ONE POLICY, MANY HARNESSES: REINFORCEMENT LEARNING FOR TRANSFERABLE HARNESS ADAPTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning has improved the performance of coding agents on repository-level software engineering tasks. However, it remains unclear whether the improvement after RL reflects success on more distinct tasks or on the same tasks under more harnesses. We conduct eight training experiments using PPO, alongside one GRPO comparison. Together with the base model, these yield ten Qwen3.5-27B checkpoints evaluated in 40 checkpoint–harness combinations on 731 SWE-Bench Pro tasks, combination produce 87,720 evaluation sessions for $\mathrm{Pass@3}$ evaluation. The key findings are: ① Task-level analysis shows that harness-specific gains primarily transfer from tasks the base model had already solved through another harness. The main gains increase success shared across harnesses without a comparable expansion in total observed task coverage. ② Gains can reach harnesses excluded from RL training, but transfer is direction-dependent: improvements in one harness can coexist with regressions in others. ③ Execution analysis links improved task success to more frequent delivery of nonempty patches, highlighting the importance of carrying edits through to an executable artifact. Together, these findings identify harness adaptation as a substantial component of the observed RL gains. They motivate training and evaluating coding agents for reliable execution across deployment harnesses, rather than treating a higher score in one harness as evidence of broader task-solving competence.

1 INTRODUCTION

Large language model (LLM) based coding agents interact with the underlying coding environment through a harness that exposes tools, parses proposed actions, manages interaction history, and determines when execution terminates. Harness design therefore shapes both coding-agent behavior and task success (Yang et al., 2024; Yao et al., 2026), which means the same LLM can succeed or fail on a software task depending on the harness through which it acts. For a model deployed through several harnesses, the practical question is which tasks succeed and how consistently that success appears across harnesses.

Reinforcement learning (RL) now trains coding agents directly through harness loops (He et al., 2026; Xu et al., 2026; Yu et al., 2026). Studies of harness design and post-training show that training and deployment configurations interact (Kim et al., 2026). An improved score under one harness, however, leaves two outcomes entangled: success on additional tasks and success on the same tasks under additional harnesses. It is important to figure out how these outcomes change together after harness RL whenever a model must work across several execution systems rather than a single preferred harness.

In this paper, we study Qwen3.5-27B and checkpoints trained through a diverse harnesses loop. We first formulate the harness-conditioned Markov Decision Process (MDP) to capture how different harnesses shape the interaction dynamics of a shared model policy. This formulation provides a common basis for studying policy optimization within harnesses and the transfer of learned behavior across them. To optimize a shared policy across heterogeneous harness-conditioned decision processes, we construct a unified actor–critic learning framework based on PPO. The frame-

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.

work couples grouped sampling with trajectory-normalized policy and value objectives, enabling joint optimization over agent trajectories with different execution structures. A shared critic provides context-dependent credit assignment, while trajectory-level aggregation assigns equal nominal weight to rollouts despite differences in length and segmentation.

The evaluation crosses ten checkpoints with Codex, OpenCode, Pi, and Claude Code on the same 731 SWE-Bench Pro tasks (Deng et al., 2025), producing 87,720 sessions for Pass@3 evaluation. We track task-level outcomes across checkpoints and harnesses to distinguish broader task coverage from success on the same tasks under more harnesses. We further analyze execution trajectories to examine how these performance changes relate to agents' ability to produce and deliver repository patches.

Our evaluation yields three key findings about harness RL. ① The main gains reflect broader sharing of task success across harnesses rather than a comparable expansion in observed task coverage. Newly solved tasks under an improved harness predominantly overlap tasks that Base had already solved elsewhere, reducing dependence on a single favorable harness. ② Transfer extends beyond the training harness but remains direction-dependent. A checkpoint can improve a held-out harness while leaving other harnesses unchanged or worse, so the training harness alone does not determine where benefits emerge. ③ Improved task success is accompanied by more frequent nonempty-patch delivery. The accompanying trajectory evidence distinguishes preparing a candidate modification from applying it to the repository and delivering it for evaluation.

Our contributions are threefold:

1. We formulate a harness-conditioned MDP that separates shared policy parameters from harness-dependent execution dynamics, providing a common basis for policy learning and transfer analysis.
2. We construct a unified PPO-based actor–critic framework that combines grouped sampling, shared-critic advantage estimation, and trajectory-normalized policy and value objectives. It accommodates both single- and multi-harness training without replacing harness execution loops.
3. We present a cross-harness empirical study that integrates checkpoint-level evaluation, task-level analysis, and execution traces to characterize where RL gains arise and how they transfer.

2 Harness-Conditioned Policy Learning

A harness specifies how generated tokens become actions and how their outcomes enter the next model context. For task z and harness $h \in \mathcal{H}$, the policy $\pi_\theta(x_t \mid c_t)$ acts through its harness-provided context c_t, including prompts, tool schemas, interaction history, and response prefixes. The full state $s = (e, m, c)$ includes the environment e, harness memory and remaining budgets m, and policy context c. Token generation and harness-conditioned transitions follow

$$x_t \sim \pi_\theta(\cdot \mid c_t), \qquad s_{t+1} \sim P_h(\cdot \mid s_t, x_t). \tag{1}$$

Let $J_h(\theta)$ denote expected per-attempt binary grader success under harness h, averaging over tasks from distribution $\mathcal{D}$ and trajectories generated by π_θ. For nonnegative harness weights summing to one, the deployment objective is

$$J(\theta) = \sum_{h \in \mathcal{H}} w_h J_h(\theta). \tag{2}$$

Training approximates this goal through shaped returns and the sampled trajectory objective in Section 3. The task union and intersection defined below serve as evaluation statistics. A measured gain under a harness absent from training constitutes *held-out-harness transfer*.

