ATHANOR: AN AGENT-NATIVE SYSTEM FOR AUTONOMOUS POST-TRAINING RESEARCH

Anonymous authors
Paper under double-blind review

ABSTRACT

Post-training research is an iterative process of diagnosing model weaknesses, testing data and algorithmic recipes, evaluating outcomes, and refining subsequent experiments from evidence. We build **Athanor**, an open agent-native system that enables frontier generalist agents to hill-climb a raw base model on target benchmarks through long-running campaigns of training experiments on real GPUs. The agent develops hypotheses and submits recipes, while system services run training, protect evaluation, and link recipes, code, artifacts, and research notes into a persistent research record. This lets the agent work on the research process itself, from hypothesis formation and experimental design to result analysis and iterative refinement. Across six successive campaigns on eight-GPU nodes, Opus 5 improves Qwen3-4B-Base from **26.4** to **63.2** across seven benchmarks using a fixed, decontaminated data pool, against 70.4 for the official instruction-tuned model. We analyze the record of these campaigns: 839 accepted experiments, 711 evaluations, and nearly a thousand research notes. We observe that the agent keeps to a narrow band of recipes, changing one setting at a time over a few trusted corpora. Its largest losses come rarely from failed jobs and mostly from data it did not inspect and the intrinsic difficulty of post-training: the same recipe can help one checkpoint but hurt another. On the strategy side, it drops early conclusions that later evidence contradicts, while directions it rules out mostly stay ruled out.

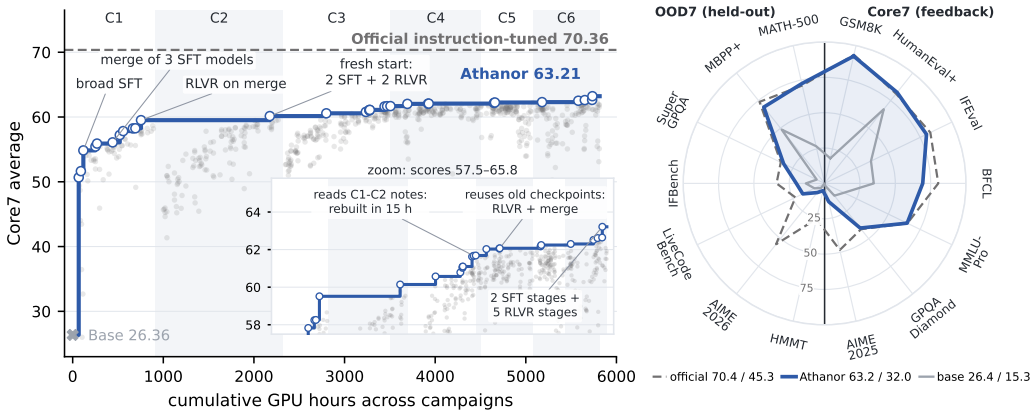


Figure 1: **Autonomous hill climbing across campaigns.** A campaign is one research period in which a researcher agent uses its allocated resources to improve the base model. Left: each faint dot is one Core7 evaluation from the six successive campaigns (C1–C6), placed end to end by cumulative GPU hours. The blue staircase is the best score so far, and the gray dashed line is the official model. Right: Core7 is the suite of seven benchmarks the researcher agent receives as feedback to improve its recipe design; OOD7 is a suite of seven held-out benchmarks evaluated once after the last campaign. The radar shows per-task scores of the base, the Athanor best, and the official model under the same evaluator (Core7 right, OOD7 left); the legend gives their Core7 / OOD7 averages.

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.

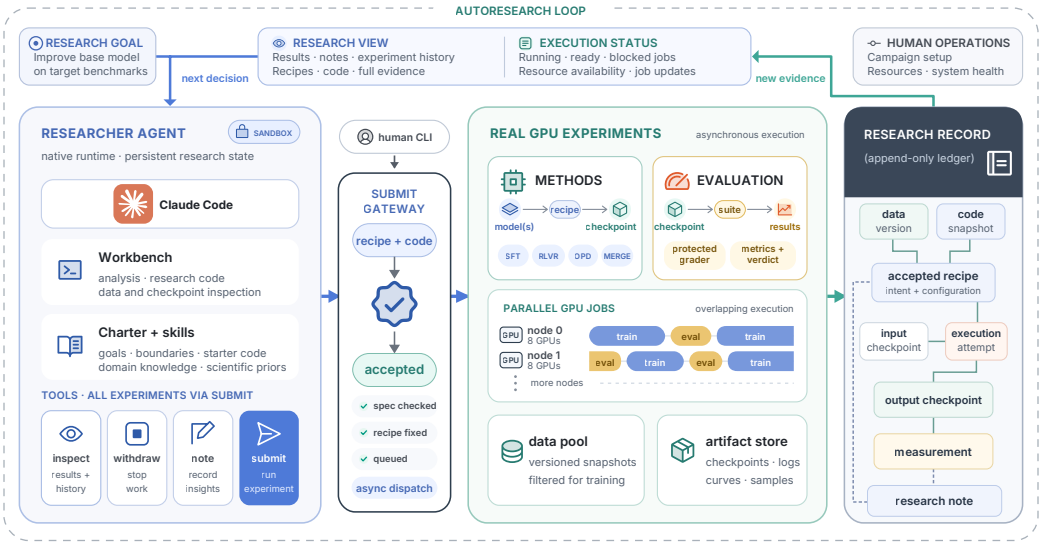


Figure 2: **Athanor at a glance.** The researcher agent reads previous results and proposes training recipes; system services execute experiments asynchronously on real GPUs, evaluate the resulting models, and return evidence for the next decision, while a research record connects proposals, execution, results, and research notes across agent sessions. Humans set the research goal and manage resources and system health. Solid record links trace execution; dashed links show note references.

1 Introduction

Recursive self-improvement (RSI), in which an AI system uses its own capabilities to build and train a more capable successor (Good, 1966; Schmidhuber, 2007), is becoming increasingly tangible with frontier generalist coding agents (Anthropic, 2026a; OpenAI, 2026a; Kimi Team, 2026) that can operate across much of the software stack (Jimenez et al., 2024; Yang et al., 2024; Merrill et al., 2026). For language models, pretraining establishes the foundation (Kaplan et al., 2020; Hoffmann et al., 2022; Grattafiori et al., 2024), while post-training shapes instruction following, reasoning, coding, and alignment (Lambert, 2026; OpenAI, 2024; Lambert et al., 2025; Yang et al., 2025; Guo et al., 2025). Yet post-training research remains an expensive iterative search: existing works typically document the final recipe rather than the trajectory that produced it (Qwen Team, 2026; Yang et al., 2026; DeepSeek-AI, 2026). Searching over data mixtures, objectives, algorithms, and hyperparameters is precisely the evidence-driven loop that has been automated (Lu et al., 2026; Novikov et al., 2025; Chen et al., 2026d) and evaluated (Chan et al., 2025; Wijk et al., 2025; Lyu et al., 2026).

We study **autonomous post-training research**, where a frontier agent diagnoses a base model's weaknesses, runs training experiments at scale, and iteratively refines its recipes from the evidence. Existing benchmarks study *only bounded forms* of this problem. PostTrainBench optimizes a separate benchmark *specialist* under a ten-hour, single-GPU budget (Rank et al., 2026); other benchmarks and systems isolate fine-tuning, reinforcement-learning engineering, or data-centric research (Li et al., 2026b; Chen et al., 2026c; Meng et al., 2026; Du et al., 2026). Complementary systems provide training workflows and infrastructure for larger-model campaigns (Yu et al., 2026; Shi et al., 2026b). Locus extends PostTrainBench to parallel, multi-day search and continues improving after CLI baselines plateau (Intology, 2026). Our target problem is the general one that frontier post-training faces: one *generalist* checkpoint must improve across seven capabilities at once, from a raw base model, over a fixed data pool, on a multi-node fleet, rather than one specialist per benchmark on one GPU. Thus, prior work covers only parts of this problem; ours is the more realistic setting, which exposes the agent's true *research* bottleneck. Testing one hypothesis may require hours of failure-prone GPU execution before it can be evaluated. Long-running search must also preserve data integrity and separate online feedback from sealed final evaluation (Xu et al., 2024; Wang et al., 2026b). To accelerate this process, we need a system that makes such experiments easy for the agent to run and keeps the evidence it saw, the recipe it proposed, the experiment that ran, and the result connected across sessions.

Thus, we build **Athanor**, an open, asynchronous agent-native system that enables frontier agents to hill-climb a raw base model through long-running campaigns of training experiments on real GPUs. Our design thesis is that, *as agent intelligence improves, the bottleneck in autonomous research shifts from generating plausible ideas to reliably executing and verifying them*. The central design question is *how to empower agents, not imprison them*, giving them freedom over research decisions while keeping experiment execution *efficient, auditable, and recoverable*.

Athanor follows three design principles. **First, separate research decisions from execution.** An off-the-shelf researcher inspects data and model behavior, develops interventions, and submits executable recipes through a small tool interface. A control plane manages resources, execution, measurement, and recovery, and returns the evidence. **Second, preserve research state across sessions.** We keep recipes, code, data versions, checkpoints, and measurements linked to the researcher's notes and revisions, so a later decision can revisit both the evidence and its interpretation. **Third, keep delayed experiments productive.** Training and evaluation run asynchronously, and completed work updates the research state without waiting for slower branches. A campaign therefore returns a model and an auditable trajectory of how it was obtained.

We evaluate Athanor in six successive autonomous **large-scale post-training** campaigns on a multi-node GPU fleet. Starting from Qwen3-4B-Base, the agent searches over data, methods, and hyperparameters from a fixed, decontaminated pool. Three campaigns from the base reach 59.5, 60.1, and 61.7 on seven benchmarks within a day each; campaigns that inherit the research record and retained checkpoints reach 63.2, against 70.4 for the official instruction-tuned model under the same evaluation protocol (Figure 1). On seven held-out benchmarks, the best checkpoint improves from 15.3 to 32.0, against 45.3 for the official model. We analyze the record of these campaigns, 839 accepted experiments, 711 evaluations, and nearly a thousand research notes, and find (Section 4.3): (1) *local search*: the agent changes one setting at a time around a few trusted recipes; (2) *recipe convergence*: independent campaigns converge to the same recipe structure; (3) *unexamined data*: the agent's largest losses come from training data it did not inspect, rarely from failed jobs; (4) *lineage awareness*: the agent learns that the same recipe can help one checkpoint and hurt another; and (5) *self-correction*: the agent retracts early conclusions and re-measures small differences, while directions closed on evidence stay closed. With execution handled, the agent's prior sets what it tries, and verification and inspection set how far it gets. Athanor thus provides a foundation for studying **autoresearch scaling**, the growth of results with experimental compute (Section 2).

2 On Autonomous Post-Training Research

2.1 Problem Formulation

Given a base model and an experimental budget, autonomous post-training research searches over executable recipes r, each specifying data, objective, method and program, hyperparameters, stage order, and initial checkpoint. Executing the recipe yields a checkpoint $M = \mathrm{Exec}(r)$. Since a recipe may start from any completed checkpoint, the search branches into lineages that compose supervised fine-tuning (SFT), reinforcement learning with verifiable rewards (RLVR), distillation, and merging (Lambert, 2026; Lambert et al., 2025; Yang et al., 2025; Guo et al., 2025; Yang et al., 2026). Let $\mathcal{P}_{t-1}$ denote the pool of checkpoints available before decision t, with $\mathcal{P}_0$ containing only the base model, and let $\mathcal{R}(\mathcal{P}_{t-1})$ denote the recipes admissible on this pool. Let the search objective $c(M)$ denote the latent capability of checkpoint M; it is not directly observable and is estimated by evaluation on a finite benchmark suite $\mathcal{B}$. A researcher agent acts as a policy π over recipes, conditioned on the history $\mathcal{H}_{t-1}$ of recipes, programs, measurements, failures, and notes. The research loop is:

$$\begin{aligned} &\text{RESEARCH} && r_t \sim \pi(\cdot \mid \mathcal{H}_{t-1}), && r_t \in \mathcal{R}(\mathcal{P}_{t-1}), \\ &\text{EXPERIMENT} && M_t = \mathrm{Exec}(r_t), && \widehat{\mathbf{p}}_t = \mathrm{Eval}_{\mathcal{B}}(M_t; \xi_t), \\ &\text{OBJECTIVE} && M_T^\star \in \arg\max_{M \in \mathcal{P}_T} c(M). \end{aligned} \tag{1}$$

At step t, the policy proposes a recipe, execution produces a checkpoint, and evaluation returns its benchmark score vector $\widehat{\mathbf{p}}_t$, where ξ_t denotes evaluation randomness. The pool grows as $\mathcal{P}_t = \mathcal{P}_{t-1} \cup \{M_t\}$, and the result and notes enter $\mathcal{H}_t$. The objective is the checkpoint of highest latent capability in the final pool, whereas the researcher observes only $\widehat{\mathbf{p}}_t$. Because $\widehat{\mathbf{p}}_t$ holds one score

per benchmark, a recipe can raise one capability while lowering another, which makes the observed search *multi-objective* (Sener & Koltun, 2018).

