.8	283.8 [78.1, 495.0]	14 / 0 / 4
Global Archive	67,183.7	+175.7	413.9 [118.5, 703.7]	14 / 1 / 3

The advantage emerges in the initial proposal. The trajectories locate most of the FES–Greedy separation at the first proposal (Figure 4), when both policies have the root measurement but no feedback from earlier proposals. Their mean gap is already 278.9 rotations and ends at 283.8. Subsequent proposals improve both policies by similar amounts: 302.8 rotations for FES and 297.8 for Greedy. The endpoint advantage therefore primarily reflects the initial candidates generated under the two policies; both continue making comparable additional progress over the remaining five opportunities.

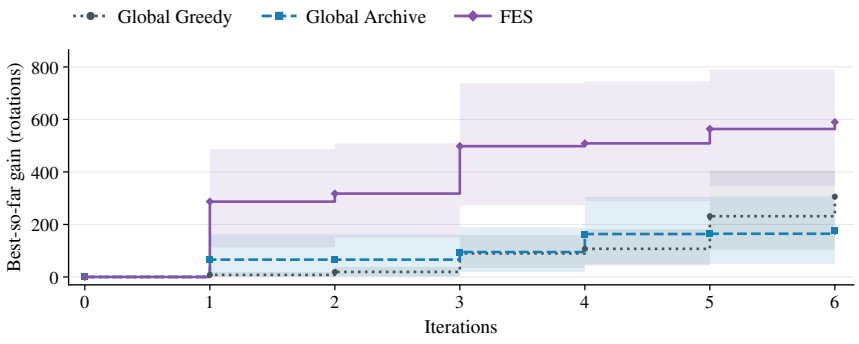

Figure 4: **Search trajectories on Adversarial Splay.** Curves show mean best-so-far gain over the root, measured in rotations, across six proposal opportunities and 18 matched blocks. Shaded bands are pointwise 95% bootstrap confidence intervals obtained by resampling complete matched blocks. Failed proposals retain the incumbent score.

Complete configurations remain the unit of selection. Separate post-search evaluations combine favorable single-surface edits in 12 composition tests. Although both constituents improve the root, 10 combinations score below the stronger constituent (Appendix D.2). Four combinations also fall

below the shared root, while two exceed their stronger constituent by 68 and 86 rotations. Because scoring is deterministic, these reversals reflect the combined access sequence itself. The control therefore identifies interactions between individually useful edits even when repeated evaluation of a configuration cannot change its score. Surface-specific search can therefore generate effective candidates, while selecting a combined configuration still requires evaluating it as a complete experiment.

4 Related Work

Autonomous post-training turns model development into sequential decision-making over costly experiments. HeuriTrain connects three lines of work: autonomous model improvement, the organization of experimental context, and reliable evidence for adaptive search.

Autonomous research and post-training. Research agents increasingly couple proposal, implementation, and evaluation in iterative loops (Jiang et al., 2025; Yamada et al., 2025). Recent systems extend this paradigm to model development: LaMDAgent searches over post-training pipelines (Yano et al., 2025), PostTrainBench evaluates model improvement under bounded compute (Rank et al., 2026), and AutoTrainess provides structured interfaces for training, evaluation, and experiment memory (Yu et al., 2026). These works establish autonomous training as a viable research setting. HeuriTrain instead focuses on the organization of the search itself: under a fixed task contract and budget, how should previous configurations, evidence, and editable surfaces determine what is tried next?

Experimental context and search organization. A growing line of work studies how accumulated experience should shape future proposals. Meta-Harness exposes prior code, scores, and execution traces for selective inspection (Lee et al., 2026); AEvo treats evolution context as mutable process-level state (Zhang et al., 2026b); and Auto Research organizes measured lineages across specialist recipe surfaces (Ning et al., 2026a). Related systems use greedy continuation or archives of prior candidates to organize code and agent evolution (Karpathy, 2026; Zhang et al., 2026a). HeuriTrain makes this organization explicit through FES, which separates parent eligibility, visible history, edit permissions, and search opportunities by surface, connecting to component-wise optimization (Potter & De Jong, 1994; Omidvar et al., 2014) while always evaluating complete Data–Recipe configurations.

Reliable evidence for adaptive experimentation. Adaptive optimization uses outcomes from earlier trials to choose subsequent experiments (Hutter et al., 2011; Snoek et al., 2012; Olson et al., 2025); autonomous research inherits the same dependence on empirical feedback (Lu et al., 2024; Yamada et al., 2025). Yet machine-learning results can vary across seeds, implementations, and training executions (Henderson et al., 2018; Bouthillier et al., 2021; Dodge et al., 2020; Mosbach et al., 2021), and agentic evaluations introduce additional trajectory-level variability (Bjarnason et al., 2026). Ning et al. (2026b) further distinguish variation across implementations from reruns of a saved artifact. Accordingly, HeuriTrain treats checkpoint re-evaluation, configuration re-execution, and repeated searches as distinct experimental units, linking each measurement to the configuration and search history that produced it.