Task coverage and cross-harness portability. For each checkpoint, a task is counted as solved under a harness if it has at least one success in the Pass@3 evaluation. The *observed union* counts tasks solved under at least one of the four harnesses; the *observed intersection* counts tasks solved under all four. The union uses twelve heterogeneous attempts. For a fixed checkpoint and target harness, let G contain tasks newly solved relative to Base under that harness. Let B contain tasks solved by Base under any harness. For $G \neq \varnothing$, define

$$\text{Overlap} = \frac{|G \cap B|}{|G|}. \tag{3}$$

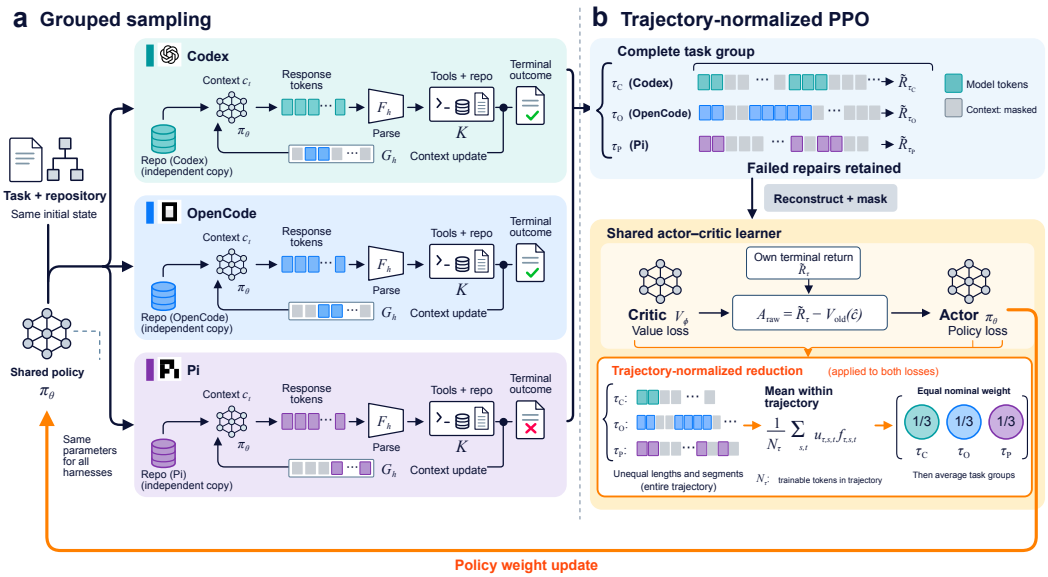


Figure 1: **Grouped sampling and trajectory-normalized PPO. a**, A shared policy executes the same task through independent harness and repository instances; three-harness sampling is illustrated. Each harness interprets responses, executes tools, and updates context (Appendix A.1). **b**, Complete task groups retain each trajectory's own terminal return, including failed repairs. Context tokens remain visible to the model and are masked from the loss. A shared critic supplies context-dependent advantages. Policy and value objectives average trainable response-token terms across each entire trajectory, then equally across trajectories and task groups. Each harness in the illustrated group receives nominal weight $1/3$. Updated policy parameters are shared by all harnesses.

This describes newly successful tasks: the fraction already solved under another Base harness. All three measures concern observed task identities at the fixed Pass@3 evaluation budget per harness.

3 Grouped Sampling and Trajectory-Normalized PPO

To study task identities across harnesses, we train checkpoints using PPO, GAE, and trajectory-token averaging with grouped sampling (Figure 1). The comparisons estimate performance differences between complete training recipes.

Grouped sampling. For each task, we sample n independent trajectories from the same initial task state and collect them into a group. In multi-harness PPO, the trajectories are distributed across harnesses and share one critic. In single-harness PPO, all n trajectories are sampled through the same harness and share one critic. Within each training run, the critic is shared across task groups and estimates returns from each trajectory's context. Complete groups with terminal outcomes, aligned token records, and trainable responses enter the update together, including successful and failed repairs.

Context-dependent credit assignment. Polar supplies model-call trajectories (Xu et al., 2026). Reconstruction preserves sampled token identities and masks harness-authored context from the loss. Every retained segment receives its trajectory's shaped return $\widetilde{R}_\tau$, a binary grader reward with a success-gated length penalty. The shared critic predicts this return from reconstructed context $\widehat{c}$. For a response position t within a trajectory, generalized advantage estimation with $\gamma = \lambda = 1$ and zero terminal bootstrap yields (Schulman et al., 2015)

$$A_t^{\text{raw}} = \widetilde{R}_\tau - V_{\phi^-}(\widehat{c}_t), \tag{4}$$

where V_{ϕ^-} is the pre-update critic; the value target at each response position is the trajectory return $\widetilde{R}_\tau$. Thus, even a group of failed repairs can retain context-dependent raw advantages, depend-

ing on its value estimates. Advantage standardization within sequence partitions precedes policy optimization; value targets retain the return scale.

Trajectory-normalized optimization. Let batch $\mathcal{B}$ contain M trajectories from complete task groups. For trajectory i, enumerate its N_i trainable response tokens across all retained segments. Both policy and value objectives use the trajectory-token averaging rule reported in Agent Lightning v1.0 (Eq. 16, attributed there to slime) (He et al., 2026):

$$\langle f \rangle_{\mathcal{B}} = \frac{1}{M} \sum_{i=1}^{M} \frac{1}{N_i} \sum_{t=1}^{N_i} f_{it}. \tag{5}$$

Here f_{it} is a token-level objective term, averaged first within each trajectory and then equally across trajectories. The actor maximizes the clipped PPO surrogate (Schulman et al., 2017), reweighted by capped captured-score weights. The critic minimizes a clipped squared-error loss using the same trajectory weights. Each trajectory receives nominal outer weight $1/M$, independent of its length or segmentation. Equal rollout allocation across harnesses gives each harness the same nominal weight. This weighting separates rollout allocation from within-trajectory credit assignment. Appendix A.3 gives the complete objectives, likelihood ratios, normalization, and optimization parameters; Appendix A.4 describes training configurations.

4 Experimental Design

4.1 Training recipes and evaluation matrix

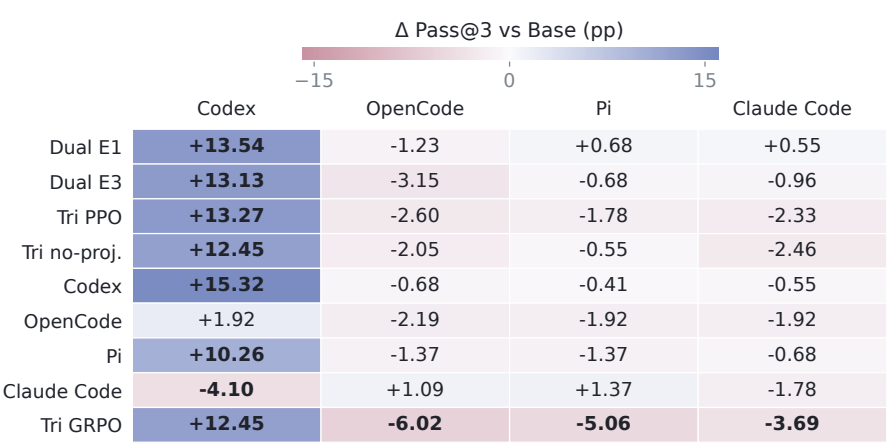
All conditions start from the same released Qwen3.5-27B parameters, denoted Base. PPO uses a source pool of 500 executable SWE-rebench V2 tasks from 500 repositories (Badertdinov et al., 2026). Each PPO update admits 50 task groups. The experimental matrix includes Base and nine trained checkpoints:

- **Dual E1/E3**: independent Base-initialized runs using Codex and OpenCode, with one trajectory per harness per task group. E1 admits 500 groups over 10 PPO updates; E3 admits 1,500 over 30 updates.
- **Tri PPO/no-projection**: Codex, OpenCode, and Pi, three trajectories per task group and 30 updates, with two reconstruction recipes.
- **Single-harness PPO**: the Codex, OpenCode, and Pi checkpoints, each trained through its named harness with three trajectories per task group and 30 updates. The Claude Code checkpoint uses its named harness with three trajectories, no projection, and 10 updates.
- **Tri GRPO**: a group-relative recipe on the same subset, with 24 groups per update and 30 updates. Each group contains eight trajectories from one harness; Codex, OpenCode, and Pi rotate across groups.

Single-harness checkpoints are named after their training harness throughout; evaluation harnesses specify where each checkpoint is deployed. The comparisons cover complete training recipes with the specified task and update budgets; each recipe has one training run.

Every checkpoint is evaluated through Codex, OpenCode, Pi, and Claude Code on the same 731 tasks in the SWE-Bench Pro public split (Deng et al., 2025). The $\mathrm{Pass@3}$ evaluation across all checkpoint–task–harness conditions yields 87,720 final sessions. Training and evaluation use a 256K-token context window, at most 8,192 generated tokens per model call, temperature 0.6, top-p 0.95, top-k 20, and a 3,600-second task limit.

The cohort contains 266 Python, 165 JavaScript, 280 Go, and 20 TypeScript tasks across 11 repositories. All observed successes occur in Python and JavaScript; every checkpoint–harness pair records zero successes on Go and TypeScript tasks. We retain all 731 tasks in every headline denominator and examine language strata in Appendix E.



	Codex	OpenCode	Pi	Claude Code
Dual E1	**+13.54**	-1.23	+0.68	+0.55
Dual E3	**+13.13**	-3.15	-0.68	-0.96
Tri PPO	**+13.27**	-2.60	-1.78	-2.33
Tri no-proj.	**+12.45**	-2.05	-0.55	-2.46
Codex	**+15.32**	-0.68	-0.41	-0.55
OpenCode	+1.92	-2.19	-1.92	-1.92
Pi	**+10.26**	-1.37	-1.37	-0.68
Claude Code	**-4.10**	+1.09	+1.37	-1.78
Tri GRPO	**+12.45**	**-6.02**	**-5.06**	**-3.69**

Figure 2: **Adaptation depends on the evaluation harness.** Base-relative Pass@3 changes (percentage points) on 731 paired tasks. Rows identify the checkpoints defined above; columns identify evaluation harnesses. Bold values mark Holm-significant changes across 36 comparisons. Appendix B reports all 36 confidence intervals and Holm-adjusted tests.

4.2 Evaluation

Pass@1 is the per-attempt success rate averaged across tasks; Pass@3 is the fraction of tasks solved at least once across $n = 3$ attempts. Appendix C.3 gives the general Pass@k estimator (Chen et al., 2021) and the additional Pass@2 results. Mean Pass@1 averages Pass@1 equally across the four harnesses. We also report the minimum and range of harness-specific Pass@3, together with the observed union, observed intersection, and overlap of newly successful tasks defined in Section 2.

Runtime status records how each session terminated, separately from grader success. ERROR and TIMEOUT remain in all session denominators. Four sessions are grader-successful despite a final ERROR and contribute to the pass estimates.

4.3 Execution outcomes

Execution-stage analysis uses the same 87,720 final sessions as the headline evaluation, linked to their corresponding grader reports. Each task–checkpoint–harness–attempt combination contributes one final session.

We report three stages: normal completion, nonempty patch delivery conditional on completion, and grader success conditional on a completed session with a nonempty patch. Patch delivery measures whether an edit reaches evaluation; conditional success measures whether that edit solves the task according to the grader.

4.4 Paired inference

The task is the paired inferential unit. Base-relative changes are $\widehat{D}_h^q = \widehat{\text{Pass@3}}_h^q - \widehat{\text{Pass@3}}_h^{\text{B}}$. We use 10,000 paired task-bootstrap replicates for pointwise 95% percentile intervals, and exact two-sided McNemar tests for paired binary Pass@3 outcomes. Holm correction covers all 36 trained-checkpoint–harness comparisons against Base. Direct Dual E1/E3 and projection/no-projection comparisons form a separate eight-test family.

A paired three-degree-of-freedom Wald test assesses equality of changes across harnesses, with Holm correction over nine checkpoints. Repository-cluster bootstrapping provides a sensitivity analysis for the Codex-minus-OpenCode change contrast. Complete tests appear in Appendix B.

Table 1: **Complete checkpoint-by-harness results.** Each cell uses 731 tasks with the Pass@3 evaluation budget. Values are percentages. Mean Pass@1 averages Pass@1 equally across the four harnesses.

Checkpoint	Codex		OpenCode		Pi		Claude Code		Mean Pass@1
	Pass@1	Pass@3	Pass@1	Pass@3	Pass@1	Pass@3	Pass@1	Pass@3	
Base	7.16	14.23	21.98	28.32	24.62	30.64	24.90	31.33	19.66
Dual E1	18.70	27.77	20.79	27.09	24.03	31.33	24.21	31.87	21.93
Dual E3	20.29	27.36	18.29	25.17	23.44	29.96	22.25	30.37	21.07
Tri PPO	20.84	27.50	19.52	25.72	22.44	28.86	21.98	29.00	21.19
Tri no-proj.	19.24	26.68	18.79	26.27	23.94	30.10	22.25	28.86	21.06
Codex	23.94	29.55	20.70	27.63	23.71	30.23	24.62	30.78	23.24
OpenCode	9.85	16.14	20.38	26.13	23.16	28.73	22.07	29.41	18.87
Pi	15.14	24.49	20.47	26.95	22.48	29.27	23.67	30.64	20.44
Claude Code	5.02	10.12	22.85	29.41	24.58	32.01	22.66	29.55	18.78
Tri GRPO	20.25	26.68	16.23	22.30	19.38	25.58	20.43	27.63	19.07