2.2 What Makes Autonomous Post-Training Hard

Combinatorial, state-dependent search. The research line of Equation (1) chooses a recipe from a high-dimensional, compositional space: discrete choices of data and method, continuous hyperparameters, and conditional stages. Gradient descent optimizes parameters within a stage; the researcher decides the order of training stages and their starting checkpoints. The choice is state-dependent: each stage trains the checkpoint produced by earlier stages, so upstream stages condition what later stages can achieve (Zhang et al., 2026a), and each new checkpoint enlarges the admissible set $\mathcal{R}(\mathcal{P}_t)$. Since the score is not differentiable with respect to the recipe, the researcher can only modify a promising recipe, observe a noisy score, and keep the change if it improves: the outer loop is **stochastic black-box hill climbing** (Russell & Norvig, 2021). A researcher agent's knowledge acts as a strong *prior* that concentrates proposals in a small effective region of the recipe space. To propose the next recipe within that region, the agent needs *dense feedback* that is informative enough to act on, and because a recipe's effect depends on where it starts, that feedback is the full lineage behind each score, not an excerpt or a summary of it.

Delayed, noisy, and incomplete feedback. The experiment line returns that feedback. Unlike NanoGPT speedrunning, where every run is scored against a common validation cross-entropy target (Jordan et al., 2024), post-training stages optimize objectives whose losses share no scale, and equal loss need not rank downstream behavior (Chen et al., 2026b). To evaluate a recipe, we must first train it, so there is a delay $\tau_{\text{fb},t} = \tau_{\text{exec}}(r_t) + \tau_{\text{eval}}(M_t, \mathcal{B})$, the time to prepare data and train *plus* the time to evaluate the checkpoint on $\mathcal{B}$. The judgment is noisy: finite evaluation can blur the order of nearby candidates (Madaan et al., 2024). It is also incomplete: GSM8K's grade-school problems cover only part of mathematics (Cobbe et al., 2021). The signal is thus *delayed, noisy, and incomplete*.

Scaling the research process. The objective line indexes the result by T, the number of experiments. Pretraining scaling laws relate loss to compute for a fixed training procedure (Kaplan et al., 2020; Hoffmann et al., 2022), and similar curves can be fit for RL post-training once the recipe is fixed (Khatri et al., 2026). Once the recipe changes, no such stable law is known for post-training. Here the recipe is what the agent is searching for, so the question is how $c(M_T^\star)$ grows as experiments, and the compute they consume, accumulate to improve overall performance. We call growth along this axis **autoresearch scaling**, where agent inference, already a large research expense (OpenAI, 2026b), buys better analysis, and compute on training accumulates research artifacts: a growing set of checkpoints and the scientific trajectory that produced them.

3 The Athanor Autoresearch Loop

We build **Athanor**, which creates the autoresearch loop of Section 2 around a researcher and a persistent control plane that share a research record (Figure 2).

3.1 The Research Loop

Algorithm 1 and Figure 2 show the overall autoresearch loop, which comprises a shared history, a research loop, and an execution control plane. The shared `history` stores the research record (Figure 2, right) of recipes, checkpoints, measurements, and notes. In the research loop, the researcher agent (left) reads the record with `inspect`, analyzes it in the workbench, develops a recipe guided by its charter and skills, and calls `submit`; the submit gateway fixes the recipe, programs, inputs, and required measurements and leaves execution to the background. The control plane records

Algorithm 1: The Athanor autoresearch loop.

```
history = persistent_record()

# Researcher
while research_open():
    evidence = inspect(history)
    recipe = agent.develop(evidence)
    submit(recipe, history)

# Control plane (runs independently)
while execution_open():
    updates = collect_job_updates()
    history.record(updates)
    # Prepare, train/merge, eval, store.
    for job in ready_work(history):
        launch_async(job)
```

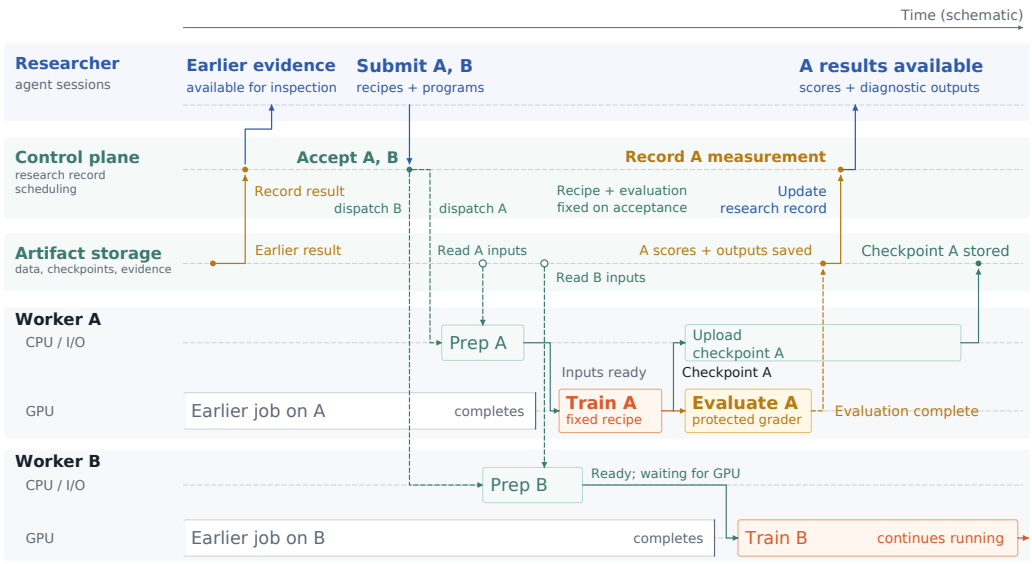


Figure 3: **Autoresearch execution flow.** The researcher submits recipes and receives feedback within an ongoing campaign. Recipe preparation and GPU training run asynchronously, so GPU training does not block recipe generation. For example, B waits for its GPU while A's completed checkpoint feeds protected evaluation and upload in parallel, and A's measurement becomes available while its upload and B's training continue. Bars show computation; dashed arrows show job requests or data.

every job update in the same history and dispatches the GPU experiments (center) as inputs and resources become available: pool data feeds a training method, and a finished checkpoint is evaluated by the protected grader and stored. New evidence returns to the researcher on arrival.

3.2 Post-Training Research Environment and Campaign Goal

We study a generalist setting in which one checkpoint must improve across a suite of target capabilities, starting from a raw base model. The environment gives the researcher a coding workspace, a fixed pool of training corpora filtered against the benchmarks, reusable checkpoints, and starter code for SFT, RLVR, and distillation on the verl trainer (Sheng et al., 2025) and for checkpoint merging. To test research ability in a controlled world, the researcher cannot use the Internet or an external teacher model. Evaluation is protected: an online suite returns every benchmark score after each experiment, and a held-out suite stays sealed until search ends. Human operators set the campaign goal, the measurement protocol, operational rules for GPU use, and the compute budget (Appendix D).

3.3 Keeping the Loop Fast and Reliable

The three design principles of Section 1 keep the loop of Algorithm 1 running over days and answer the demands of Section 2. Runtime details appear in Figure 5 and Appendix C.

Separate research decisions from execution. The control plane owns the tedious recurring work: resource allocation, input preparation, checkpoint storage, evaluation, and recovery. The researcher uses native coding and analysis tools to choose data, objectives, hyperparameters, and checkpoint composition, interpret results, and develop data processing, training programs, and efficiency changes, starting from backend defaults, starter code, and optional skills. A frozen evaluator is protected from researcher modification; scores, item-level outcomes, and diagnostic logs follow the campaign's disclosure rules, and a sealed final suite (OOD7) tests the finalists once. Operators maintain platform health; within these boundaries the researcher retains control over scientific decisions.

Preserve research state across sessions. The research record, an append-only ledger, links each accepted experiment's recipe, code, data versions, and input checkpoints to execution attempts, output

artifacts, and measurements; checkpoint references form a lineage DAG in which one output can seed several branches and a merge can combine several parents. Notes record the researcher's hypotheses, interpretations, and plans with references to experiments, measurements, code, or other notes, and revisions retain the original text, so forward and reverse queries reveal what produced a result, where it was reused, and which interpretations discuss it (Section 4.3). Execution records, notes, and session transcripts persist outside the agent context, so a researcher can resume a session, start fresh, or be replaced while keeping the same evidence.

Keep delayed experiments productive. Evaluation must wait for its checkpoint, while independent recipe branches proceed concurrently. Acceptance fixes the evaluations each experiment needs; the executor schedules preparation, training, merging, and evaluation as inputs and resources become available, and once a checkpoint is ready its evaluation and upload overlap, so measurements can inform the next decision while uploads and other jobs continue (Figure 3). Accepted requests and execution states persist across sessions; after an interruption, the control plane checks existing work and resumes unfinished execution, measurement, or checkpoint upload.

4 Autonomous Post-Training at Scale

We first state our experimental setting (Section 4.1). Then, we report model improvement, how far the campaigns raise the base model (Section 4.2). Next, we analyze the research trajectory, how the researcher searches and revises its recipes (Section 4.3). Finally, we measure execution effectiveness: how quickly the researcher turns evidence into experiments, how fast the system returns feedback, and whether campaigns continue through failures (Section 4.4).

4.1 Experimental Setting

Setting. Claude Opus 5 (Anthropic, 2026b) acts as the researcher improving Qwen3-4B-Base[1] (Yang et al., 2025). It may select data, branch from checkpoints, and compose SFT, RLVR, on-policy distillation (OPD) from its own checkpoints, and merging over a pool of 264 decontaminated corpora spanning instruction following, function calling, code, and mathematics (Appendix D.3), on up to 28 eight-GPU nodes; a 4B model keeps each experiment short enough for many per campaign.

Evaluation and references. The feedback suite, Core7, comprises AIME 2025, BFCL, GPQA Diamond, GSM8K, HumanEval+, IFEval, and MMLU-Pro; the researcher receives the full score vector after each experiment, and the unweighted average ranks checkpoints. The held-out suite, OOD7, comprises AIME 2026, HMMT February 2026, MATH-500, LiveCodeBench, MBPP+, IFBench, and SuperGPQA. References are the starting model and the official instruction-tuned model under the same frozen evaluator (Appendix D).

Campaign settings. Every campaign has the same goal: the highest Core7 average it can reach within its time window. Each campaign runs fresh researcher sessions with a shared charter and on-demand skills (Appendix D.2). The six successive campaigns (Table 1) differ only in what they start with. The first three start from the base model, and the third also receives the research record of the first two. The later three also receive checkpoints kept from earlier campaigns.

4.2 Model Improvement

Search from the base. C1 raises the running best, the *frontier*, from 26.36 to 59.51 in 23.3 hours, over 146 accepted experiments and 71 Core7 evaluations using 915 GPU hours (Table 1). Its winning lineage stacks supervised fine-tuning, a DARE-TIES merge (Yu et al., 2024a; Yadav et al., 2023) of three fine-tuned checkpoints, and RLVR on the merged checkpoint. Only 11 of the 71 evaluations define a frontier point, and 8 fall more than five points below the frontier of their time (Figure 8). Two further campaigns from the base reach 60.14 and 61.67, the third within 15.1 hours after inheriting the record of the first two.

[1] https://huggingface.co/Qwen/Qwen3-4B-Base

Table 1: **Statistics of the six successive campaigns.** Hours are wall-clock research time. Exp. counts accepted recipes. Improved counts new checkpoints that beat their parent by more than the *noise threshold* of 1.3 points, the researcher's own estimate of the smallest real Core7 difference under the fixed evaluation protocol (Section 4.3). Best is the highest online Core7 score; the last column gives that checkpoint's lineage. Qwen3-4B-Base scores 26.36 and Qwen3-4B-Instruct-2507 70.36.

Campaign	Hours	Exp.	GPU·h	Improved	Evals	Best	Best checkpoint
C1	23.3	146	915	26/60	71	59.51	merge of 3 SFT, then RLVR
C2	21.1	153	1,404	42/92	116	60.14	2 SFT, then 2 RLVR
C3	15.1	136	1,187	46/113	164	61.67	4 SFT, then 3 RLVR
C4	20.8	150	1,002	2/70	130	62.07	C3 lineage, RLVR, merge
C5	14.7	124	569	2/66	112	62.24	re-evaluated C4 checkpoint
C6	29.0	130	737	11/63	118	63.21	2 SFT, then 5 RLVR

Table 2: **Five findings from the record of the six campaigns.**

Finding	What the record shows	Measurement
(1) Local search	One setting changes at a time, around a few trusted corpora	66–87% of SFT and 60–92% of RLVR experiments differ from another in one setting
(2) Recipe convergence	Independent campaigns reach the same core recipe	All six winners: Dolci-Instruct SFT, then Dolci-RL-Zero RLVR at learning rate 10^{-6}
(3) Unexamined data	Losses come from data rather than execution errors	12 of 51 trained corpora never inspected
(4) Lineage awareness	The agent traces a recipe's history: its gain depends on the starting checkpoint	Same RLVR recipe: $+1.77$ and -0.52 on different parents
(5) Self-correction	The agent revises earlier hypotheses; closed directions persist across campaigns	C1 drops its 50-step RLVR rule after a 200-step run wins; all 3 directions retried in C6 lose again

Continued search. The campaigns that inherit the record and retained checkpoints (C4–C6) reach 62.07, 62.24, and 63.21. C5's best score, 62.24, is a repeat evaluation of an inherited C4 checkpoint; its own best new checkpoint scores 62.17. C6 reaches 63.21, 7.2 points below the official instruction-tuned model, with a lineage rebuilt from the base.