5 Conclusion

We presented HeuriTrain, a harnessed framework for autonomous post-training under explicit task contracts and compute budgets. HeuriTrain formulates model improvement as sequential search over complete Data–Recipe configurations, making parent selection, available evidence, and edit permissions explicit. FES organizes Data and Recipe exploration through separate experimental states while retaining complete configurations as the unit of evaluation and selection. Across two post-training tasks and a deterministic control, agents improve their starting configurations through successive experiments, with policy strengths varying across settings. Analyses of agent decisions and repeated evaluations reveal how interactions between edits and execution variability shape discovered gains, connecting search organization with the reliability of its outcomes. Future work should investigate when to combine factorized exploration with joint refinement and how to allocate budgets between discovering configurations and verifying promising results.

Reproducibility Statement

Appendix A.1 specifies the task interfaces and prompt excerpts. Appendices C and D report individual post-training and Splay results, while Appendix E describes the execution and analysis protocols. Accompanying materials include reference-harness extracts, retained result records, and analysis scripts.

AI Use Statement

AI tools assisted with literature search and discovery, research ideation and experiment design, implementation and execution support, data analysis and interpretation, visualization, and drafting and polishing sections of the paper. The research agents used in the experiments are components of the studied system and are described in Section 3.1: Claude Opus for the paired IFEval and Countdown global searches, and Claude Sonnet for Adversarial Splay. AI tools were not used to generate synthetic datasets or prove mathematical claims. The authors take responsibility for the final text, claims, citations, and artifacts.

References

Anthropic. Claude 3.7 Sonnet and Claude Code, February 2025. URL https://www.anthropic.com/news/claude-3-7-sonnet.

Anthropic. Models overview, 2026. URL https://platform.claude.com/docs/en/models/overview. Accessed September 26, 2026.

Bjarni Haukur Bjarnason, André Silva, and Martin Monperrus. On randomness in agentic evals. *arXiv preprint arXiv:2602.07150*, 2026.

Xavier Bouthillier, Pierre Delaunay, Mirko Bronzi, Assya Trofimov, Brennan Nichyporuk, Justin Szeto, Naz Sepah, Edward Raff, Kanika Madan, Vikram Voleti, Samira Ebrahimi Kahou, Vincent Michalski, Dmitriy Serdyuk, Tal Arbel, Chris Pal, Gaël Varoquaux, and Pascal Vincent. Accounting for variance in machine learning benchmarks. *Proceedings of Machine Learning and Systems*, 3, 2021.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah A. Smith. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*, 2020.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. doi: 10.1609/aaai.v32i1.11694.

Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization*, pp. 507–523. Springer, 2011. doi: 10.1007/978-3-642-25566-3_40.

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. URL https://arxiv.org/abs/2502.13138.

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

Yoonho Lee, Roshen Nair, Qizheng Zhang, Kangwook Lee, Omar Khattab, and Chelsea Finn. Meta-Harness: End-to-end optimization of model harnesses. *arXiv preprint arXiv:2603.28052*, 2026. URL https://arxiv.org/abs/2603.28052.

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 preprint arXiv:2408.06292*, 2024.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. On the stability of fine-tuning BERT: Misconceptions, explanations, and strong baselines. In *International Conference on Learning Representations*, 2021.

Jingjie Ning, Xiaochuan Li, Ji Zeng, Hao Kang, and Chenyan Xiong. Auto research with specialist agents develops effective and non-trivial training recipes, 2026a. URL https://arxiv.org/abs/2605.05724.

Jingjie Ning, Shanshan Zhong, Xiaochuan Li, Ji Zeng, and Chenyan Xiong. One run is not an idea: The implementation lottery in automated research. *arXiv preprint arXiv:2607.26587*, 2026b.

Miles Olson, Elizabeth Santorella, Louis C. Tiao, Sait Cakmak, Mia Garrard, Samuel Daulton, Zhiyuan Jerry Lin, Sebastian Ament, Bernard Beckerman, Eric Onofrey, Paschal Igusti, Cristian Lara, Benjamin Letham, Cesar Cardoso, Shiyun Sunny Shen, Andy Chenyuan Lin, Matthew Grange, Elena Kashtelyan, David Eriksson, Maximilian Balandat, and Eytan Bakshy. Ax: A platform for adaptive experimentation. In *Proceedings of the Fourth International Conference on Automated Machine Learning*, volume 293 of *Proceedings of Machine Learning Research*, pp. 21/1–25. PMLR, 2025.