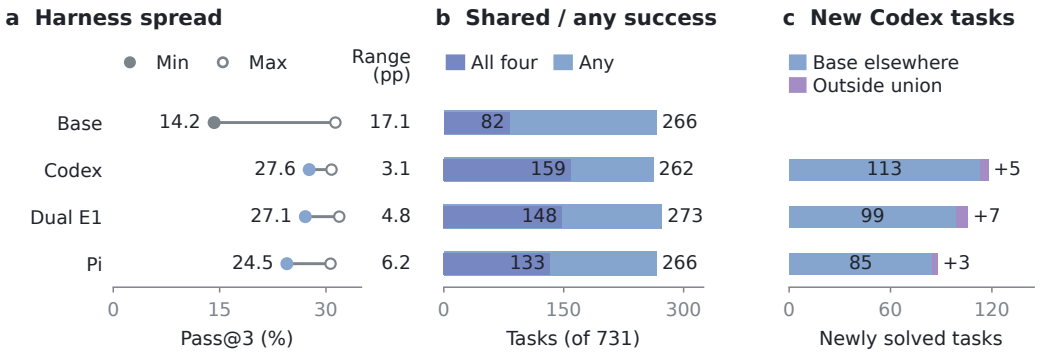


Figure 3: **Task coverage and cross-harness portability.** All panels use 731 tasks with the Pass@3 evaluation budget per harness. **a**, Minimum-to-maximum intervals of four harness-specific Pass@3 values; the separate range column gives percentage-point differences. **b**, Common success (all four) nested within observed coverage (any harness). **c**, New Codex successes partitioned into tasks Base solved elsewhere and tasks outside Base's observed union. Counts describe this fixed budget.

5 Results

The complete checkpoint-by-harness matrix concentrates training gains in Codex. Table 1 and Figure 2 show these results. Base Pass@3 ranged from 14.23% under Codex to 31.33% under Claude Code. The Codex checkpoint produced the largest Codex gain, 15.32 percentage points. The 95% confidence interval ranged from 12.59 to 18.06 percentage points. Its other three harness estimates were below Base. All seven Holm-significant gains occurred under Codex. The Claude Code checkpoint had a significant Codex loss and the highest OpenCode and Pi point estimates. Tri GRPO fell below Base on all three non-Codex harnesses. All nine trained checkpoints showed heterogeneous harness changes after correction. The largest adjusted p-value was 0.032.

5.1 The training gains across harnesses

The Codex checkpoint expanded the common-success intersection from 82 to 159 tasks; the observed union across all twelve attempts changed from 266 to 262 (Figure 3). The intersection gained 89 and lost 12 tasks; the union gained 12 and lost 16. Worst-harness Pass@3 rose to 27.63%, and the harness range contracted from 17.10 to 3.15 percentage points. Of 118 newly solved Codex tasks, 113 had Base successes elsewhere, including 85 solved under all three other harnesses. Five lay outside Base's observed union. For Dual E1, 99 of 106 new Codex successes overlapped Base successes elsewhere; its intersection and union contained 148 and 273 tasks.

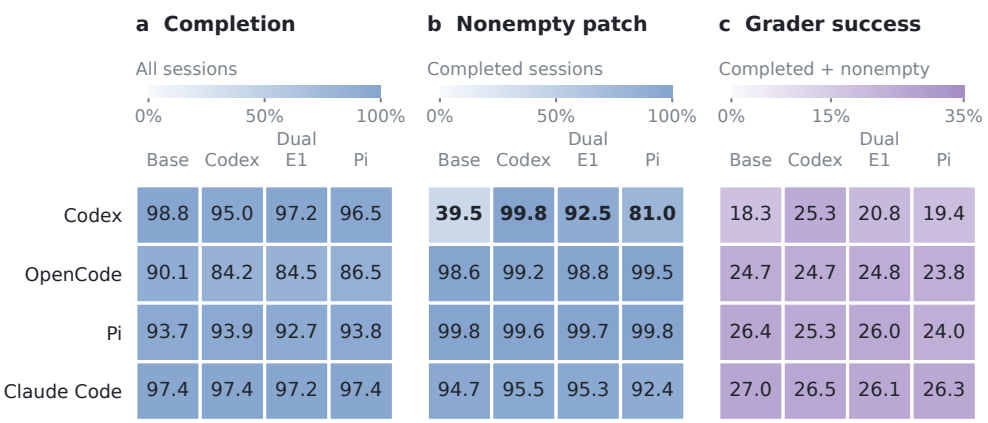


Figure 4: **Execution stages on the full evaluation cohort.** Four checkpoints are shown from the ten-checkpoint, 87,720-session analysis. Each checkpoint–harness cell contains 2,193 sessions from the Pass@3 evaluation of 731 tasks. **a**, Completion over all sessions, including runtime failures. **b**, Nonempty-patch delivery conditional on completion. **c**, Grader success conditional on completion and a nonempty patch. Rates use each checkpoint's own subsets. Rows identify evaluation harnesses and columns identify checkpoints. Color scales span 0–100% in **a,b** and 0–35% in **c**; bold values highlight Codex patch delivery.

5.2 Harnesses transfer

Pi training increased Codex Pass@3 from 14.23% to 24.49%, a 10.26-point gain (95% CI, 7.66 to 12.86; Holm-adjusted $p = 2.26 \times 10^{-13}$). This checkpoint was trained exclusively through Pi. Its Pi estimate changed by -1.37 points, with an interval spanning zero; the largest improvement occurred on the held-out harness. Of 88 newly solved Codex tasks, 85 had Base successes elsewhere; 13 previously solved Codex tasks lost success. The intersection reached 133 tasks, and the union remained at 266, with 14 tasks entering and 14 leaving.

Descriptive point estimates on the 45-task SWE-bench Verified cohort followed the same Pi-to-Codex direction (Appendix F). Codex Pass@3 rose from 35.56% to 42.22%, and Pi fell from 44.44% to 31.11%. The best trained four-harness mean Pass@3 matched Base; all trained worst-harness estimates were at or below Base.