Held-out transfer. The best checkpoint improves the base from 15.34 to 32.04 on the sealed OOD7 suite, evaluated once after the last campaign closed, showing transfer beyond the visible tasks; the official model scores 45.27.

4.3 Trajectory Analysis

The record of the six campaigns holds 839 accepted experiments, 711 Core7 evaluations, and nearly a thousand notes written by the researcher. We summarize it as five behaviors, with their measurements in Table 2; the verbatim notes behind the cases are in Appendices E.4, E.6.1, and E.6.2.

(1) Local search. The researcher behaves as a careful experimenter rather than an explorer. Campaigns from the base spend the first quarter of their hours mostly on SFT (36 of 58 and 39 of 63 experiments in C1 and C2), while C4, which inherits checkpoints, opens with RLVR on them (33 of 70); after that the researcher advances by one-variable contrasts around recipes that have worked, holding most settings at the values it trusts. What gets explored is therefore set by the agent's prior.

(2) Recipe convergence. Independent campaigns rediscover the same recipe. C1 and C2 never saw each other's work, yet both winning lineages contain SFT on Dolci-Instruct, an instruction-tuning mixture, followed by RLVR at 10^{-6} on Dolci-RL-Zero, a verifiable-reward prompt set (Team Olmo et al., 2025), and so do all six winners, including C6's, rebuilt from the base. Later stages vary within a few kinds (a second instruction SFT with verifier-checked RLVR, merges as a hedge, or alternating code and mathematics RLVR); what differs is depth and dose, not structure. Once the structure is

found, more campaigns tune it rather than replace it, which is why the frontier of Figure 1 moves only from 59.5 to 63.2 over six campaigns.

(3) Unexamined data. The research fails on data rather than execution: 92 of 839 experiments failed to run, and each failure reached the researcher as a recorded error. The costly failures are runs that completed correctly yet made the model worse, on training data the researcher had not inspected. It trained on 12 of its 51 corpora without ever inspecting their rows; one of them took an SFT checkpoint below the base (trace below). It did inspect the function-calling corpora, yet function-calling SFT left strong checkpoints worse than their parent in 19 of 21 attempts: it read the rows but not what they would do to the model. Most of the pool was never used, 213 of 264 corpora (Appendix D.3). The next gains lie in inspecting data and model behavior before training.

Finding (3), unexamined data: a corpus trained without inspection (C2)

Evidence. A from-base SFT on natural-instructions, identical to the other single-corpus SFT runs (*screens*) except for the corpus, was predicted to land near 45–46. It scored 17.79, the campaign's only checkpoint below the base: the multiple-choice (MC) tasks GPQA and MMLU-Pro fell below chance (0.5 and 1.4) and GSM8K to 1.0, while BFCL stayed at 56.7.

RESEARCHER *A sub-chance MC score is not capability loss, it is answer-emission loss. natural-instructions targets are bare labels and short spans with no reasoning and no restatement, so the model stops emitting a parseable answer . . . bfcl survives because a tool call is itself a short structured fragment.*

Action. The researcher dropped the corpus instead of retuning it and wrote a rule for later screens: a below-chance multiple-choice score is *"a format verdict, never a knowledge verdict."*

When the researcher does inspect, diagnosis guides the recipe. In C2, a per-task reading overturned the researcher's own rule about what raises the Core7 average (Appendix E.5), and the light second-stage SFT it found recurs in the winning lineages of C2 through C5.

(4) Lineage awareness. The researcher learns that a recipe's value belongs to the checkpoint it is applied to. The same RLVR recipe raised one checkpoint from 58.37 to 60.14 and lowered another from 58.36 to 57.84 (Table 14), and an RLVR corpus that had failed on RL-trained checkpoints gained 4.0 points on a fresh SFT checkpoint (trace under (5)). The researcher learned to treat a winning recipe as *"a hypothesis on a new parent, not a result"* (Appendix E.4). A result is interpretable only with its lineage, which is why Athanor records it.

(5) Self-correction. The researcher corrects itself quickly within a campaign, but the record carries its prohibitions further than its discoveries. *Revising hypotheses:* early conclusions fall to the next experiment. C1 fixed RLVR at 50 steps until one longer run reached 59.51 at 200 steps, its best checkpoint, and C2 declared merging closed before two merges became its best checkpoints at the time. *Rereading failures:* when a corpus swap drove C2's multiple-choice scores below chance, the researcher traced the drop to a lost answer format rather than lost knowledge and made that a rule (trace under (3)). *Calibrating noise:* C4 moved its noise threshold, its own estimate of the smallest real Core7 difference, from 1.2 to 0.5 to 1.3 points after the same checkpoint scored 62.03 and 60.63 under two evaluation seeds. *Keeping closures:* three directions closed earlier were retried in C6 and repeated their losses, and merging, closed three times, was submitted 59 more times in C5 and C6; none of the 34 evaluated merges beat a parent by more than the noise threshold. The one reopening that paid did so because C6 read the condition under which a direction had been closed. A record that keeps these conditions, not only verdicts, lets a later session reopen the right door.

Finding (5), self-correction: a closed direction reopens on a new checkpoint (C6)

Evidence. Earlier in C6, the register of closed directions had closed Dolci-RL-Zero and Skywork-OR1 RLVR after both failed on inherited RL-trained checkpoints, whose policy entropy at the first RLVR step was already low. A later run applied Dolci-RL-Zero RLVR to a fresh SFT checkpoint.

RESEARCHER *Why the register was wrong . . . all on converged inits, the inherited seeds whose step-0 entropy reads 0.05–0.45. [This run] uses the same corpus . . . and pays +3.971, because its init opens at entropy 9.24.*

Outcome. The stage entered C6's best lineage (63.21). Hours later the researcher retracted the entropy rule: an init at 9.49 *"paid nothing."*

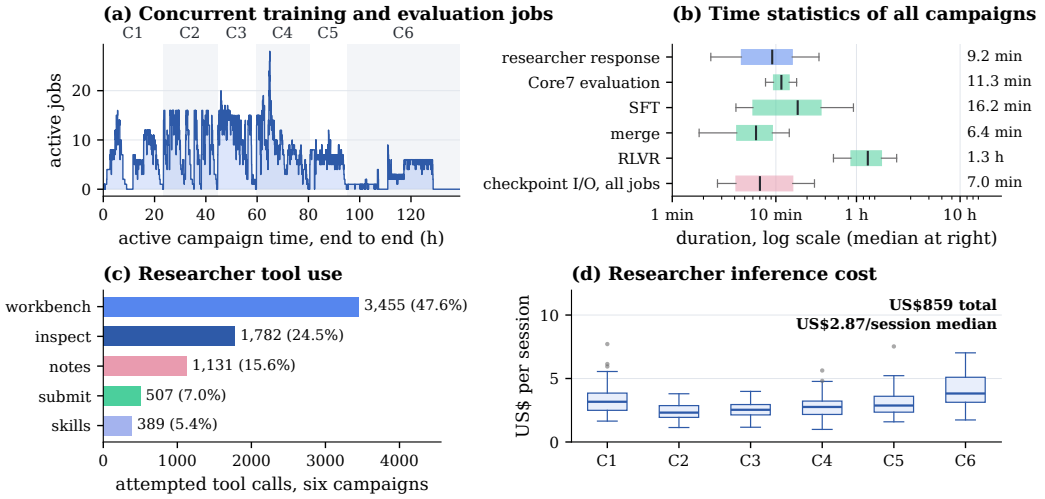


Figure 4: **Agent and system dynamics across the six campaigns.** (a) Concurrent training and evaluation jobs, with the campaigns placed end to end. (b) Time statistics of all campaigns: researcher response, computation of successful jobs by type, and checkpoint I/O pooled over all jobs; boxes span quartiles, whiskers the 10th–90th percentiles. (c) Researcher tool calls by kind. (d) Model cost of each successful session at API list price.

What limits autoresearch scaling. Along the axis of Section 2, returns fall quickly with compute (Figure 1): the first 915 GPU hours, all of C1, raise the Core7 average by 33.2 points, and the remaining 4,900 add 3.7. The findings suggest why. Local search and recipe convergence keep later campaigns tuning one recipe structure, and each campaign after C3 moves the frontier by less than the researcher's noise threshold. The curve is set by what the researcher tries and checks: plain, bland attempts improve little, so autoresearch needs research novelty and diversity.

4.4 Execution Effectiveness

Agent efficiency. A new score was answered by the next accepted experiment after a median of 9.2 minutes (Figure 4b). The 278 successful sessions cost US$859 at list price (median US$2.87). Of 7,264 tool calls, 48% were workbench analysis, 25% inspect calls, 16% notes, and 7% submissions (Figure 4c,d): the researcher's work is mostly reading, analyzing, and reasoning over evidence.

System efficiency. On eight GPUs, a successful Core7 evaluation computed for a median of 11 minutes, an SFT job 16, a merge 6, and an RLVR job 77, and moving checkpoints in and out added a median of 7 minutes per job (Figure 4b); with data preparation, queueing, and evaluation, a score arrived a median of 1.7 hours after submission (τ_{fb}, Section 2). At least one GPU job was active 86% of the time, with a mean of 6.2 concurrent jobs and a peak of 28 (Figure 4a).

Long-horizon reliability. All but one of 1,381 launches ended in a recorded terminal state; excluding 39 stopped on request, 10% failed (4% of SFT, 25% of RLVR, and 6% of evaluation jobs). These failures, and those of 124 of 402 sessions, stayed local: each new session starts from the record.

5 Conclusion

Athanor lets a frontier agent run post-training research end to end and keeps a research lineage of recipes, notes, and results. Across six campaigns, the agent improves Qwen3-4B-Base from 26.4 to 63.2 on seven benchmarks, against 70.4 for the official instruction-tuned model, and from 15.3 to 32.0 on a held-out suite. Our results show that agents are capable of autonomous post-training over long horizons but still lack research novelty and diversity. We hope that Athanor and our findings will provide a foundation for scalable autonomous AI research.

AI USE STATEMENT

Following the ICLR 2027 policy on AI use, we disclose the following uses. *As the object of study:* in the reported campaigns, the researcher agent (Claude Opus 5) proposed and refined hypotheses, designed experiments, implemented data-processing and training programs, and interpreted results; these actions are what the paper analyzes. *In building the system:* coding agents helped write and revise Athanor's software, which the authors designed, reviewed, and tested. *In analysis and writing:* AI assistants helped write the scripts that compute statistics from the campaign records, create and refine figures, search and summarize related literature, check and format references, and draft and edit text for readability. The idea of autoresearch on post-training, the system design, and the design of the campaigns are the authors' own. We have checked the AI-assisted analyses, figures, references, and text against the underlying records and sources, and we take full responsibility for the paper.

REFERENCES

agentlans. train-of-thought. Hugging Face dataset, 2025. URL https://huggingface.co/datasets/agentlans/train-of-thought.

Wasi Uddin Ahmad, Aleksander Ficek, Mehrzad Samadi, Jocelyn Huang, Vahid Noroozi, Somshubra Majumdar, and Boris Ginsburg. OpenCodeInstruct: A large-scale instruction tuning dataset for code LLMs. *arXiv preprint arXiv:2504.04030*, 2025. URL https://arxiv.org/abs/2504.04030.

Anthropic. Claude Fable 5 and Claude Mythos 5. https://www.anthropic.com/news/claude-fable-5-mythos-5, 2026a.

Anthropic. Introducing Claude Opus 5. https://www.anthropic.com/news/claude-opus-5, July 2026b.

Daman Arora, Himanshu Singh, and Mausam. Have LLMs advanced enough? A challenging problem solving benchmark for large language models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7527–7543, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.468. URL https://aclanthology.org/2023.emnlp-main.468/.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc V. Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021. URL https://arxiv.org/abs/2108.07732.

Mislav Balunovic, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. MathArena: Evaluating LLMs on uncontaminated math competitions. In *Advances in Neural Information Processing Systems*, volume 38. Curran Associates, Inc., 2025. doi: 10.52202/085713-0679. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/1d27c01ebd3e3aebe226b44fc970d803-Paper-Datasets_and_Benchmarks_Track.pdf.

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 *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=6s5uXNWGIh.

Guhong Chen, Yingcheng Shi, Yongbin Li, Binhua Li, Xander Xu, Hu Wei, Shiwen Ni, Min Yang, and Jieping Ye. EvoTrainer: Co-evolving LLM policies and training harnesses for autonomous agentic reinforcement learning. *arXiv preprint arXiv:2606.03108*, 2026a. URL https://arxiv.org/abs/2606.03108.