Mohammad Nabi Omidvar, Xiaodong Li, Yi Mei, and Xin Yao. Cooperative co-evolution with differential grouping for large scale optimization. *IEEE Transactions on Evolutionary Computation*, 18(3):378–393, June 2014. doi: 10.1109/TEVC.2013.2281543. URL https://ieeexplore.ieee.org/document/6595612.

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, 2022. doi: 10.52202/068431-2011. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html.

Jiayi Pan, Junjie Zhang, Xingyao Wang, Lifan Yuan, Hao Peng, and Alane Suhr. TinyZero. GitHub repository, 2025. URL https://github.com/Jiayi-Pan/TinyZero.

Mitchell A. Potter and Kenneth A. De Jong. A cooperative coevolutionary approach to function optimization. In Yuval Davidor, Hans-Paul Schwefel, and Reinhard Männer (eds.), *Parallel Problem Solving from Nature—PPSN III*, volume 866 of *Lecture Notes in Computer Science*, pp. 249–257, Berlin, Heidelberg, 1994. Springer. doi: 10.1007/3-540-58484-6_269. URL https://link.springer.com/chapter/10.1007/3-540-58484-6_269.

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.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300.

Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, volume 25, 2012.

Guiyao Tie, Zeli Zhao, Dingjie Song, Fuyang Wei, Rong Zhou, Yurou Dai, Wen Yin, Zhejian Yang, Jiangyue Yan, Yao Su, Zhenhan Dai, Yifeng Xie, Yihan Cao, Lichao Sun, Pan Zhou, Lifang He, Hechang Chen, Yu Zhang, Qingsong Wen, Tianming Liu, Neil Zhenqiang Gong, Jiliang Tang, Caiming Xiong, Heng Ji, Philip S. Yu, and Jianfeng Gao. A survey on post-training of large language models. *arXiv preprint arXiv:2503.06072*, 2025. URL https://arxiv.org/abs/2503.06072.

Yutaro Yamada, Robert Tjarko Lange, Cong Lu, Shengran Hu, Chris Lu, Jakob Foerster, Jeff Clune, and David Ha. The AI scientist-v2: Workshop-level automated scientific discovery via agentic tree search. *arXiv preprint arXiv:2504.08066*, 2025.

Taro Yano, Yoichi Ishibashi, and Masafumi Oyamada. LaMDAgent: An autonomous framework for post-training pipeline optimization via LLM agents. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 30078–30095. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.1529. URL https://aclanthology.org/2025.emnlp-main.1529/.

Zhaojian Yu, Penghao Yin, Shuzheng Gao, Shilin He, Kai Cai, and Xiao-Ping Zhang. AutoTrainess: Teaching language models to improve language models autonomously. *arXiv preprint arXiv:2606.31551*, 2026. URL https://arxiv.org/abs/2606.31551.

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*, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/aa5f5e6eb6f613ec412f1d948dfa21a5-Abstract-Conference.html.

Jiayi Zhang, Yongfeng Gu, Jianhao Ruan, Maojia Song, Yiran Peng, Zhiguang Han, Jinyu Xiang, Zhitao Wang, Caiyin Yang, Yixi Ouyang, Bang Liu, Chenglin Wu, and Yuyu Luo. Harnessing agentic evolution. *arXiv preprint arXiv:2605.13821*, 2026b. URL https://arxiv.org/abs/2605.13821.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023. URL https://arxiv.org/abs/2311.07911.

A Task Environments and Agent Interfaces

A.1 Editable Surfaces and Researcher Prompts

Editable artifacts. The IFEval instruction snapshot registers `tasks/data_select_ifeval/environment/select_data_env.py` for Data and `tasks/data_select_ifeval/environment/train_config.yaml` for Recipe. The selector receives a read-only pool of 50,000 rows and returns 1–5,000 unchanged pool occurrences, with multiplicity checked. Recipe edits retain the existing YAML keys and protected model, data, and output values; the registered validator supplies parameter bounds. Training and data-shuffle seeds are fixed to zero by the Harness. The quoted snapshot records a root score of $197/541$; its files are distinct from the requests for the eight-pair study.

Listing 1: IFEval task-instruction excerpts. Original wording from the Data and Recipe instruction snapshots; omitted passages are marked.

```
Edit exactly `tasks/data_select_ifeval/environment/select_data_env.py`. Do not
create, rename, stage, or commit files; do not call Claude, Bolt, Docker, a GPU
service, an evaluator, or the network. The Host supplies the incumbent and all
completed measurements in the request evidence.
[...]
The Host invokes the selector with `POOL` (a read-only 50,000-row JSON array),
`SELECTED` (output path), `MAX_SELECTED=5000`, and `SEED=0`. Each row has exactly
`id`, `messages`, and `source`. Write a deterministic non-empty JSON array of at
most 5,000 unmodified pool occurrences. Multiplicity is checked. Use only the
Python standard library; never synthesize, rewrite, merge, or fetch rows.
[...]
Only these recipe fields may change: `lora_rank`, `lora_alpha`, `lora_target`,
`lora_dropout`, `preprocessing_num_workers`, `cutoff_len`,
`per_device_train_batch_size`, `gradient_accumulation_steps`, `learning_rate`,
`num_train_epochs`, `lr_scheduler_type`, `warmup_ratio`, and `logging_steps`.
```