5.3 Codex gains accompanied increased nonempty patch delivery

Under Codex, gains accompanied more frequent nonempty-patch delivery (Figure 4). Base completed 98.81% of sessions, of which 39.55% delivered a nonempty patch. For the Codex checkpoint under Codex, completion decreased to 94.98% and conditional delivery increased to 99.76%. Grader success among completed sessions with nonempty patches rose from 18.32% (857 sessions) to 25.26% (2,078 sessions). Under the other three harnesses, Base already delivered nonempty patches in more than 94% of completions. For the Pi checkpoint under Codex, completion was 96.49%, delivery was 81.00%, and grader success among the 1,714 delivered patches was 19.37%. This combines a large delivery increase with a smaller conditional-success increase.

An inspected AIX facts task connects preparation to delivery. Base under Codex prepared a candidate change and stopped before applying it. A Pi-trained Codex session wrote the modification to the repository and submitted a grader-accepted patch.

5.4 Dual training budgets differed in repeated success and task coverage

Dual E3 increased Codex per-attempt success and reduced Pass@3 coverage relative to E1 (Figure 5). These independent Base-initialized runs used 10 and 30 updates. Codex Pass@1 was 18.70% and 20.29%; Pass@3 was 27.77% and 27.36% for E1/E3, respectively. They solved 76 and 92 tasks

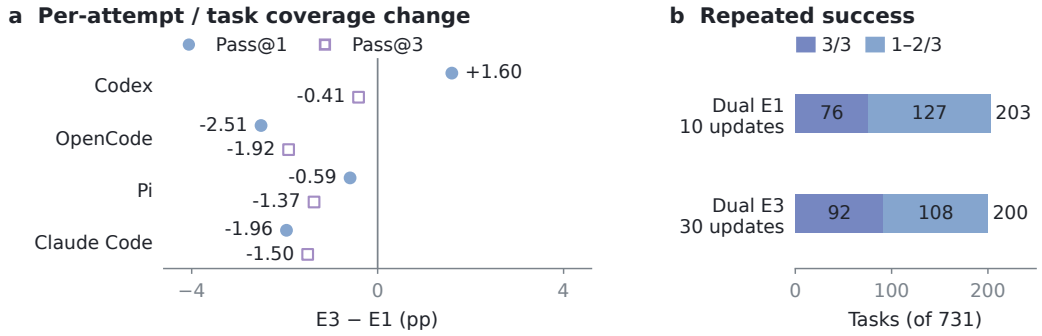


Figure 5: **Per-attempt success and** Pass@3 **coverage can diverge.** Dual E1/E3 are independent Base-initialized endpoints after 10/30 updates. **a**, E3-minus-E1 differences over 731 paired tasks: filled circles, Pass@1; open squares, Pass@3. **b**, Codex tasks with $3/3$ or $1/3$–$2/3$ successful attempts in Pass@3 evaluation; totals count tasks solved at least once.

three times, and 203 and 200 at least once. E3 had lower Pass@3 under all four harnesses, with a mean difference of -1.30 points (95% paired bootstrap CI, -2.36 to -0.27). Its three non-Codex Pass@1 estimates also decreased. Thus, per-attempt success and fixed-budget coverage rank the Codex endpoints differently.

Codex and Dual E1 had the highest four-harness mean Pass@3, above both Tri PPO recipes. No-projection minus projection averaged 0.21 points (95% CI, -0.79 to 1.20), with mixed harness-specific directions. Tri GRPO fell below Tri PPO on every harness; their source pools contained 190 and 500 tasks, respectively. These comparisons concern complete recipes, including harness mixtures, reconstruction, optimization, and training pools.

5.5 Summary

① **The main gains broadened success shared across harnesses.** For the Codex checkpoint, the common-success intersection nearly doubled while the observed union stayed close to Base. Most newly solved Codex tasks under the Codex, Dual E1, and Pi checkpoints had already been solved by Base through other harnesses. The combination of a larger intersection and a narrower harness-performance range indicates more consistent success across execution interfaces within the fixed Pass@3 evaluation budget.

② **Transfer extended to held-out harnesses and remained direction-dependent.** Pi training produced its largest improvement under Codex, which was excluded from its training rollouts. The same checkpoint's Pi point estimate was below Base. Across the full matrix, all Holm-significant gains occurred under Codex, and several checkpoints declined under other harnesses. The benefits of training therefore depended on the pairing of training and deployment harnesses, making the full cross-harness evaluation essential to characterizing transfer.

③ **Improved task success accompanied more frequent nonempty-patch delivery.** Under Codex, both the Codex and Pi checkpoints increased patch delivery among completed sessions, alongside higher grader success among delivered patches. The AIX case illustrates the distinction between preparing an edit and applying it to the repository for evaluation. Together, the aggregate rates and inspected trajectory support a behavioral interpretation of the observed gains, linking task success to the completion of repository edits and their delivery through the harness.

6 Discussion

At the fixed evaluation budget, most new Codex successes were tasks Base had solved elsewhere, while the observed intersection nearly doubled and the union changed little (Section 5.1). The overlap describes the composition of newly successful tasks at the fixed sampling budget. Latent task-solving ability and the causal contribution of RL require separate identification. Pi-to-Codex transfer

shows that the training interface can differ from the principal beneficiary. The associated patch-delivery increase provides an execution-level counterpart to these task-level changes (Section 5.3). Together, these observations motivate evaluating task identities and artifact delivery alongside aggregate success.

The conclusions concern fixed checkpoints of Qwen3.5-27B at the stated attempt budgets. Each recipe has one training run; recipe comparisons include their specified differences in data and optimization. Successes occur in Python and JavaScript, and the 45-task Verified extension shows direction-specific transfer with every trained minimum harness score at or below Base. The execution-stage association and inspected case support a behavioral interpretation; controlled interventions would be needed to identify its causal contribution.

7 Related Work

Harness-dependent performance. SWE-agent links interface design to software-task success (Yang et al., 2024). Harness-Bench evaluates model–harness pairings (Yao et al., 2026), and Kim et al. (2026) examine harness design and post-training jointly. Orchard reports a matched-harness advantage after SFT (Peng et al., 2026); Qwen3-Coder-Next reports asymmetric cross-scaffold transfer after mid-training and improved format generalization with diverse tool templates (Cao et al., 2026). We track task identities across RL checkpoints to distinguish shared cross-harness success from total observed task coverage.

Harness reinforcement learning. Agent Lightning separates agent execution from optimization (Luo et al., 2025; He et al., 2026). Polar captures model calls within harnesses and demonstrates GRPO (Xu et al., 2026); OpenForgeRL trains agents through harnesses across environments (Yu et al., 2026). ClawGym II reports PPO, GRPO, and mixed-harness GRPO (Song et al., 2026). We use established PPO (Schulman et al., 2017), generalized advantage estimation (Schulman et al., 2015), and the trajectory-token averaging rule reported by Agent Lightning v1.0 (He et al., 2026) to study task identities across training and deployment harnesses.