Huanran Chen, Huaqing Zhang, Xiao Li, Yinpeng Dong, Ke Shen, and Jun Zhu. Nexus: Same pretraining loss, better downstream generalization via common minima. *arXiv preprint arXiv:2604.09258*, 2026b.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Wanyi Chen, Xiao Yang, Xu Yang, Tianming Sha, Qizheng Li, Zhuo Wang, Bowen Xian, Fang Kong, Weiqing Liu, and Jiang Bian. Agent^2 RL-Bench: Can LLM agents engineer agentic RL post-training? *arXiv preprint arXiv:2604.10547*, 2026c.

Yang Chen, Zhuolin Yang, Zihan Liu, Chankyu Lee, Peng Xu, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. AceReason-Nemotron: Advancing math and code reasoning through reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. URL https://openreview.net/forum?id=EgArbnS0BA.

Yueh-Han Chen, Jiaxin Wen, and Jan Hendrik Kirchner. Automated researchers can mitigate well-characterized alignment failures. *arXiv preprint arXiv:2608.28945*, 2026d. URL https://arxiv.org/abs/2608.28945.

Zehui Chen, Kuikun Liu, Qiuchen Wang, Wenwei Zhang, Jiangning Liu, Dahua Lin, Kai Chen, and Feng Zhao. Agent-FLAN: Designing data and methods of effective agent tuning for large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 9354–9366, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.557. URL https://aclanthology.org/2024.findings-acl.557/.

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

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 preprint arXiv:1803.05457*, 2018. URL https://arxiv.org/abs/1803.05457.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. URL https://arxiv.org/abs/2110.14168.

DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026. URL https://arxiv.org/abs/2606.19348.

Guanting Dong, Keming Lu, Chengpeng Li, Tingyu Xia, Bowen Yu, Chang Zhou, and Jingren Zhou. Self-play with execution feedback: Improving instruction-following capabilities of large language models. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=cRR0oDFEBC.

Xeron Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, et al. SuperGPQA: Scaling LLM evaluation across 285 graduate disciplines. In *Advances in Neural Information Processing Systems*, volume 38. Curran Associates, Inc., 2025. doi: 10.52202/085713-3766. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/a3c5af1f56fc73eef1ba0f442739f5ca-Paper-Datasets_and_Benchmarks_Track.pdf.

Yaxin Du, Xiyuan Yang, Zhifan Zhou, Wanxu Liu, Zixing Lei, Zimeng Chen, Fenyi Liu, Haotian Wu, Yuzhu Cai, Zexi Liu, et al. DataMaster: Data-centric autonomous AI research. *arXiv preprint arXiv:2605.10906*, 2026.

Haoyang Fang, Wei Zhu, Boran Han, Alex Zhang, Zhenyu Pan, Shuo Yang, Shuai Zhang, Jiading Gai, Peng Tang, Cuixiong Hu, Xuan Zhu, Huzefa Rangwala, George Karypis, and Bernie Wang. LLMZero: Discovering adaptive training strategies for RL post-training via LLM agents. *arXiv preprint arXiv:2606.18388*, 2026. URL https://arxiv.org/abs/2606.18388.

Yanlin Fei, Nazhou Liu, Xinmiao Yu, Shaolong Chen, Lei Li, Rahul Thapa, Madalina Ciobanu, Navan Preet Singh, Qingqing Mao, and Ritankar Das. How do agents fail on AutoResearch: End-to-end diagnostic evaluation on 100 real-world frontier research tasks. *arXiv preprint arXiv:2608.14905*, 2026. URL https://arxiv.org/abs/2608.14905.

Glaive AI. glaive-function-calling-v2. Hugging Face dataset, 2023. URL https://huggingface.co/datasets/glaiveai/glaive-function-calling-v2.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vladimir Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. Arcee's MergeKit: A toolkit for merging large language models. In Franck Dernoncourt, Daniel Preoţiuc-Pietro, and Anastasia Shimorina (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 477–485, Miami, Florida, US, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-industry.36. URL https://aclanthology.org/2024.emnlp-industry.36/.

Irving John Good. Speculations concerning the first ultraintelligent machine. In *Advances in Computers*, volume 6, pp. 31–88. Elsevier, 1966. doi: 10.1016/S0065-2458(08)60418-0. URL https://www.sciencedirect.com/science/article/pii/S0065245808604180.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, 2025. doi: 10.1038/s41586-025-09422-z. URL https://www.nature.com/articles/s41586-025-09422-z.

Karen Hambardzumyan, Nicolas Baldwin, Edan Toledo, Rishi Hazra, Michael Kuchnik, Bassel Al Omari, Thomas Simon Foster, Anton Protopopov, Jean-Christophe Gagnon-Audet, Ishita Mediratta, Kelvin Niu, Michael Shvartsman, Alisia Lupidi, Alexis Audran-Reiss, Parth Pathak, Tatiana Shavrina, Despoina Magka, Hela Momand, Derek Dunfield, Nicola Cancedda, Pontus Stenetorp, Carole-Jean Wu, Jakob Nicolaus Foerster, Yoram Bachrach, and Martin Josifoski. AIRA$_2$: Overcoming bottlenecks in AI research agents. *arXiv preprint arXiv:2603.26499*, 2026. URL https://arxiv.org/abs/2603.26499.

Jujie He, Jiacai Liu, Chris Yuhao Liu, Rui Yan, Chaojie Wang, Peng Cheng, Xiaoyu Zhang, Fuxiang Zhang, Jiacheng Xu, Wei Shen, Siyuan Li, Liang Zeng, Tianwen Wei, Cheng Cheng, Bo An, Yang Liu, and Yahui Zhou. Skywork open reasoner 1 technical report. *arXiv preprint arXiv:2505.22312*, 2025. URL https://arxiv.org/abs/2505.22312.

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. DeepMath-103K: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://arxiv.org/abs/2504.11456.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 30016–30030. Curran Associates, Inc., 2022. doi: 10.52202/068431-2176. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/c1e2faff6f588870935f114ebe04a3e5-Abstract.html.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=t9U3LW7JVX.

Intology. Scaling automated post-training. https://intology.ai/blog/scaling-automated-post-training, 2026.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida I. Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=chfJJYC3iL.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations*, volume 2024, pp. 54107–54157, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/edac78c3e300629acfe6cbe9ca88fb84-Abstract-Conference.html.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? A large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421, 2021. doi: 10.3390/app11146421. URL https://www.mdpi.com/2076-3417/11/14/6421.

Jiajie Jin, Yuyang Hu, Kai Qiu, Qi Dai, Chong Luo, Guanting Dong, Xiaoxi Li, Tong Zhao, Xiaolong Ma, Gongrui Zhang, Zhirong Wu, Bei Liu, Zhengyuan Yang, Linjie Li, Lijuan Wang, Hongjin Qian, Yutao Zhu, and Zhicheng Dou. Toward generalist autonomous research via hypothesis-tree refinement. *arXiv preprint arXiv:2606.11926*, 2026. URL https://arxiv.org/abs/2606.11926.

Liqiang Jing, Zhehui Huang, Xiaoyang Wang, Wenlin Yao, Wenhao Yu, Kaixin Ma, Hongming Zhang, Xinya Du, and Dong Yu. DSBench: How far are data science agents from becoming data science experts? In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=DSsSPr0RZJ.

Keller Jordan, Jeremy Bernstein, Brendan Rappazzo, @fernbear.bsky.social, Boza Vlado, You Jiacheng, Franz Cesista, Braden Koszarsky, and @Grad62304977. modded-nanogpt: Speedrunning the NanoGPT baseline, 2024. URL https://github.com/KellerJordan/modded-nanogpt.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Devvrit Khatri, Lovish Madaan, Rishabh Tiwari, Rachit Bansal, Venkata Sai Surya Subramanyam Duvvuri, Manzil Zaheer, Inderjit S. Dhillon, David Brandfonbrener, and Rishabh Agarwal. The art of scaling reinforcement learning compute for LLMs. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=FMjeC9Msws.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. QASC: A dataset for question answering via sentence composition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8082–8090, 2020. URL https://ojs.aaai.org/index.php/AAAI/article/view/6319.

Yubin Kim, Ken Gu, Chanwoo Park, Chunjong Park, Samuel Schmidgall, A. Ali Heydari, Yao Yan, Zhihan Zhang, Yuchen Zhuang, Yun Liu, Mark Malhotra, Paul Pu Liang, Hae Won Park, Yuzhe Yang, Xuhai Xu, Yilun Du, Shwetak Patel, Tim Althoff, Daniel McDuff, and Xin Liu. Towards a science of scaling agent systems, 2025. URL https://arxiv.org/abs/2512.08296.

Kimi Team. Kimi K3: Open frontier intelligence. *arXiv preprint arXiv:2607.24653*, 2026. URL https://arxiv.org/abs/2607.24653.

Nathan Lambert. *Reinforcement Learning from Human Feedback: LLM Alignment and Post-Training*. Manning Publications, July 2026. ISBN 9781633434301. URL https://www.manning.com/books/reinforcement-learning-from-human-feedback.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James Validad Miranda, Alisa Liu, Nouha Dziri, Xinxi Lyu, et al. Tülu 3: Pushing frontiers in open language model post-training. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=i1uGbfHHpH.

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

Qizheng Li, Yifei Zhang, Xiao Yang, Xu Yang, Zhuo Wang, Bowen Xian, Weiqing Liu, and Jiang Bian. FT-Dojo: Towards autonomous LLM fine-tuning with language agents. In *Forty-third International Conference on Machine Learning*, 2026b. URL https://icml.cc/virtual/2026/poster/63475.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Joy Jia Yin Lim, Xin Huang, Hao Peng, Yaxi Lu, Xin Cong, Zhong Zhang, Maosong Sun, and Yankai Lin. What is missing from AI post-training AI: An empirical analysis. *arXiv preprint arXiv:2608.19072*, 2026. URL https://arxiv.org/abs/2608.19072.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? Rigorous evaluation of large language models for code generation. In *Advances in Neural Information Processing Systems*, volume 36, pp. 21558–21572. Curran Associates, Inc., 2023. doi: 10.52202/075280-0943. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/43e9d647ccd3e4b7b5baab53f0368686-Paper-Conference.pdf.

Chris Lu, Cong Lu, Robert Tjarko Lange, Yutaro Yamada, Shengran Hu, Jakob Foerster, David Ha, and Jeff Clune. Towards end-to-end automation of AI research. *Nature*, 651(8107):914–919, 2026. doi: 10.1038/s41586-026-10265-5. URL https://www.nature.com/articles/s41586-026-10265-5.

Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/11332b6b6cf4485b84afadb1352d3a9a-Abstract-Conference.html.

Xiaotian Luo, Dizhan Xue, Fengxingyu Wang, Chuanrui Hu, and Yafeng Deng. HarnessBank: Semantic gene-bank search with gated verification for agent-harness self-evolution. *arXiv preprint arXiv:2607.13683*, 2026. URL https://arxiv.org/abs/2607.13683.

Alisia Lupidi, Bhavul Gauri, Thomas Simon Foster, Bassel Al Omari, Despoina Magka, Alberto Pepe, Alexis Audran-Reiss, Muna Aghamelu, Nicolas Baldwin, Lucia Cipolina-Kun, Jean-Christophe Gagnon-Audet, Chee Hau Leow, Sandra Lefdal, Hossam Mossalam, Abhinav Moudgil, Saba Nazir, Emanuel Tewolde, Isabel Urrego, Jordi Armengol Estape, Amar Budhiraja, Gaurav Chaurasia, Abhishek Charnalia, Derek Dunfield, Karen Hambardzumyan, Daniel Izcovich, Martin Josifoski, Ishita Mediratta, Kelvin Niu, Parth Pathak, Michael Shvartsman, Edan Toledo, Anton Protopopov, Roberta Raileanu, Alexander Miller, Tatiana Shavrina, Jakob Foerster, and Yoram Bachrach. AIRS-Bench: a suite of tasks for frontier AI research science agents. *arXiv preprint arXiv:2602.06855*, 2026. URL https://arxiv.org/abs/2602.06855.

Bohan Lyu, Yucheng Yang, Siqiao Huang, Jiaru Zhang, Qixin Xu, Xinghan Li, Xinyang Han, Yicheng Zhang, Huaqing Zhang, Runhan Huang, et al. MLS-Bench: A holistic and Rigorous assessment of AI systems on building better AI. *arXiv preprint arXiv:2605.08678*, 2026. URL https://arxiv.org/abs/2605.08678.

Lovish Madaan, Aaditya K. Singh, Rylan Schaeffer, Andrew Poulton, Sanmi Koyejo, Pontus Stenetorp, Sharan Narang, and Dieuwke Hupkes. Quantifying variance in evaluation benchmarks. *arXiv preprint arXiv:2406.10229*, 2024.

Fanqing Meng, Lingxiao Du, Qiguang Chen, Ziqi Zhao, Haocheng Lu, Mengkang Hu, and Michael Qizhe Shieh. RSIBench-Data: Benchmarking data-centric research for recursive self-improvement. *arXiv preprint arXiv:2607.25886*, 2026.

Mike Merrill, Alexander Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Shin, Thomas Walshe, E Kelly Buchanan, et al. Terminal-Bench: Benchmarking agents on hard, realistic tasks in command line interfaces. In *International Conference on Learning Representations*, volume 2026, pp. 40903–40986, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/444a3737adaee10d86ad2ef5f74468e6-Abstract-Conference.html.