Paired IFEval requests. The paired study uses fresh, tool-free calls. Each request contains a frozen parent and evidence from completed earlier rounds and returns three sibling proposals. FES requests restrict the editable files and history to the scheduled surface; Archive requests permit both surfaces. Eight requests per arm provide 24 proposal opportunities. The system message is reproduced below; the task-instruction excerpt above belongs to the separate root-197 snapshot.

Listing 2: System message for the paired IFEval proposal interface.

```
You are the independent research author for one IFEval development-feedback panel. Your
    entire experimental evidence is in the supplied JSON request. Produce exactly three
    independent siblings from the frozen parent in one response. Return only JSON with a
    proposals array; do not use tools.
```

Countdown interfaces. Data edits change `tasks/countdown_data_recipe/environment/select_data.py`; Recipe edits change `tasks/countdown_data_recipe/execution/launch_grpo.sh`. The Data selector emits 1–20,000 unchanged eligible pool rows. The Recipe instruction permits bounded values at existing launcher parameters, including learning rate, KL and entropy coefficients, rollout count, and the CLI's additional registered argument keys. It prohibits adding commands or changing model, evaluator, seed, or training horizon. The retained surface-search instructions use a width-one greedy rule. The global author interface instead receives the parent selected by the Harness and permits Data, Recipe, or joint changes in one direct-child commit. Its local CPU checks precede Harness training and evaluation.

Listing 3: Countdown global candidate-author prompt, abridged. Double-brace fields are template variables.

```
You are the candidate author for one fixed-budget Countdown search slot. You are not the
search controller, scheduler, evaluator, or budget authority. Current detached HEAD is
    the Host-selected
parent ({{parent}}). Produce exactly one direct-child commit, then stop.
[...]
Only `{{SELECTOR}}` (DATA) and `{{LAUNCHER}}` (RECIPE) may change. You may change DATA,
    RECIPE, or BOTH in
the same direct-child commit. Recipe edits may alter only existing
Host-registered parameter RHS values.
```

Splay requests. Claude Sonnet supplies the proposals in all 18 matched Splay blocks. The native task registers `accesses.txt`, containing 4,096 keys in $[0, 1023]$, and the read-only evaluator `main.py`. The search request adds the permitted position scope, the parent sequence, measured history, the proposal position, and a limit of 1–64 parent-relative edits. It requests index–value replacements inside `<edits>` tags and no explanation, overriding the native whole-file output interface. FES alternates even and odd positions; the global policies may edit both.

Listing 4: Splay proposal prompt, abridged. Template fields specify the permitted scope and maximum edit count.

```
Maximize raw rotations. {{scope}}
Make a genuine experimental proposal with 1 to {{self.args.max_edits}} parent-relative
replacements. For this experiment,
ignore the upstream instruction to print the complete file: the adapter materializes it.
[...]
Return exactly one block and no explanation:
<edits>
INDEX VALUE
</edits>
Put exactly one INDEX VALUE pair on each line. Indices must be unique, and every
VALUE must differ from CURRENT_VALUE at that index. A no-op is invalid and consumes
the proposal slot.
```

Execution boundaries. The instruction and request layers specify allowed edits and available feedback. Paired IFEval calls have no tools; the Countdown coding interface permits local inspection and commits while reserving training and scoring for the Harness. The public reference harness executes trusted local code and does not provide process-level sandboxing. Recorded rationales are retained where the response interface requests them; the Splay interface returns edits alone.

A.2 Root Provenance and Reporting

The unadapted reference is the target model without the task's additional training. A trained root is a complete initial Data–Recipe configuration evaluated before search. The selected endpoint is the highest-scoring measured complete configuration, including that root. Search gains are endpoint minus root; initialization gains are root minus the unadapted reference, in percentage points. Matched searches reuse the same root.

Table 4: **Recorded root configurations.** IFEval entries give strict-correct prompt counts; Countdown gives strict Pass@1.

Record or study	Root score
IFEval single-search discovery (Table 1) / Instruction snapshot	197/541
IFEval eight-pair matched comparison (Table 2 & 5)	187/541
Countdown adopted `m0_v2`	41.00%

The supplied instructions start from registered roots. The IFEval Recipe instruction freezes the canonical Data selection; Countdown starts from the adopted `m0_v2` launcher. Root-initialization requests are separate artifacts from these search instructions[cite: 1]. Specifically, the individual

discovery trajectories in Table 1 (and the decision traces in Figure 2) start from the canonical initial configuration snapshot $x_0^{(197)}$ ($197/541$, 36.41%), against which their headline gains are computed (+12.94 pp for Greedy, +9.25 pp for Archive, and +6.66 pp for FES)[cite: 1]. In contrast, the eight-pair matched study in Section 3.2 evaluates policies from a shared standardized root $x_0^{(187)}$ ($187/541$, 34.57%) under uniform tool-free slot contracts[cite: 1]. Each study strictly evaluates gains relative to its own registered root, preserving independent evaluation integrity across cohorts.