Credit assignment and portability. GRPO constructs a relative baseline from same-question rewards (Shao et al., 2024). Using a Qwen3-8B supervised warm start, Le et al. (2026) find no detectable held-out advantage from cross-harness over within-harness GRPO grouping. They primarily use frozen trajectories, with partial on-policy recollection and repeated-seed controls. Their cross-minus-within grouping comparison holds harness exposure fixed. Our held-out transfer comparison is trained-checkpoint minus Base, a different estimand. We examine which task identities change success, alongside observed union, intersection, and patch delivery.

Success under finite sampling budgets. The $\mathrm{Pass@}k$ framework measures success across attempt budgets (Chen et al., 2021). Yue et al. (2025) report improved low-budget success after RL and broader base-model coverage at large budgets. ProRL reports coverage gains under prolonged RL (Liu et al., 2025). We jointly track solved tasks and their successful harnesses under the $\mathrm{Pass@3}$ evaluation budget.

8 Conclusion

Harness-conditioned policy learning exposes two distinct outcomes of harness RL: observed task coverage and success shared across execution interfaces. Our task-level evaluation locates the main gains in shared success, with direction-dependent transfer to held-out harnesses and increased nonempty-patch delivery. These results motivate cross-harness portability as an explicit training and evaluation objective alongside fixed-budget task coverage.

AI USE STATEMENT

Generative AI assisted with manuscript restructuring and language editing, template migration, computational figure reformatting, and checks against experimental code, evaluation records, and trajectories.

REFERENCES

Ibragim Badertdinov, Maksim Nekrashevich, Anton Shevtsov, and Alexander Golubev. SWE-rebench V2: Language-agnostic SWE task collection at scale. *arXiv preprint arXiv:2602.23866*, 2026. doi: 10.48550/arXiv.2602.23866. URL https://arxiv.org/abs/2602.23866.

Ruisheng Cao, Mouxiang Chen, Jiawei Chen, Zeyu Cui, Yunlong Feng, Binyuan Hui, Yuheng Jing, Kaixin Li, Mingze Li, Junyang Lin, Zeyao Ma, Kashun Shum, Xuwu Wang, Jinxi Wei, Jiaxi Yang, Jiajun Zhang, Lei Zhang, Zongmeng Zhang, Wenting Zhao, and Fan Zhou. Qwen3-Coder-Next technical report. *arXiv preprint arXiv:2603.00729*, 2026. doi: 10.48550/arXiv.2603.00729. URL https://arxiv.org/abs/2603.00729.

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. doi: 10.48550/arXiv.2107.03374. URL https://arxiv.org/abs/2107.03374.

Xiang Deng, Jeff Da, Edwin Pan, Yannis Yiming He, Charles Ide, Kanak Garg, Niklas Lauffer, Andrew Park, Nitin Pasari, Chetan Rane, Karmini Sampath, Maya Krishnan, Srivatsa Kundurthy, Sean Hendryx, Zifan Wang, Vijay Bharadwaj, Jeff Holm, Raja Aluri, Chen Bo Calvin Zhang, Noah Jacobson, Bing Liu, and Brad Kenstler. SWE-Bench Pro: Can AI agents solve long-horizon software engineering tasks? *arXiv preprint arXiv:2509.16941*, 2025. doi: 10.48550/arXiv.2509.16941. URL https://arxiv.org/abs/2509.16941.

Zhiyuan He, Siwei Zhang, Zhiwen Zhou, Yuqing Yang, Yu Kang, Yuge Zhang, Luna K. Qiu, Tin Yan Tsui, Jiahang Xu, and Chong Luo. Agent Lightning v1.0: Towards harnessed agentic RL. *arXiv preprint arXiv:2608.17528*, 2026. doi: 10.48550/arXiv.2608.17528. URL https://arxiv.org/abs/2608.17528.

Kyungmin Kim, Youngbin Choi, Seoyeon Lee, Suhyeon Jun, Dongwoo Kim, and Sangdon Park. The interplay of harness design and post-training in LLM agents. *arXiv preprint arXiv:2606.25447*, 2026. doi: 10.48550/arXiv.2606.25447. URL https://arxiv.org/abs/2606.25447.

Chenqian Le, Jiayi Cheng, Qijia He, Runhao Li, Yinghao Li, and Xupeng Chen. What does multi-harness RL learn? credit assignment and portability in coding agents. *arXiv preprint arXiv:2609.04518*, 2026. doi: 10.48550/arXiv.2609.04518. URL https://arxiv.org/abs/2609.04518.

Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. ProRL: Prolonged reinforcement learning expands reasoning boundaries in large language models. *arXiv preprint arXiv:2505.24864*, 2025. doi: 10.48550/arXiv.2505.24864. URL https://arxiv.org/abs/2505.24864.

Xufang Luo, Yuge Zhang, Zhiyuan He, Zilong Wang, Siyun Zhao, Dongsheng Li, Luna K. Qiu, and Yuqing Yang. Agent Lightning: Train ANY AI agents with reinforcement learning. *arXiv preprint arXiv:2508.03680*, 2025. doi: 10.48550/arXiv.2508.03680. URL https://arxiv.org/abs/2508.03680.

OpenAI. Introducing SWE-bench Verified. OpenAI research release, August 13, 2024.

Baolin Peng, Wenlin Yao, Qianhui Wu, Hao Cheng, Xiao Yu, Rui Yang, Tao Ge, Alessandro Sordoni, Xingdi Yuan, Yelong Shen, Pengcheng He, Tong Zhang, Zhou Yu, and Jianfeng Gao. Orchard: An open-source agentic modeling framework. *arXiv preprint arXiv:2605.15040*, 2026. doi: 10.48550/arXiv.2605.15040. URL https://arxiv.org/abs/2605.15040v1.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015. doi: 10.48550/arXiv.1506.02438. URL https://arxiv.org/abs/1506.02438.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. doi: 10.48550/arXiv.1707.06347. URL https://arxiv.org/abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. doi: 10.48550/arXiv.2402.03300. URL https://arxiv.org/abs/2402.03300.