Dhruv Nathawani, Shuoyang Ding, Vitaly Lavrukhin, Igor Gitman, Somshubra Majumdar, Evelina Bakhturina, Boris Ginsburg, and Jane Polak Scowcroft. Nemotron-Post-Training-Dataset-v2, August 2025. URL https://huggingface.co/datasets/nvidia/Nemotron-Post-Training-Dataset-v2.

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

NVIDIA. NVIDIA Nemotron Nano 2: An accurate and efficient hybrid Mamba-Transformer reasoning model, 2025. URL https://arxiv.org/abs/2508.14444.

OpenAI. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. URL https://arxiv.org/abs/2303.08774.

OpenAI. Learning to reason with LLMs. https://openai.com/index/learning-to-reason-with-llms/, 2024.

OpenAI. GPT-5.6: Frontier intelligence that scales with your ambition. https://openai.com/index/gpt-5-6/, 2026a.

OpenAI. Research acceleration: The view inside OpenAI. https://openai.com/index/research-acceleration-view-inside-openai/, September 6 2026b.

Shishir G. Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The Berkeley function calling leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 48371–48392. PMLR, 2025. URL https://proceedings.mlr.press/v267/patil25a.html.

Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hanna Hajishirzi. Generalizing verifiable instruction following. In *Advances in Neural Information Processing Systems*, volume 38. Curran Associates, Inc., 2025. doi: 10.52202/085713-1645. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/46499a0622ecf568b72d17b61e45dbd5-Paper-Datasets_and_Benchmarks_Track.pdf.

Ao Qu, Han Zheng, Zijian Zhou, Yihao Yan, Yihong Tang, Shao Yong Ong, Fenglu Hong, Kaichen Zhou, Chonghe Jiang, Minwei Kong, Jiacheng Zhu, Xuan Jiang, Sirui Li, Cathy Wu, Bryan Kian Hsiang Low, Jinhua Zhao, and Paul Pu Liang. CORAL: Towards autonomous multi-agent evolution for open-ended discovery. *arXiv preprint arXiv:2604.01658*, 2026. URL https://arxiv.org/abs/2604.01658.

Qwen Team. Qwen3.5: Towards native multimodal agents. https://qwen.ai/blog?id=qwen3.5, 2026.

Ben Rank, Hardik Bhatnagar, Ameya Prabhu, Shira Eisenberg, Karina Nguyen, Matthias Bethge, and Maksym Andriushchenko. PostTrainBench: Can LLM agents automate LLM post-training? In *Forty-third International Conference on Machine Learning*, 2026. URL https://icml.cc/virtual/2026/poster/63667.

Redwood Research. mbpp_extended. Hugging Face dataset, 2023. URL https://huggingface.co/datasets/redwoodresearch/mbpp_extended.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level Google-Proof Q&A benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, 4 edition, 2021.

Jürgen Schmidhuber. Gödel machines: Fully self-referential optimal universal self-improvers. In *Artificial general intelligence*, pp. 199–226. Springer, 2007. doi: 10.1007/978-3-540-68677-4_7. URL https://link.springer.com/chapter/10.1007/978-3-540-68677-4_7.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/432aca3a1e345e339f35a30c8f65edce-Paper.pdf.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. HybridFlow: A flexible and efficient RLHF framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297. Association for Computing Machinery, 2025. doi: 10.1145/3689031.3696075. URL https://doi.org/10.1145/3689031.3696075.

Jiajun Shi, Siyuan Tao, Yuhao Wu, Zexuan Wang, Jingyuan Zhang, Jiaheng Liu, Xinping Lei, Xinrong Zhang, Siyuan Fang, Zhewen Tan, Tianle Cai, Junhao Fang, Jiameng Huang, Yueyang Wang, Jinkai Liu, Yuxuan Zhang, Jian Yang, Zhoujun Li, Shen Yan, Wenhao Huang, and Ge Zhang. S3Gym: Can LLMs turn self-testing and self-judging into self-improvement?, 2026a. URL https://arxiv.org/abs/2608.31100.

Zhan Shi, Bing He, Yisi Sang, Benoit Dumoulin, and Hanqing Lu. A-Evolve-Training: Autonomous post-training of a 30B model. *arXiv preprint arXiv:2606.20657*, 2026b.

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

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421/.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, Jacob Morrison, Jake Poznanski, Kyle Lo, et al. Olmo 3. *arXiv preprint arXiv:2512.13961*, 2025.

Wenyi Wang, Piotr Piekos, Nanbo Li, Firas Laakom, Yimeng Chen, Mateusz Ostaszewski, Mingchen Zhuge, and Jürgen Schmidhuber. Huxley-Gödel machine: Human-level coding agent development by an approximation of the optimal self-improving machine. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=T0EiEuhOOL.

Xiaohua Wang, Muzhao Tian, Yuqi Zeng, Zisu Huang, Jiakang Yuan, Bowen Chen, Jingwen Xu, Mingbo Zhou, Wenhao Liu, Muling Wu, et al. Reward hacking in the era of large models: Mechanisms, emergent misalignment, challenges. *arXiv preprint arXiv:2604.13602*, 2026b.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems*, volume 37, pp. 95266–95290. Curran Associates, Inc., 2024. doi: 10.52202/079017-3018. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/ad236edc564f3e3156e1b2feafb99a24-Paper-Datasets_and_Benchmarks_Track.pdf.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. In Leon Derczynski, Wei Xu, Alan Ritter, and Tim Baldwin (eds.), *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pp. 94–106, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4413. URL https://aclanthology.org/W17-4413/.

Lilian Weng. Harness engineering for self-improvement. https://lilianweng.github.io/posts/2026-07-04-harness/, July 2026. Blog post.

Hjalmar Wijk, Tao Roa Lin, Joel Becker, Sami Jawhar, Neev Parikh, Thomas Broadley, Lawrence Chan, Michael Chen, Joshua M. Clymer, Jai Dhyani, Elena Ericheva, Katharyn Garcia, Brian Goodrich, Nikola Jurkovic, Megan Kinniment, Aron Lajko, Seraphina Nix, Lucas Jun Koba Sato, William Saunders, Maksym Taran, Ben West, and Elizabeth Barnes. RE-bench: Evaluating frontier AI R&D capabilities of language model agents against human experts. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 66772–66832. PMLR, 2025. URL https://proceedings.mlr.press/v267/wijk25a.html.

Yuhao Wu, Jingyuan Zhang, Jiajun Shi, Xinping Lei, Qingshui Gu, Yuxuan Zhang, Zexuan Wang, Chen He, Chen Huang, Maojia Song, Zhiyuan Zeng, Shaowen Wang, Jinkai Liu, Yunfeng Shi, Jiaheng Liu, Shen Yan, Wenhao Huang, Ge Zhang, and Wenxuan Zhang. HarnessDev: Can LLMs create and evolve their own agent harness?, 2026a. URL https://arxiv.org/abs/2609.01437.

Yuhao Wu, Jingyuan Zhang, Jiajun Shi, Yuxuan Zhang, Xinping Lei, Junting Zhou, Zexuan Wang, Yuchen Wu, Huan Zhou, Duo Wang, Yinzhu Piao, Yongchang Peng, Yunfeng Shi, Jin Chen, Zuo Wang, Jinkai Liu, Jiaheng Liu, Wenxuan Zhang, Shen Yan, Wenhao Huang, and Ge Zhang. Aspire: Can models self-evolve from vague goals?, 2026b. URL https://arxiv.org/abs/2608.31111.

Cheng Xu, Shuhao Guan, Derek Greene, and M-Tahar Kechadi. Benchmark data contamination of large language models: A survey. *arXiv preprint arXiv:2406.04244*, 2024.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. Magpie: Alignment data synthesis from scratch by prompting aligned LLMs with nothing. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=YkBRJ91viF.

Zhangchen Xu, Junda Chen, Yue Huang, Dongfu Jiang, Jiefeng Chen, Hang Hua, Zijian Wu, Zheyuan Liu, Zexue He, Lichi Li, Shizhe Diao, Jiaxin Pei, Jinsung Yoon, Hao Zhang, Mengdi Wang, Radha Poovendran, Misha Sra, Alex Pentland, and Zichen Chen. AutoLab: Can frontier models solve long-horizon auto research and engineering tasks?, 2026. URL https://arxiv.org/abs/2606.05080.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-merging: Resolving interference when merging models. In *Advances in Neural Information Processing Systems*, 2023.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

John Yang, Carlos Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *Advances in Neural Information Processing Systems*, volume 37, pp. 50528–50652. Curran Associates, Inc., 2024. doi: 10.52202/079017-1601. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/5a7c947568c1b1328ccc5230172e1e7c-Abstract-Conference.html.

Zhuolin Yang, Zihan Liu, Yang Chen, Wenliang Dai, Boxin Wang, Sheng-Chieh Lin, Chankyu Lee, Yangyi Chen, Dongfu Jiang, Jiafan He, et al. Nemotron-Cascade 2: Post-training LLMs with cascade RL and multi-domain on-policy distillation. *arXiv preprint arXiv:2603.19220*, 2026.

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

Xunjian Yin, Xinyi Wang, Liangming Pan, Li Lin, Xiaojun Wan, and William Yang Wang. Gödel agent: A self-referential agent framework for recursively self-improvement. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27890–27913, Vienna, Austria, July 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.1354. URL https://aclanthology.org/2025.acl-long.1354/.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super Mario: Absorbing abilities from homologous models as a free lunch. In *Proceedings of the 41st International Conference on Machine Learning*, 2024a.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. MetaMath: Bootstrap your own mathematical questions for large language models. In *International Conference on Learning Representations (ICLR)*, 2024b. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/c400474e8a36d0812fdee52739288b12-Abstract-Conference.html.

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.

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

Charlie Zhang, Graham Neubig, and Xiang Yue. On the interplay of pre-training, mid-training, and RL on reasoning language models. In *Forty-third International Conference on Machine Learning*, 2026a. URL https://openreview.net/forum?id=TBaUfO9znF.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Tjarko Lange, and Jeff Clune. Darwin Gödel machine: Open-ended evolution of self-improving agents. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=pUpzQZTvGY.

Jianguo Zhang, Tian Lan, Ming Zhu, Zuxin Liu, Thai Hoang, Shirley Kokane, Weiran Yao, Juntao Tan, Zhiwei Liu, Yihao Feng, Juan Carlos Niebles, Shelby Heinecke, Huan Wang, Silvio Savarese, and Caiming Xiong. xLAM: A family of large action models to empower AI agent systems. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 11583–11597, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.naacl-long.578. URL https://aclanthology.org/2025.naacl-long.578/.

Yifan Zhang, Yunheng Zou, Shaokun Zhang, Jian Hu, Hao Zhang, Binfeng Xu, Jan Kautz, and Yi Dong. Agora: Git as shared memory for collective AutoResearch, 2026c. URL https://arxiv.org/abs/2609.18094.

Bingchen Zhao, Despoina Magka, Minqi Jiang, Xian Li, Roberta Raileanu, Tatiana Shavrina, Jean-Christophe Gagnon-Audet, Kelvin Niu, Shagun Sodhani, Michael Shvartsman, Andrei Lupu, Alisia Lupidi, Karen Hambardzumyan, Martin Josifoski, Edan Toledo, Thomas Foster, Lucia Cipolina Kun, Derek Dunfield, Abhishek Charnalia, Alexander Miller, Oisin Mac Aodha, Jakob Foerster, and Yoram Bachrach. The automated LLM speedrunning benchmark: Reproducing NanoGPT improvements. In *Advances in Neural Information Processing Systems*, volume 38. Curran Associates, Inc., 2025. doi: 10.52202/085713-0928. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/27e7f21d16fb840f3720ac87ad896220-Abstract-Datasets_and_Benchmarks_Track.html.

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.

CONTENTS

1 Introduction **2**

2 On Autonomous Post-Training Research **3**
2.1 Problem Formulation . . . 3
2.2 What Makes Autonomous Post-Training Hard . . . 4

3 The Athanor Autoresearch Loop **4**
3.1 The Research Loop . . . 4
3.2 Post-Training Research Environment and Campaign Goal . . . 5
3.3 Keeping the Loop Fast and Reliable . . . 5

4 Autonomous Post-Training at Scale **6**
4.1 Experimental Setting . . . 6
4.2 Model Improvement . . . 6
4.3 Trajectory Analysis . . . 7
4.4 Execution Effectiveness . . . 9

5 Conclusion **9**

A Insights and Discussion **21**

B Related Work **22**

C System Details **25**
C.1 Experiment Execution and Protected Evaluation . . . 25
C.2 Workspaces and Persistent State . . . 25
C.3 Research Record and Researcher Sessions . . . 26
C.4 Researcher Policy . . . 26

D Experimental Details **27**
D.1 Campaign Contract . . . 27
D.2 Researcher Configuration . . . 27
D.3 Data Pool . . . 28
D.4 Visible and Held-Out Suites . . . 30
D.5 Checkpoint Selection and Measurement . . . 30

E Additional Results and Research Traces **31**
E.1 What Was Evaluated . . . 31
E.2 Campaign Sequence and Held-Out Transfer . . . 31
E.3 Search Dynamics of the First Campaign . . . 31
E.4 Continued Research and Recipe Interactions . . . 32
E.5 Evidence for the Diagnosis Case . . . 33
E.6 Research Trajectories . . . 33

Appendix

A Insights and Discussion

Information shapes the search. The researcher forms hypotheses from what the research record exposes, so what it contains changes the search. The full Core7 vector, not the average, showed C2's researcher that a light second-stage SFT had raised the two tasks where its leader trailed most, and it withdrew its rule about what raises the Core7 average (Section 4.3 and Appendix E.5). Repeated readings changed the decision rule itself, raising the researcher's noise threshold to about 1.3 points. The inherited record shapes later campaigns. C3, which inherited the records of C1 and C2 but no checkpoints, rebuilt a lineage from the base to 61.67 in 15.1 hours, against 23.3 hours for C1 to reach 59.51; C6 reopened an RLVR corpus because the record stated it had been ruled out only on RL-trained checkpoints, and gained 4.0 points on a fresh one, while the three other closures it retried repeated their losses. What the record keeps, full vectors, repeats, lineage, and the conditions under which a direction was closed, is therefore part of the research design.