Registered root settings. The root-$197/541$ IFEval configuration uses $5{,}000$ of $50{,}000$ examples, one epoch, learning rate 10^{-4}, batch size two with four accumulation steps, and a $2{,}048$-token cutoff[cite: 1]. LoRA rank/alpha are $16/32$, with dropout 0.05 on all target modules; training uses bf16 in LLaMA-Factory[cite: 1]. Countdown's root uses $20{,}000$ examples, learning rate 10^{-6}, 200 updates of 16 prompts, four rollouts per prompt, a $1{,}024$-token response limit, KL coefficient 0.001, and zero entropy coefficient[cite: 1]. The Splay root repeats the 10-bit bit-reversal ordering four times[cite: 1].

B Search Policies and Execution

B.1 Harness loop

Algorithm 1 Budgeted search over complete post-training configurations

Input Registered task $\mathcal{E}$, root x_0, research agent A, search policy q.
Init Train and evaluate x_0 from θ_0; initialize $\mathcal{L}$ with its configuration, checkpoint, score, and registered resource charges.
while an opportunity remains and the registered budget permits execution **do**
  $(p_t, \mathcal{M}_t, \mathcal{U}_t, b_t) \leftarrow q(\mathcal{L}, B)$; render o_t for A.
  Receive edit δ_t and any accompanying rationale; check permissions, validity, and duplicates.
  if the proposed edit is accepted **then**
    $x_t \leftarrow \Gamma(x_{p_t}, \delta_t)$; train from θ_0; evaluate the final checkpoint.
    if execution succeeds **then** attach checkpoint, score, and diagnostics;
    else mark failure without a measured score; **end if**
  else mark rejection without a measured score; **end if**
  Append parent, request, edits, outcome, and cost to $\mathcal{L}$; charge the attempt.
  At the feedback boundary, update policy state from committed records.
end while; return $(x_{\hat{t}}, \theta_{\hat{t}}^{\mathrm{final}})$ for the best measured ledger entry.

B.2 Global Parent Selection

Both global policies retain an archive of valid measured complete configurations. Greedy selects the current best-scoring configuration; Archive samples uniformly from the eligible archive. The latter includes the root and lower-scoring configurations. The historical implementation uses a seeded SHA-256 pseudorandom draw over an ASCII-sorted archive; score and child count do not enter its sampling rule. The request contains the selected parent's configuration and the history admitted by the registered renderer. Algorithm 2 gives the global policy transition for one-proposal requests; panel-based studies freeze the parent and evidence for all siblings before execution.

Algorithm 2 Global Greedy and Global Archive under a registered task contract

Input Measured root x_0, search policy q, search seed s, registered budget B.
Init Archive $\mathcal{A} \leftarrow \{x_0\}$; incumbent $x^\star \leftarrow x_0$.
while the registered budget permits another request **do**
  if $q = \text{Greedy}$ **then** $p \leftarrow x^\star$;
  else sample p uniformly from $\mathcal{A}$ using the registered seeded rule.
  Render the parent, permitted archive history, edit scope, and remaining budget.
  Invoke the research agent; receive the proposed complete candidate.
  Validate; train and evaluate if accepted; record the disposition and charge the attempt.
  if the candidate x is valid, unique, and measured **then**
    $\mathcal{A} \leftarrow \mathcal{A} \cup \{x\}$.
    if $y(x) > y(x^\star)$ **then** $x^\star \leftarrow x$.
  end if; advance to the next request.
end while; **return** $x^\star$ and its final checkpoint.

C Post-Training Results

C.1 Eight Paired IFEval Searches

The study contains eight seed-matched FES–Global Archive pairs from the $187/541$ root. Each arm receives eight rounds of three proposals. Training and shuffle seeds are zero; search seeds vary across pairs. All 16 searches use their 24 opportunities. FES records 192 scored candidates; Archive records 169, with 14 failed and nine duplicate positions. Table 5 lists every selected endpoint.

Table 5: All eight IFEval search pairs. Entries are strict-correct prompts out of 541; the trained root is $187/541$. Differences use the integer counts before rounding and are in percentage points (FES minus Global Archive). Seed labels 1–8 index the paired runs; original random seeds are retained in the accompanying run metadata.