Huatong Song, Fei Bai, Ming Yang, Renyuan Li, Jia Deng, Jujie He, Zhange Zhang, Daixuan Cheng, Yan Xing, Qi Yun, Xuxing Chen, Danyang Li, Feng Chang, Chuan Hao, Ran Tao, Jian Yang, Bryan Dai, Wayne Xin Zhao, Mingjie Tang, and Ji-Rong Wen. ClawGym II: Exploring black-box RL on agent harness. *arXiv preprint arXiv:2608.16798*, 2026. doi: 10.48550/arXiv.2608.16798. URL https://arxiv.org/abs/2608.16798.

Binfeng Xu, Hao Zhang, Shaokun Zhang, Songyang Han, Mingjie Liu, Jian Hu, Shizhe Diao, Zhenghui Jin, Yunheng Zou, Michael Demoret, Jan Kautz, and Yi Dong. Polar: Agentic RL on any harness at scale. *arXiv preprint arXiv:2605.24220*, 2026. doi: 10.48550/arXiv.2605.24220. URL https://arxiv.org/abs/2605.24220.

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

Yilun Yao, Xinyu Tan, Chao-Hsuan Liu, Yaoming Li, Zhengyang Wang, Wenhan Yu, Zhewen Tan, Yuxuan Tian, Guangxiang Zhao, Lin Sun, Xiangzheng Zhang, and Tong Yang. Harness-Bench: Measuring harness effects across models in realistic agent workflows. *arXiv preprint arXiv:2605.27922*, 2026. doi: 10.48550/arXiv.2605.27922. URL https://arxiv.org/abs/2605.27922.

Xiao Yu, Baolin Peng, Ruize Xu, Hao Zou, Qianhui Wu, Hao Cheng, Wenlin Yao, Nikhil Singh, Zhou Yu, and Jianfeng Gao. OpenForgeRL: Train harness-native agents in any environment. *arXiv preprint arXiv:2607.21557*, 2026. doi: 10.48550/arXiv.2607.21557. URL https://arxiv.org/abs/2607.21557.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025. doi: 10.48550/arXiv.2504.13837. URL https://arxiv.org/abs/2504.13837.

A Reconstruction and Optimization

A.1 Harness-dependent dynamics

For the full state $s = (e, m, c)$ defined in Section 2, the harness-dependent transition law factors as

$$P_h(s' \mid s, x) = \sum_{a,y} F_h(a \mid s, x)\, K(e', y \mid e, a)\, G_h(m', c' \mid s, x, a, y), \tag{6}$$

where $s' = (e', m', c')$, a is the interpreted action, and y is its execution output. Here F_h interprets generated tokens, K executes actions, and G_h updates harness memory and model context. Ordinary response tokens extend the prefix through a null environment action.

Harness configurations. Codex, OpenCode, Pi, and Claude Code use the corresponding Qwen3.5-27B checkpoint. Codex uses extra-high reasoning effort; OpenCode enables reasoning and tool calling. Each harness uses fixed settings across checkpoints in SWE-Bench Pro evaluation.

A.2 Event-derived reconstruction

Harnesses retain their tool schemas, parsers, context management, recovery logic, and termination rules. Following Polar's model-call architecture (Xu et al., 2026), capture records response token IDs, decoder-returned scores, and each call's canonical prompt tokens, including the chat template and tool schema.

Projection. Captured response spans are matched to reasoning, content, or tool items in harness records.

For the projection recipes, matched character spans, including structural delimiters and stop suffixes, define token indicators $g_t \in \{0, 1\}$. Unaligned or boundary-split tokens are rejected, and the retained prefix ends at

$$\kappa = \min\big(\{t : g_t = 0\} \cup \{L + 1\}\big) - 1,$$

where L is the response length. Calls merge only when the reconstructed stream exactly prefixes the next captured canonical prompt. Compaction or context rewrites start a new retained segment.

No projection. The no-projection recipe assigns unit masks to captured response tokens and joins calls through matching canonical prompt prefixes. Under both recipes, context remains visible with zero loss, each segment retains its trajectory's own terminal outcome, and all retained segments share one trajectory-token denominator.

A.3 Optimization details

Grouped sampling. Tri PPO allocates one rollout per task to each of Codex, OpenCode, and Pi; single-harness PPO allocates three to the same harness. Dual PPO allocates one each to Codex and OpenCode. All rollouts begin from the same task state. Training uses complete groups with aligned capture, terminal outcomes, and trainable response tokens for every rollout.

Critic architecture and initialization. The critic is a separate Qwen3.5-27B backbone initialized from the same Base weights as the actor. Its parameters are shared across all trajectories and task groups within a training run, independently of the actor's parameters. A scalar linear head replaces the vocabulary output projection. Its weights are initialized from $\mathcal{N}(0, 0.02^2)$, with zero bias.

Reward and segment-level GAE. Let s index retained segments of trajectory τ, t index response positions, and $u_{\tau,s,t}$ be the trainable-token mask. The trainable length is $N_\tau = \sum_{s,t} u_{\tau,s,t}$. The evaluator's binary outcome R_τ receives a success-gated length penalty based on this total:

$$\widetilde{R}_\tau = R_\tau \left[1 - \alpha_\ell \min\{N_\tau / N_\ell, 1\}\right].$$

Here $\alpha_\ell = 0.05$ is the maximum penalty and $N_\ell = 32{,}768$ is its trainable-token scale. Evaluation uses the binary grader outcome.

Each retained segment forms a training sequence with the same $\widetilde{R}_\tau$ at its last response position and zero reward elsewhere. Write $T_{\tau,s}$ for that position and $V^-_{\tau,s,t} = V_{\phi^-}(\widehat{c}_{\tau,s,t})$ for the stored critic prediction at the reconstructed context. GAE (Schulman et al., 2015) is computed backwards within each segment:

$$\begin{aligned} r^{\mathrm{rew}}_{\tau,s,t} &= \widetilde{R}_\tau \, \mathbf{1}[t = T_{\tau,s}], \\ \delta_{\tau,s,t} &= r^{\mathrm{rew}}_{\tau,s,t} + \gamma V^-_{\tau,s,t+1} - V^-_{\tau,s,t}, \\ A^{\mathrm{raw}}_{\tau,s,t} &= \delta_{\tau,s,t} + \gamma\lambda A^{\mathrm{raw}}_{\tau,s,t+1}, \\ G_{\tau,s,t} &= A^{\mathrm{raw}}_{\tau,s,t} + V^-_{\tau,s,t}. \end{aligned} \tag{7}$$

Both $V^-_{\tau,s,T_{\tau,s}+1}$ and $A^{\mathrm{raw}}_{\tau,s,T_{\tau,s}+1}$ are zero. With $\gamma = \lambda = 1$, this gives $G_{\tau,s,t} = \widetilde{R}_\tau$ and $A^{\mathrm{raw}}_{\tau,s,t} = \widetilde{R}_\tau - V_{\phi^-}(\widehat{c}_{\tau,s,t})$ throughout each segment. The shared critic fits these segment return targets across harnesses.