Post-training science. Post-training itself remains an open scientific problem: data composition and dose interact with initialization, objective, stage order, and checkpoint composition, and the space is too large to map by hand. The campaigns surface these interactions as measured lineages. The same RLVR recipe raised one checkpoint from 58.37 to 60.14 and lowered a nearly identical one from 58.36 to 57.84 (Table 14). Function-calling SFT on strong checkpoints left the model worse than its parent in 19 of 21 experiments, a forgetting effect; RLVR tied its parent in 140 of 180 measured checkpoints, yet Dolci-RL-Zero RLVR, closed after failing on RL-trained checkpoints, gained 4.0 points on a fresh SFT checkpoint, a plasticity effect; and one corpus swap took an SFT checkpoint below the base by changing how answers were emitted, a data-generalization effect. Because every such result is tied to an executable recipe and its parent, questions about forgetting, plasticity, and data generalization become testable, and autonomous search can accelerate their study.

Scope and transfer. The 4B target is an experimental substrate for studying how an agent does post-training research: it permits many complete training and evaluation cycles within one campaign, exposing how the researcher responds to delayed, noisy, and conflicting evidence. The recipes it finds may not transfer to larger models or other families. Athanor's separation of research from execution holds at any model scale; applying it to frontier training changes the infrastructure, budget, and feedback horizon, and is future work.

Toward a broader research loop. The reported campaigns fix a filtered data pool and exclude Internet access and external teachers. These boundaries constrain data sources and evaluation targets. Combining Athanor with data flywheels that acquire or generate new training examples, along with automated benchmark construction and evaluation, would broaden the data sources and capabilities under study. That broader setting would also increase contamination and post hoc evaluator optimization risks, requiring corresponding integrity controls. We view this composition as a path toward longer-horizon self-improving AI research.

B Related Work

Automated post-training. PostTrainBench (Rank et al., 2026) is the closest benchmark: an agent receives a small base model, one target benchmark, one GPU, and ten hours, and must produce a specialist checkpoint. Locus (Intology, 2026) scales this setting to parallel, multi-day search that keeps improving after CLI-only agents plateau; on a larger budget, its per-benchmark specialists together exceed the official Qwen3-1.7B instruct release. Other work gives the agent structured training workflows (Yu et al., 2026), narrows the loop to one part of the pipeline (Li et al., 2026b; Chen et al., 2026c; Meng et al., 2026; Du et al., 2026; Fang et al., 2026; Chen et al., 2026a), or runs multi-round campaigns on a larger model (Shi et al., 2026b) or on safety objectives, as automated alignment researchers (AAR) do (Chen et al., 2026d). Trajectory analyses of these settings report non-monotonic search and limited strategy revision (Lim et al., 2026; Meng et al., 2026); Section 4.3 asks the same questions here.

Autonomous research. Since GPT-4 established broad task competence (OpenAI, 2023), inference-time methods have turned a model call into an iterative agent through deliberation, tool use, and language feedback across attempts (Wei et al., 2022; Yao et al., 2023; Shinn et al., 2023; Yuksekgonul et al., 2025), and coding agents made the loop executable by inspecting an environment, modifying artifacts, running them, and revising from the result (Yang et al., 2024; Merrill et al., 2026). Autonomous research extends it to scientific programs and experiments (Lu et al., 2026; Novikov et al., 2025); recent systems add parallel execution, isolated evaluation, and persistent research evidence (Hambardzumyan et al., 2026; Qu et al., 2026; Jin et al., 2026; Li et al., 2026a), or share results between many workers through an append-only Git history (Zhang et al., 2026c). Another line evolves the agent harness itself (Hu et al., 2025; Yin et al., 2025; Zhang et al., 2026b; Luo et al., 2026; Wang et al., 2026a; Weng, 2026), and controlled studies ask when multi-agent coordination helps (Kim et al., 2025). Benchmarks score ML engineering and AI R&D (Chan et al., 2025; Jing et al., 2025; Wijk et al., 2025; Zhao et al., 2025; Lyu et al., 2026), research-process tasks (Lupidi et al., 2026; Chi et al., 2026; Fei et al., 2026; Xu et al., 2026), and agent self-development (Wu et al., 2026b; Shi et al., 2026a; Wu et al., 2026a). Athanor sits with the research systems: its evolved artifact is a post-training recipe and the checkpoint lineage it produces, its evaluator is protected and multi-benchmark, and its record is what Section 4.3 analyzes.

System comparisons. Table 3 places Athanor between two families. Post-training systems (Locus, AutoTrainess, A-Evolve-Training, AAR) optimize a model, but the agent can use the Internet, the target is one benchmark specialist or a narrower objective than a generalist model, and checkpoint parentage is recorded only in part. Research systems (AIRA_2, CORAL, Arbor, Praxist) run parallel search and keep research state, some with full hypothesis and evidence lineage, but what they improve is a program or a task-specific solution rather than model weights, and none records checkpoint lineage. Athanor is the only row with full support in every feature column: it optimizes one generalist checkpoint, records both the checkpoint lineage and the research lineage behind it, runs experiments asynchronously, and seals the final evaluation, while it withholds the Internet so that gains come from the fixed pool and the agent's own analysis. Tables 4 and 5 make the same split by artifact: post-training work differs in which part of the pipeline the agent controls, broader autoresearch in whether the artifact is a solution, a method, or the agent itself, and Athanor's artifact is a recipe with the checkpoint lineage it produces.

Table 3: **Research and execution features of neighboring systems.** Internet access concerns the researcher's ability to retrieve external information or data during a campaign.

System	Internet access	Model as artifact	Generalist checkpoint	Checkpoint lineage	Async execution	Sealed final eval.	Research lineage
Locus (Intology, 2026)	Yes	Y	–	P	Y	P	P
AutoTrainess (Yu et al., 2026)	Yes	Y	–	P	–	P	P
A-Evolve-Training (Shi et al., 2026b)	Yes*	Y	P	P	P	P	P
AAR (Chen et al., 2026d)	Yes	Y	P	P	P	Y	Y
$AIRA_2$ (Hambardzumyan et al., 2026)	–	–	–	–	Y	Y	P
CORAL (Qu et al., 2026)	Optional	–	–	–	Y	P	P
Arbor (Jin et al., 2026)	–	P	–	–	P	P	Y
Praxist (Li et al., 2026a)	–	P	–	–	P	P	Y
Athanor	**No**	Y	Y	Y	Y	Y	Y

Y explicit support; P partial or task-dependent support; – not reported. A generalist checkpoint targets multiple capabilities in one model. Checkpoint lineage records parent–child relationships across training stages; research lineage links hypotheses, artifacts, and evidence. Sealed final evaluation is separate from search feedback; partial support includes evaluator isolation or held-out checks used during selection. Automated alignment researchers (AAR) jointly improve safety dimensions with capability-preservation checks; parallelism is across researcher loops. *A-Evolve-Training reports public-leaderboard retrieval; broader access is unspecified. CORAL disables Internet in its main comparisons and reports web-enabled variants separately.

Table 4: **Automated post-training settings.** Agent control ranges from specialized training interventions to sustained recipe search.

Work or family	Evolved or graded artifact	Search and evaluation boundary	Persistent evidence
PostTrainBench (Rank et al., 2026)	Data, code, and recipe for one specialist model	Internet access; one benchmark, one GPU, ten hours	Submitted workspace and artifacts
Locus (Intology, 2026)	Specialist post-training recipe	Internet access; parallel, multi-day PostTrainBench search	Cross-worker experiment history
AutoTrainess (Yu et al., 2026)	PostTrainBench specialist recipe	Workflow interfaces for planning, data, training, and evaluation	Explicit plans and experiment logs
Specialized training benchmarks (Li et al., 2026b; Chen et al., 2026c; Meng et al., 2026)	Domain fine-tuning, RL pipelines, or data interventions	Task-specific training and feedback under controlled budgets	Submitted models, code, data, and trajectories
Method-specific search systems (Du et al., 2026; Fang et al., 2026; Chen et al., 2026a)	Training data, RL strategies, or policy–harness pairs	Search focused on a selected part of the post-training pipeline	Data or strategy trees, memory, and evolution histories
A-Evolve-Training (Shi et al., 2026b)	One 30B model (also run at 120B and 550B) and its data/recipe	Four autonomous rounds with development and challenge scores	Auditable multi-round trajectory
AAR (Chen et al., 2026d)	Post-training method combinations, data, and model weights	Internet access; multi-day parallel safety search with capability checks and separate generalization tests	Pre-result method reports, code snapshots, shared findings, and scores
Athanor	**One generalist checkpoint and its recipe lineage**	**Multi-benchmark feedback; sealed final tests; asynchronous workers**	**Hypotheses, runs, measurements, and checkpoint lineage**

Table 5: **Autoresearch by research artifact.**

Research family	Optimized or scored artifact	Search or evaluation organization	Retained or scored evidence
ML engineering and AI R&D benchmarks (Chan et al., 2025; Jing et al., 2025; Wijk et al., 2025; Zhao et al., 2025; Lyu et al., 2026)	Competition solutions, training programs, or generalizable methods	Task-specific evaluation of engineering, efficiency, or method performance	Submitted artifacts and evaluator outcomes
Research-task benchmarks (Lupidi et al., 2026; Chi et al., 2026; Fei et al., 2026; Xu et al., 2026)	Scientific research solutions (AIRS), AI-improving algorithms (AI4AI), or complete research attempts (AutoResearch Eval, AutoLab)	Task-native or process-level evaluation	Outcomes, trajectories, and stage-level failures
Self-development benchmarks (Wu et al., 2026b; Shi et al., 2026a; Wu et al., 2026a)	Learning target, experience pathway, model, or agent harness	Hidden or held-out tests of whether changes persist and transfer	Evolved artifacts, trajectories, and evaluator outcomes
Scientific discovery systems (Lu et al., 2026; Novikov et al., 2025)	Scientific programs, experiments, or papers	Iterative experiments or program evolution with task-specific evaluation	Programs, experimental results, or manuscripts
Asynchronous research systems (Hambardzumyan et al., 2026; Qu et al., 2026)	Candidate programs or ML solutions	Parallel exploration with isolated execution and evaluator feedback	Candidate evaluations and persistent research state
Hypothesis and evidence systems (Jin et al., 2026; Li et al., 2026a)	Task-specific research artifacts	Hypothesis refinement or generational search over inherited evidence	Hypotheses, findings, insights, and artifact lineage
Agent-harness self-evolution systems (Hu et al., 2025; Yin et al., 2025; Zhang et al., 2026b; Luo et al., 2026; Wang et al., 2026a)	Agent framework, components, or implementation	Recursive or population-based search on agent tasks	Accepted variants, archives, or ancestry
Athanor	**Model recipe and checkpoint lineage**	**Asynchronous search with protected multi-benchmark evaluation**	**Recipes, artifacts, measurements, failures, and hypotheses**

C System Details

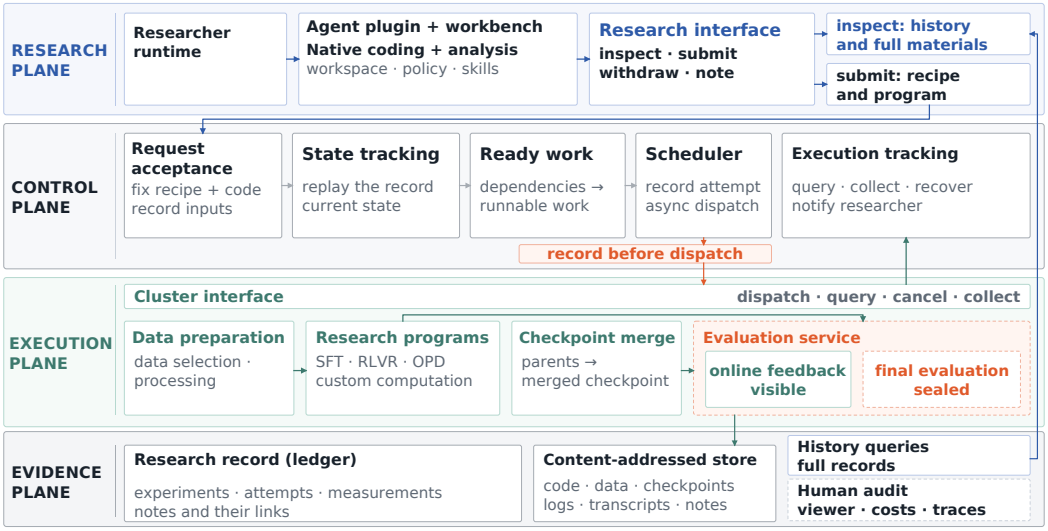


Figure 5: **Runtime architecture.** The researcher reads the history and submits experiments through the research interface. The control plane fixes each accepted request, waits for its dependencies and resources, launches it on the cluster, and tracks it to completion. The research record links attempts, checkpoints, scores, and notes; the researcher's queries and the human audit view both read it. Researcher sessions and training jobs start and end independently.