Seed index	FES	Global Archive	Difference
1	228	253	-4.62
2	258	246	$+2.22$
3	262	242	$+3.70$
4	251	248	$+0.55$
5	267	250	$+3.14$
6	255	239	$+2.96$
7	260	245	$+2.77$
8	264	247	$+3.14$

The mean paired difference is 1.73 pp, with a 95% interval of $[-0.25, 3.10]$ pp. The unadapted reference scores $256/541$ across 541 prompts and 834 instructions. Rescoring its saved responses with the frozen verifier reproduces all four aggregate metrics.

C.2 Executed IFEval Data–Recipe Crossing

The Data edit adds response-ending checks; the Recipe edit changes LoRA rank/alpha from $32/64$ to $64/128$. The parent is evaluated in the preceding round; the three children are proposed together, with one execution per cell. File hashes match the corresponding single-surface and joint configurations. The difference of contrasts is $(248 - 266) - (267 - 259) = -26$ correct prompts.

Table 6: **IFEval Data–Recipe crossing.** Entries are strict-correct prompt counts out of 541. Rows specify the Data configuration and columns the Recipe configuration; each cell contains one execution.

	Parent Recipe	Edited Recipe
Parent Data	259	266
Edited Data	267	248

C.3 Countdown Campaigns and Surface Endpoints

The two archived campaigns, labeled Batch 1 and Batch 2, train Qwen2.5-3B for 200 updates and evaluate strict Pass@1 on 2,000 public-development problems. Both use training seed zero and the 41.00% root. Batch 2 is a fresh adaptive search. Candidate identifiers are local to each batch and branch. The Recipe-only reference edits the training recipe and is separate from the CD-B1 Global Greedy and Global Archive searches. Table 7 lists the retained endpoints.

Table 7: **Countdown surface endpoints.** Scores are strict Pass@1 (%); gains are percentage points over the 41.00% root. Changes describe the retained configuration relative to root. Batch labels identify separate adaptive searches.

Batch	Branch	Candidate	Changes from root	Score	Gain
1	FES Data	`p21`	Exact-use filter and solution floor; increased exposure of selected addition/subtraction rows with four numbers	49.40	+8.40
1	FES Recipe	`c10`	Rollouts $4 \to 6$; GRPO→REINFORCE++ baseline	49.40	+8.40
1	Recipe-only	`c12`	Learning rate $10^{-6} \to 2 \times 10^{-6}$	55.50	+14.50
2	FES Data	`p74`	1,600 rows; four-number draw share $\approx 0.497 \to 0.75$	44.20	+3.20
2	FES Recipe	`c04`	Learning rate $10^{-6} \to 3 \times 10^{-6}$; rollouts $4 \to 8 \to 12$	57.10	+16.10
2	Recipe-only	`c09`	GRPO→RLOO	45.10	+4.10

C.4 Countdown Candidate Records

Batch 1's Data branch records 15 paid attempts, including four failures before training and one byte-identical repeated execution. Batch 2 records 11 Data and six Recipe candidates, all trained and scored. Tables 8–10 give the parent, submitted intervention, score, and recorded outcome. "Promoted" and "retained" indicate updates to the branch incumbent and the final retained candidate, respectively. Other measured candidates are marked "not promoted".

Table 8: **Batch 1: FES Data records.** Interventions are relative to the listed parent. Scores are strict Pass@1 (%); a dash denotes a failure before training. The repeated execution `p12` reuses `p03`'s configuration.

ID	Parent	Intervention	Score	Outcome
`p01`	root	Drop every proper-subset-solvable row; 16,516 rows	—	Pre-training failure
`p02`	root	Drop rows labeled trivial shortcut; 17,858 rows	—	Pre-training failure
`p03`	root	Exact-use filter and `MIN_SOLUTIONS=2`; 13,418 rows	48.25	Promoted
`p04`	`p03`	Per-arity solution floors $k_3 \geq 3$, $k_4 \geq 32$; 9,538 rows	8.85	Not promoted
`p05`	`p03`	Match arity mix to public development; 11,287 rows	—	Pre-training failure
`p06`	`p03`	Two-sided development-calibrated pass-rate band; 4,796 rows	—	Pre-training failure
`p08`	`p03`	Same predicate; first 800 rows in hash order	43.20	Not promoted
`p09`	`p03`	Drop rows solvable using only addition/subtraction	1.80	Not promoted
`p10`	`p03`	Relax exact-use filter: drop shortcuts omitting at least two numbers	21.10	Not promoted
`p11`	`p03`	Remove the final row in hash order	13.70	Not promoted
`p12`	`p03`	Byte-identical repeated execution	47.00	Replicate
`p14`	`p03`	Lower only the three-number solution floor: {3:1, 4:2}	17.65	Not promoted
`p15`	`p03`	Add two copies of each addition/subtraction-solvable four-number row	48.80	Promoted
`p20`	`p15`	Apply the same ×3 exposure to class `("mult",3)`	37.70	Not promoted
`p21`	`p15`	Add a fourth copy for 1,133 addition/subtraction four-number rows with one subtraction route	49.40	Retained

Table 9: **Batch 2: FES Data records.** Scores are strict Pass@1 (%). Each intervention is applied to its listed parent and evaluated with the root Recipe.

ID	Parent	Intervention	Score	Outcome
p71	root	Drop proper-subset-solvable rows; $20{,}000 \to 16{,}516$ rows	21.70	Not promoted
p72	root	Top 35% by winnability within each arity; $6{,}999$ rows	16.55	Not promoted
p73	root	Selection size $20{,}000 \to 1{,}600$; two draws per row	41.75	Promoted
p74	p73	Four-number share $\approx 0.497 \to 0.75$; quota $1{,}200/400$	44.20	Retained
p75	p74	Four-number share $0.75 \to 1.0$	12.40	Not promoted
p76	p74	Selection size $1{,}600 \to 3{,}200$; one draw per row	5.20	Not promoted
p77	p74	Selection size $1{,}600 \to 800$; four draws per row	15.65	Not promoted
p78	p74	Salted-order redraw with composition held fixed	5.30	Not promoted
p79	p74	Purge proper-subset-solvable rows with composition held fixed	34.40	Not promoted
p80	p74	Four-number share $0.75 \to 0.8125$; quota $1{,}300/300$	3.40	Not promoted
p81	p74	Disjoint second row block with composition held fixed	41.20	Not promoted

Table 10: **Batch 2: FES Recipe records.** Scores are strict Pass@1 (%). Rows follow the six serial Recipe proposals. Each candidate uses the root Data configuration.

ID	Parent	Intervention	Score	Outcome
c01	root	Learning rate $10^{-6} \to 3 \times 10^{-6}$	44.35	Promoted
c02	c01	Rollouts $4 \to 8$	48.70	Promoted
c03	c02	Learning rate $3 \times 10^{-6} \to 6 \times 10^{-6}$	6.05	Not promoted
c04	c02	Rollouts $8 \to 12$	57.10	Retained
c05	c04	Rollouts $12 \to 16$	8.30	Not promoted
c06	c04	Entropy coefficient $0 \to 0.001$	24.05	Not promoted

C.5 Countdown Evaluation and Candidate Selection

Training-reward selection uses the mean reward over updates 151–200; the root has mean training reward 0.352 and strict Pass@1 41.00%. Strict evaluation reads the last answer block and requires every supplied number exactly once.

C.6 Countdown Unadapted Reference

The unadapted Qwen2.5-3B scores 0.85% $(17/2{,}000)$ using the same ordered items and prompt template, greedy decoding, a $1{,}024$-token limit, bfloat16, and the strict scorer. Independent rescoring agrees on every new response and reproduces the scores of all $8{,}000$ retained responses from root and three trained checkpoints. The new CRC runtime uses TP=1, vLLM 0.8.5.post1, PyTorch 2.6.0+cu124, and Transformers 4.52.4; the recorded discovery runtime uses TP=2, vLLM 0.15.1, PyTorch 2.9.1, and Transformers 4.57.6. The base-model revision is `3aab1f1954e9cc14eb9509a215f9e5ca08227a9b`; its configuration hash matches the recorded model configuration, and historical weight hashes are unavailable. Complete responses, item hashes, and runtime manifests are retained with the results.

D Adversarial Splay Results

D.1 Matched Search Endpoints

Each of 18 matched blocks contains one search per policy, with six proposal opportunities per search and a shared root of $67{,}008$ rotations. The task uses $4{,}096$ accesses with keys in 0–$1{,}023$. FES allocates three proposals to even positions and three to odd positions. Of 324 proposal opportunities, 320 produce scores; four unscored attempts retain the preceding best score.

Table 11: **Adversarial Splay endpoints.** Rows are matched search seeds; columns report the best root-inclusive rotation count after six proposals. Differences are FES minus the indicated global policy. Seed indices are shared with Table 12.

Seed	Global Greedy	Global Archive	FES	Δ Greedy	Δ Archive
1	67,093	67,008	67,008	−85	+0
2	69,015	67,008	68,315	−700	+1,307
3	67,203	67,017	67,862	+659	+845
4	67,150	67,027	67,358	+208	+331
5	67,058	67,008	68,243	+1,185	+1,235
6	67,246	67,042	68,292	+1,046	+1,250
7	67,578	67,306	67,693	+115	+387
8	67,289	67,158	67,233	−56	+75
9	67,287	67,008	68,006	+719	+998
10	67,060	67,008	67,258	+198	+250
11	67,025	67,836	67,037	+12	−799
12	67,067	67,098	67,147	+80	+49
13	67,579	67,008	67,990	+411	+982
14	67,272	67,008	67,136	−136	+128
15	67,544	67,081	68,243	+699	+1,162
16	67,027	67,008	67,133	+106	+125
17	67,147	67,942	67,197	+50	−745
18	67,008	67,736	67,606	+598	−130
Mean	67,313.8	67,183.7	67,597.6	+283.8	+413.9

D.2 Composed Configurations

After search, the selected even-position and odd-position edits are combined and evaluated. Table 12 lists the 12 blocks in which both surface winners exceed the root. These evaluations are separate from the six-proposal searches. The joint score is below the stronger singleton in 10 of these blocks and below the root in four; the other two joint scores exceed the stronger singleton by 68 and 86 rotations.