C.1 Experiment Execution and Protected Evaluation

Execution. An experiment states its intent, inputs, recipe, resources, and expected outputs. On acceptance, the system fills in the training configuration, copies the recipe's program files, records the request, and attaches the evaluations its outputs will receive. Data preparation and training then run in the background. Each launch is recorded before the job starts, so a job that outlives a crash can still be found and finished. Training reuses `verl` (Sheng et al., 2025) for SFT, RLVR, and on-policy distillation (OPD), and `mergekit` (Goddard et al., 2024) for merging checkpoints. Programs the researcher writes itself follow the same rules for inputs, resources, and records, and can produce data, scores, or analyses as well as checkpoints.

Protected evaluation. The evaluator is frozen, and the researcher has no permission to change it. Evaluation may run on the node that trained the checkpoint; the researcher receives the scores and the per-problem outputs it is allowed to see, and the evaluator's credentials stay outside its workspace. The campaign fixes which suites are visible and which stay sealed. Training, checkpoint upload, and evaluation are recorded separately, so a failure in one step keeps the work of the others.

C.2 Workspaces and Persistent State

The research workspace holds the researcher's analysis, recipe programs, and scratch files; each worker holds copies of its inputs, intermediate training state, and outputs waiting for upload (Figure 6). The research record keeps accepted requests, attempts, scores, links to stored files, and notes. Shared storage keeps data, checkpoints, code snapshots, and evaluation outputs.

Submission fixes the recipe and saves a copy of its programs, so later edits in the workspace do not change an accepted experiment. Inspection returns recorded results and the files the researcher may read. Uploaded files outlive both researcher sessions and workers; worker copies are deleted once no job needs them. The researcher and the workers share no filesystem.

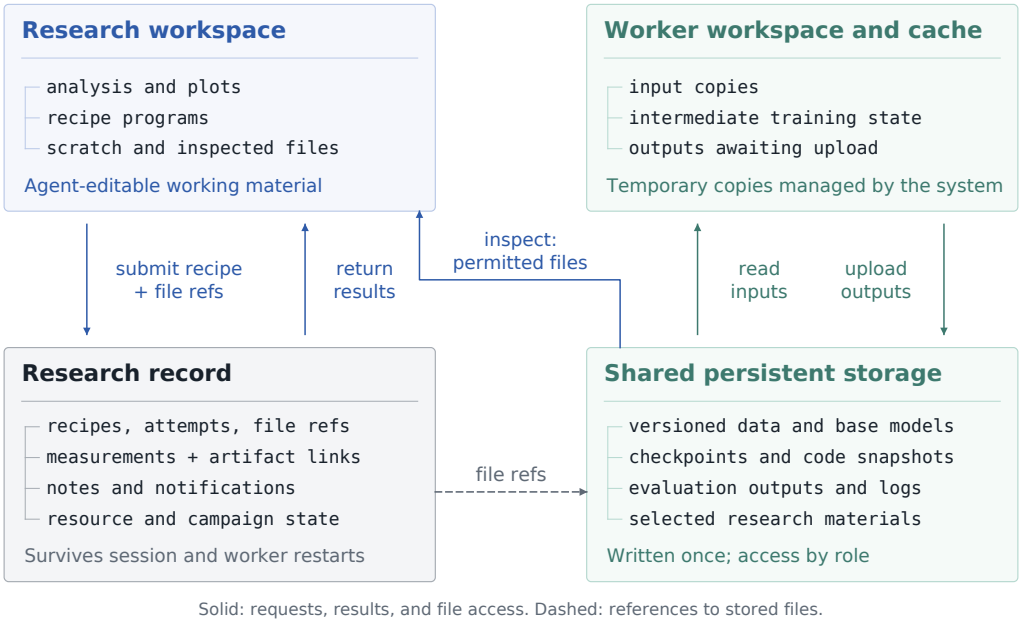


Figure 6: **Workspaces and persistent research state.** The researcher and the workers each have their own working files; the research record links accepted requests and scores to files in shared storage. The researcher reads permitted files through inspection, while workers read inputs and upload outputs. The four regions show who owns each file and how long it lives.

C.3 Research Record and Researcher Sessions

The research record stores accepted experiments, job events, scores, and links to stored files. Notes keep their author, text, the records they cite, and their revisions; session transcripts keep the agent's messages and tool calls. A query returns a record with the records linked to it in either direction.

A new researcher session starts from the campaign goal and pointers into the history; a continuing session receives only what is new. In the reported campaigns every session started fresh. Notifications report new results, idle nodes, or messages from a human, and wait until a session can receive them. Jobs and evaluations keep running whether or not a session is active. After a restart, the system rebuilds its state from the record and checks each job on the cluster before resuming it.

C.4 Researcher Policy

Research guidance is kept separate from the execution rules. A short charter, recipe examples, and skills loaded on demand give the researcher its starting knowledge, while the agent runtime provides its own coding, analysis, and context-management tools. The record notes which guidance each session used. The charter excerpts and skills used in the reported campaigns appear in Appendix D.2.

D EXPERIMENTAL DETAILS

D.1 CAMPAIGN CONTRACT

The paper reports six successive campaigns under one fixed setup (Table 1): the first three start from the base model, the third already inheriting the research record of the first two; the later three also inherit retained checkpoints. Table 6 lists what the researcher may change and what stays fixed.

Table 6: **Campaign setup.** The researcher searches within these boundaries; Athanor records the recipe behind every checkpoint.

Item	Setting
Target and researcher	Qwen3-4B-Base; Claude Opus 5
Search actions	Data mixtures, hyperparameters, checkpoint branches, and compositions of SFT, RLVR, OPD, and model merging
Fixed	One filtered data pool, the Core7 and OOD7 suites, the training code, and up to 28 eight-GPU nodes
Not available	Internet, external teacher models, OOD7 problems and scores, grader code, evaluator credentials
Recorded for selection	Per-benchmark scores, Core7 average, lineage, repeated evaluations

D.2 RESEARCHER CONFIGURATION

Every researcher session starts fresh and receives the same charter. Two other sources give the researcher prior knowledge (Table 7): default training settings, and skills written by us that the researcher can load by name but not edit.

Table 7: **Researcher priors in the reported campaigns.**

Level	Binding	Intended content
Backend defaults	Applied unless the recipe overrides them	Default hyperparameters and settings for each training method
Charter	Read at every session	Objective, rules, boundaries, and which tool to use for what
Skills	Loaded on demand	Research heuristics, diagnostics, and search strategies

The following clauses are quoted from the charter. In it, a *wake* is a researcher session, a *node* is an accepted experiment, a *typed spec* is the structured recipe passed to `submit`, *insights* and *notes* are entries the researcher writes into the record, and the *research status document* is a short summary it keeps up to date.

Selected clauses from the model-visible researcher charter

Goal. *Post-train this campaign's base model as well as it can be done, and leave every result attributable to what produced it. The feedback scores of the frozen suite are the whole of how "as well as it can be done" is measured here.*

Continuity. *Every wake is a fresh process. You remember nothing from the previous one. The ledger is the continuity, and the only things that reach your successor are what entered it: nodes and their results, your insights, your notes, and the research status document.*

Wake protocol. *One wake runs in this order: orient; read the evidence, not the summary; decide, and act through `submit`; record. An experiment runs only if you sent it as a typed spec. Prose is not an action.*

Boundaries. *Scheduling and stopping are the system's. The grader is frozen and outside every surface you can reach. If something in this system looks wrong to you, mark it and leave it standing. Do not repair it in your scratch directory or compensate for it in the next spec.*

The charter sets the goal, the order of work in a session, and the boundaries. Advice on research method comes from the eight skills in Table 8.

Table 8: **Research skills.** The researcher loads a skill when it faces the decision the skill covers.

Skill	Purpose
`hypothesis-generation`	Open new mechanism-level directions when no evidence-backed candidate is ready.
`experiment-design`	Turn current evidence into comparisons that distinguish competing explanations.
`four-methods`	Choose among SFT, RLVR, OPD, and model merging, including the checkpoint from which to continue.
`recipe-parameters`	Set and interpret coupled optimization, batch, sequence-length, and training-dose choices.
`data-strategy`	Translate a behavioral deficit into candidate sources, filtering, and data mixtures.
`evaluation-diagnosis`	Judge score differences using task-level results, cases, repeats, and evaluator identity.
`plateau-response`	Decide whether stalled progress calls for continuation, branching, composition, or a pivot.
`post-training-hazards`	Diagnose capability loss or mismatch between training signals and scores across SFT, RLVR, OPD, and merging.

D.3 Data Pool

All six campaigns use the same fixed pool of 264 corpora: 205 contain conversations or prompt–response pairs usable for SFT, and 59 contain prompts with checkable answers usable for RLVR. For each corpus the researcher can see its name, size, format, and source. Questions overlapping the evaluation benchmarks were filtered out before the pool was fixed, and the record keeps the exact corpora and row counts behind every recipe.

Table 9: **Main corpora the researcher trained on, by coverage.** Accepted experiments of the six campaigns used 51 corpora of the 264 in the pool (Figure 7); the list gives the main ones. Corpora never trained on include function-calling sets the researcher discussed but did not use (Hermes and several ShareGPT-format sets) and math sets such as NuminaMath-20k and Bespoke-Stratos-17k.

Coverage	Used for SFT	Used for RLVR
Instruction following	Dolci-Instruct (Team Olmo et al., 2025); Tulu-3 mixture and personas (Lambert et al., 2025); Magpie-Pro-300K (Xu et al., 2025); AutoIF-61k (Dong et al., 2025)	AutoIF instruction checks (Dong et al., 2025)
Reasoning and general	Nemotron post-training v2 (Nathawani et al., 2025; NVIDIA, 2025); Dolci-Think (Team Olmo et al., 2025); MetaMathQA (Yu et al., 2024b); train-of-thought (agentlans, 2025)	Dolci-RL-Zero (mixture) (Team Olmo et al., 2025)
Function calling	xLAM function calling (Zhang et al., 2025); glaive function calling v2 (Glaive AI, 2023); Agent-FLAN (Chen et al., 2024)	—
Mathematics	—	Dolci-RL-Zero (math) (Team Olmo et al., 2025); Skywork-OR1 (He et al., 2025); AceReason-Math (Chen et al., 2025); DeepMath-103K (He et al., 2026)
Code	—	Skywork-OR1 (code) (He et al., 2025); MBPP-extended (Redwood Research, 2023); OpenCodeInstruct (MBPP subset) (Ahmad et al., 2025)
Science and knowledge	—	ScienceQA (Lu et al., 2022); ARC (Clark et al., 2018); SciQ (Welbl et al., 2017); CommonsenseQA (Talmor et al., 2019); QASC (Khot et al., 2020); MedQA-USMLE (Jin et al., 2021); JEEBench (Arora et al., 2023)

Figure 7 shows how much of the pool the six campaigns touched. The researcher's catalog searches returned 211 of the 264 corpora, and accepted experiments trained on 51 of them. Coverage differs by domain: 10 of 65 searched mathematics corpora were used, none of the 12 preference corpora, and 8 of 9 science QA corpora. Use is also concentrated: five corpora account for 44% of all corpus uses in accepted experiments, the pattern behind the local search of Section 4.3.

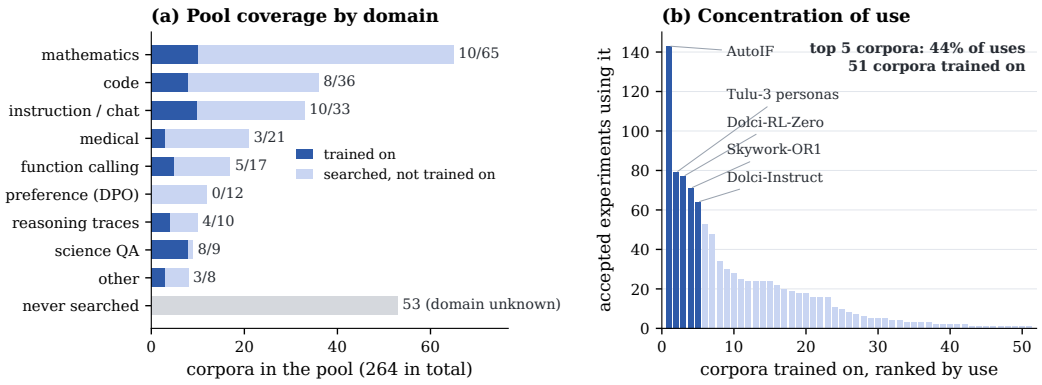


Figure 7: **Data-pool coverage of the six campaigns.** (a) Corpora in the pool by domain: trained on in at least one accepted experiment, returned by the researcher's catalog searches but never trained on, or never returned by any search. (b) The 51 corpora trained on, ranked by the number of accepted experiments that used each.