Table 12: **Adversarial Splay composition results.** Even and Odd are the selected single-surface scores; Joint evaluates their combined edits. Rows include every block in which both single-surface scores exceed the 67,008-rotation root. The last two columns subtract the stronger single-surface score and the root from Joint, respectively.

Seed	Even	Odd	Joint	Δ stronger	Δ root
2	68,315	67,016	68,204	−111	+1,196
4	67,358	67,179	66,853	−505	−155
5	68,243	67,237	67,463	−780	+455
6	68,292	67,476	66,914	−1,378	−94
7	67,693	67,281	67,309	−384	+301
8	67,233	67,166	66,352	−881	−656
10	67,190	67,258	67,326	+68	+318
12	67,128	67,147	67,100	−47	+92
13	67,990	67,338	65,101	−2,889	−1,907
14	67,136	67,129	67,222	+86	+214
15	68,243	67,050	68,069	−174	+1,061
18	67,151	67,606	67,508	−98	+500

E Experimental and Analysis Protocols

E.1 Budgets and Feedback Units

An opportunity is charged when a proposal is requested, including invalid submissions, duplicate configurations, and failed executions. An evaluated candidate has a measured outcome. Panel requests share one frozen parent and pre-round history across sibling proposals. In the paired IFEval study, FES alternates Data and Recipe rounds; both policies sample uniformly within their eligible archives.

In Table 1, "Evaluated" reports the exact count of unique, executable complete configurations that successfully finished training and received formal checkpoint scoring before reaching search convergence or the allocated budget. While Table 13 specifies outer envelope bounds (e.g., up to 38 author slots across five rounds for root-197 IFEval), actual executed candidates were 20, 21, and 18 for Greedy, Archive, and FES, respectively, due to Harness-level early-stopping on converged branches and deduplication of redundant proposals. Similarly, Countdown FES reports the 17 fully trained and evaluated candidates (11 Data and 6 Recipe checkpoints) that reached completion under its compute cap.

Countdown allocates 32 H100-hours to each FES surface. The recorded root-197 IFEval searches use ten panel calls for Greedy and 38 individual author slots for FES. IFEval follows the candidate's epoch count; Countdown uses the final checkpoint after 200 updates.

Table 13: **Registered search budgets.** Limits are per search unless specified otherwise. GPU-hour entries retain their recorded hardware units.

Study	Proposal allowance	Compute cap
IFEval, root $197/541$	38 in five rounds	30 H100-h
IFEval, eight-pair study	24 in eight rounds	14 H200-h
Countdown global searches	25	64 H100-h
Countdown full FES searches	Within compute cap	64 H100-h
Countdown Recipe-only search	Within compute cap	32 H100-h
Adversarial Splay	Six per policy per block	CPU evaluation

E.2 Paired IFEval Statistical Protocol

Differences are computed from strict-correct counts before conversion to percentages. The statistical unit is a complete seed-matched search pair. The percentile-bootstrap interval uses the full discrete distribution of resampling eight pairs with replacement. Sample standard deviations use eight selected endpoints per policy. The exact two-sided mean sign-flip test enumerates all 2^8 assignments and gives $p = 0.1172$, under sign exchangeability.

E.3 Saved-Checkpoint Evaluation Protocol

The re-evaluation uses 32 saved IFEval adapters, with the adapter weights and recorded decoding settings fixed: Qwen2.5-3B-Instruct, bf16 LoRA, greedy generation at temperature zero, and a 1,280-token limit. The later evaluation uses a revised wrapper and runtime and retains generated responses. No additional training is performed for this comparison. Figure 3 reports the paired scores.

E.4 Adversarial Splay Statistical Protocol

The statistical unit is a matched block containing all three policies. Endpoint differences are computed within each of the 18 blocks before aggregation. The main-text 95% endpoint intervals use 100,000 paired-block percentile-bootstrap resamples. Each resample retains the three policy outcomes from a sampled block. Exact two-sided sign-flip tests enumerate all 2^{18} sign assignments to each vector of paired endpoint differences, including zero differences. Holm correction is applied jointly to the FES-Greedy and FES-Archive contrasts; both adjusted p-values are 0.0354.