D.4 Visible and Held-Out Suites

The two suites (Table 10) cover related capability families. Core7 comprises 6,483 items: the full GSM8K test set evaluated zero-shot (1,319), HumanEval+ (164), thirteen BFCL categories (3,641), GPQA Diamond (198), IFEval (541), a 500-question MMLU-Pro sample, and AIME 2025 with four samples per problem (120); all other tasks use greedy decoding, with fixed per-task output limits and a presence penalty of 1.5. Repeated evaluations of the same weights still differ slightly (Table 13). OOD7 comprises 2,105 items under the same decoding settings. Every score in the paper uses these two fixed suites. OOD7 tests whether gains selected on Core7 carry over to unseen benchmarks.

Table 10: **Visible and held-out evaluation suites.** Core7 guides online search; OOD7 remains sealed until the campaign ends.

Suite	Benchmarks	Visibility and use
Core7	AIME 2025 (Balunovic et al., 2025); BFCL (Patil et al., 2025); GPQA Diamond (Rein et al., 2024); GSM8K (Cobbe et al., 2021); HumanEval+ (Chen et al., 2021; Liu et al., 2023); IFEval (Zhou et al., 2023); MMLU-Pro (Wang et al., 2024)	The researcher receives all seven scores. Their unweighted average ranks checkpoints and defines the online best.
OOD7	AIME 2026 and HMMT February 2026 (Balunovic et al., 2025); MATH-500 (Hendrycks et al., 2021; Lightman et al., 2024); LiveCodeBench (Jain et al., 2025); MBPP+ (Austin et al., 2021; Liu et al., 2023); IFBench (Pyatkin et al., 2025); SuperGPQA (Du et al., 2025)	Hidden from the researcher and used only for the final evaluation.

D.5 Checkpoint Selection and Measurement

Every Core7 evaluation returns all seven scores and their average. The online best is the highest average returned so far. Where a checkpoint was evaluated more than once, the repeats show how much its score varies (Table 13).

E Additional Results and Research Traces

E.1 What Was Evaluated

Table 11: **Core7 evaluations by the method of the evaluated checkpoint.** "Re-eval" is a requested repeat evaluation of an existing checkpoint.

Campaign	SFT	RLVR	Merge	OPD	Re-eval	Total
C1	22	18	26	0	5	71
C2	48	32	31	0	5	116
C3	59	88	10	6	1	164
C4	10	78	20	1	21	130
C5	19	42	24	2	25	112
C6	24	69	10	4	11	118
All	182	327	121	13	68	711

Table 12: **Outcomes of accepted experiments.** The researcher withdrew some experiments before launch. "Measured" counts experiments with a Core7 score: re-evaluations (Table 11) plus "New", the evaluated new checkpoints, each compared with its parent's best score against the 1.3-point noise threshold: better, within, or worse. "Repeats" counts Core7 readings of checkpoints already read, plus first readings of inherited ones. Eval share is evaluation's share of GPU hours.

Campaign	Accepted	Withdrawn	Failed	Measured	New	Improved	Within	Worse	Repeats	Eval share
C1	146	45	24	65	60	26	24	10	2	35%
C2	153	13	7	97	92	42	40	10	3	38%
C3	136	14	2	114	113	46	50	17	1	31%
C4	150	12	39	91	70	2	55	13	21	32%
C5	124	8	14	91	66	2	42	22	20	53%
C6	130	29	6	74	63	11	40	12	6	27%
All	839	121	92	532	464	129	251	84	53	

E.2 Campaign Sequence and Held-Out Transfer

C4, C5, and C6 start with 80, 91, and 97 checkpoints kept from earlier campaigns. C4's best score is a TIES merge of RLVR checkpoints trained from an inherited checkpoint, and C5's is a repeat evaluation of an inherited RLVR checkpoint (Appendix E.4). C6's best online score, 63.21, belongs to a checkpoint built from the base within the campaign: two 3,300-step SFT stages followed by five RLVR stages of 100–150 steps.

Final evaluation. After the sixth campaign ended, its four leading candidates were re-evaluated on Core7 and OOD7. The selected checkpoint, the 63.21 online best, scores 62.84 on Core7 (AIME 2025 14.2, BFCL 69.0, GPQA Diamond 39.4, GSM8K 92.5, HumanEval+ 82.3, IFEval 78.7, MMLU-Pro 63.8) and 32.04 on OOD7 (AIME 2026 8.3, HMMT 5.3, IFBench 20.0, LiveCodeBench 17.1, MATH-500 72.4, MBPP+ 69.0, SuperGPQA 32.2); the other three candidates score 62.79, 62.11, and 61.79 on Core7 and 32.39, 32.31, and 32.69 on OOD7. The re-evaluation lies within the repeat noise of the online score (Table 13), so the main text reports the online best.

E.3 Search Dynamics of the First Campaign

The first campaign's early SFT proposals score 50.67, 51.64, and 54.84, and a second Dolci-Instruct stage reaches 55.90. RLVR on that two-stage SFT checkpoint raises the frontier to 57.21 at step 100 and 57.83 at step 200, and two further 50-step RLVR stages reach 58.24 and 58.26; the best, 59.51, applies 200 RLVR steps to a DARE-TIES merge of three fine-tuned checkpoints (56.09).

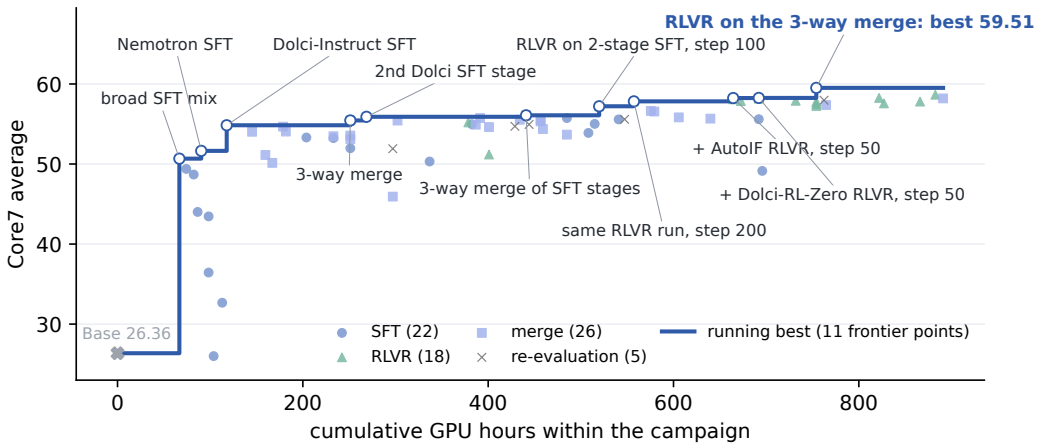


Figure 8: **Search trajectory of the first campaign.** Every returned Core7 evaluation is placed at the cumulative GPU hours of the campaign and colored by its post-training method; crosses are re-evaluations of existing checkpoints. The dark-blue staircase is the observed running best, and each of its 11 points is labeled with the recipe that produced it.

E.4 Continued Research and Recipe Interactions

Continued improvement. The fourth campaign's best online score, 62.07, is a TIES merge of RLVR checkpoints trained from an inherited checkpoint; one of those RLVR checkpoints first scores 62.03 and, inherited by the fifth campaign, reaches 62.24 on a later evaluation. Table 13 collects all seven evaluations of that checkpoint on the same Core7 suite, some with different evaluation seeds. Their spread shows how far a single best reading can sit above the checkpoint's mean.

Table 13: **Repeated evaluation of one checkpoint.** All seven Core7 evaluations in C4–C6 score the same model weights; sd is their standard deviation. Pooled over all 30 checkpoints read more than once (83 readings), one reading has a standard deviation of 0.42.

Core7 evaluations	Mean ± sd	Best observed
62.03, 60.63, 62.24, 61.43, 61.83, 62.23, 61.57	61.71 ± 0.57	62.24

Recipe interactions. Table 14 shows two checkpoints with nearly identical initial scores but opposite responses to the same instruction-verification RLVR recipe. A reasoning-data recipe improves the second checkpoint instead. The researcher records the lesson as *"a route that won in the 58-band is a hypothesis on a new parent, not a result"* and continues from the improved checkpoints. In a separate comparison, it keeps the two merge parents and settings fixed and changes only the reference checkpoint from which each parent's weight changes are measured: their shared post-trained ancestor gives 59.41, the raw base 58.50. Each result is a single evaluation.

Table 14: **Recipe effects depend on the checkpoint lineage.** RLVR uses the same settings across three source–parent combinations; the merge comparison changes only the reference checkpoint.

Intervention	Starting checkpoint	Before	After
Instruction-verification RLVR	Parent A	58.37	60.14
Instruction-verification RLVR	Parent B	58.36	57.84
Reasoning-data RLVR	Parent B	58.36	59.43
TIES with raw-base reference	Parents A and B	—	58.50
TIES with shared-ancestor reference	Parents A and B	—	59.41

E.5 Evidence for the Diagnosis Case

Table 15: **Per-task Core7 scores behind the C2 diagnosis case.** Both rows are single evaluations, as quoted in the researcher's status notes. The second-stage SFT raises BFCL and AIME, the two tasks where the first leader trailed the campaign's best scores most, while IFEval and GPQA fall, leaving AIME and GPQA as the deficits the researcher names below.

Checkpoint	AIME	BFCL	GPQA	GSM8K	HumanEval+	IFEval	MMLU-Pro	Core7
Dolci SFT from base	4.2	45.5	34.8	88.0	69.5	73.9	57.2	53.30
Second-stage SFT	7.5	56.4	30.8	89.7	75.0	62.5	58.0	54.27

Agent research trace: a per-task reading changes the recipe (C2)

Evidence. The leader at 53.30 was a single-corpus Dolci-Instruct SFT from the base. Over two sessions (*wakes*) the researcher had credited first the data mixture (*composition*) and then the first-stage SFT corpus (the *carrier*) for raising the Core7 average. A light second-stage SFT on a weaker checkpoint then reached 54.27.

RESEARCHER *Wake 15 said composition, wake 16 overturned it for carrier; the truth is both, and carrier is not first. The best carrier ever screened (Dolci-Instruct-SFT, 53.300) is beaten by a lesser carrier plus 4 gpu_h of stage-2 . . . I believe the next +1 comes from the stage-2 operator applied somewhere new.*

Action. The researcher withdrew "carrier is the first lever," named AIME and GPQA as the leader's remaining deficits, and proposed merging with the RL checkpoint that held them.

E.6 Research Trajectories

E.6.1 A Complete Research Episode

The four notes below span eleven hours of C4, during which the researcher's noise threshold (its *ruler*) moved twice and two of its own conclusions were retracted. In the notes, a *wake* is a researcher session, *macro* is the Core7 average, and s50, s100, and s150 are checkpoints at those RLVR steps. Quotes keep the researcher's wording; all-caps text is set in lower case.

How C4 set its noise threshold

07:52. *The comparison ruler is 0.50 macro, not 1.2 . . . Wakes 12–22 . . . called anything inside 1.2 a tie.*

08:28. *The comparison ruler is* $\sim$1.3 *macro, not 0.50, and the top of this board is one big tie . . . [the best checkpoint scored] 60.629 [at evaluation seed 1] . . . 62.029 [at seed 0]. Same weights, same frozen 6483 items . . . eval seed is the only difference.*

12:29. *I retracted my own wake-41 headline . . . Wake 41 read s50 61.643* $\rightarrow$ *s100 60.671 as 'dose turned,* -0.97*' . . . The s150 [checkpoint] landed at 62.000: the three . . . span 1.33 against a 1.27 ruler and are one noise band.*

18:52. *It read [a checkpoint] scoring 62.000 at both seeds as proof of . . . stability. 139 items flipped underneath that number; the agreement is cancellation . . . not stability.*

Outcome. C5 and C6 inherited this 1.3-point noise threshold through the record; Section 4.3 uses it too.

E.6.2 Training Dose

Agent research trace: a dose rule fails (C1)

Evidence. Four sessions (*wakes*) of C1 had fixed RLVR at 50 steps. One RLVR run on a single checkpoint (a *ladder*, saved at several step counts) then scored 58.09, 58.17, and 59.51 at steps 50, 100, and 200 (s50, s100, s200).

RESEARCHER *The dose rule was wrong. One ladder, one variable: s50 58.086, s100 58.171 (flat, 0.24 sd), s200 59.514 (+1.34, 3.8 sd). RLVR here is plateau-then-break; four wakes closed dose at 50 from ladders that stopped inside the flat window. Never default to steps=50.*

Outcome. The 200-step checkpoint became C1's best, 59.51.