Normalization and likelihood ratios. The PPO actor standardizes raw advantages over trainable response tokens within each context-parallel partition and data-parallel aggregation group. Let $p(\tau, s, t)$ denote a token's partition. The Claude Code PPO run uses four context-parallel partitions; other PPO runs use two. Pooling tokens across harnesses and trajectories within each group gives mean $\mu_{u,p}$ and Bessel-corrected variance $\sigma^2_{u,p}$, with $\epsilon_{\mathrm{norm}} = 10^{-8}$:

$$A_{\tau,s,t} = \frac{A^{\mathrm{raw}}_{\tau,s,t} - \mu_{u,p(\tau,s,t)}}{\sqrt{\sigma^2_{u,p(\tau,s,t)} + \epsilon_{\mathrm{norm}}}}.$$

Suppressing (τ, s), let β_t be the completion-local log-probability returned by the rollout decoder. Pre-update and current trainer log-probabilities on the reconstructed context are ν_t and $\ell_{\theta,t}$. The capped captured-score weight and PPO ratio are

$$w_t = \min\{\exp(\nu_t - \beta_t), c_w\}, \qquad r_t = \exp(\ell_{\theta,t} - \nu_t). \tag{8}$$

With $c_w = 2$, these weights multiply the clipped PPO surrogate directly. The captured score β_t precedes top-k/top-p truncation.

Reduction and hyperparameters. For B accepted task groups with m slots each, let $\tau_{z,j}$ denote the trajectory in slot j of group z. Both actor and critic normalize each trajectory by its trainable-token count and use equal trajectory weights:

$$\langle f \rangle_{\mathcal{B}} = \frac{1}{Bm} \sum_{z=1}^{B} \sum_{j=1}^{m} \frac{1}{N_{\tau_{z,j}}} \sum_{s,t} u_{\tau_{z,j},s,t} f_{\tau_{z,j},s,t}.$$

Flattening these $M = Bm$ trajectories and their trainable response positions gives Equation 5. Actor and critic learning rates are 10^{-6} and 10^{-5}, respectively. The policy-ratio clipping threshold is $\varepsilon = 0.2$, and the value clipping threshold is also 0.2. Entropy and KL coefficients are zero. The clipped critic prediction is

$$V^{\mathrm{clip}}_\phi(\widehat{c}) = V_{\phi^-}(\widehat{c}) + \mathrm{clip}\big(V_\phi(\widehat{c}) - V_{\phi^-}(\widehat{c}), -0.2, 0.2\big).$$

Each PPO rollout batch receives one optimizer step for the actor and one for the critic. GRPO uses one actor update per batch. The actor maximizes the clipped PPO surrogate (Schulman et al., 2017); the critic minimizes a clipped squared-error loss under the same reduction:

$$\begin{aligned} \mathcal{L}_\pi &= \langle w \min\{rA, \mathrm{clip}(r, 1-\varepsilon, 1+\varepsilon)A\} \rangle_{\mathcal{B}}, \\ \mathcal{L}_V &= \Big\langle \max\{(V_\phi - G)^2, (V^{\mathrm{clip}}_\phi - G)^2\} \Big\rangle_{\mathcal{B}}. \end{aligned} \tag{9}$$

Token indices are suppressed. Here A is the standardized advantage, and r and w are the tokenwise likelihood ratio and capped captured-score weight in Equation 8. The value target $G = \widetilde{R}_\tau$ retains its trajectory's return scale.

A.4 Training configurations

All nine training runs start from the same Base parameters. The eight PPO runs use the same 500 training tasks; GRPO uses a 190-task subset. Table 2 summarizes the training budgets using the checkpoint names defined in Section 4. Dual E1 and E3 are independent Base-initialized runs.

Table 2: **Training configurations.** Nominal trajectories per update are 100 for Dual, 150 for other PPO recipes, and 152 for Tri GRPO.

Checkpoint	Updates	Source tasks
Dual E1	10	500
Dual E3	30	500
Tri PPO	30	500
Tri no-proj.	30	500
Codex	30	500
OpenCode	30	500
Pi	30	500
Claude Code	10	500
Tri GRPO	30	190

B Paired Statistical Inference

Tasks are the inferential units, with $\mathrm{Pass@3}$ evaluation samples aggregated within each task–harness pair. Table 3 reports all 36 Base-relative checkpoint–harness tests. Pointwise 95% percentile intervals use 10,000 paired task-level bootstrap replicates. Exact two-sided McNemar tests compare model-only gains with Base-only losses; Holm correction covers all 36 tests. Eleven comparisons are Holm-significant: seven Codex gains, the Claude Code checkpoint's loss on Codex, and Tri GRPO losses on the other three harnesses.

Training-budget and reconstruction comparisons form a separate eight-test Holm family (Table 4). The independent 30-update Dual E3 run has four-harness mean $\mathrm{Pass@3}$ 1.30 points below the 10-update Dual E1 run (E3 minus E1, 95% CI $[-2.36, -0.27]$). Individual-harness Holm-adjusted p-values range from 0.696 to 1. Tri no-projection minus Tri projection is 0.21 points (95% CI $[-0.79, 1.20]$).

For task i, let $d_{i,h}$ be trained-minus-Base binary $\mathrm{Pass@3}$ success under harness h. In harness order Codex (C), OpenCode (O), Pi (P), and Claude Code (A), define

$$c_i = (d_{i,C} - d_{i,O},\ d_{i,C} - d_{i,P},\ d_{i,C} - d_{i,A})^\top.$$

With sample covariance S, the paired Wald statistic is $731\bar{c}^\top S^+ \bar{c}$, where S^+ is the pseudoinverse. The test uses a chi-square reference distribution with three degrees of freedom and Holm correction over nine checkpoints. Table 5 reports all nine Wald results.

The repository analysis targets the Codex-minus-OpenCode contrast $731^{-1}\sum_i (d_{i,C} - d_{i,O})$. Repository-cluster intervals use 10,000 bootstrap resamples of 11 repositories with replacement, including all tasks for each sampled repository occurrence. Each replicate computes the task-weighted mean contrast. The 2.5th and 97.5th percentiles across replicates give unadjusted pointwise 95% intervals.

C Complete Evaluation Results

C.1 Task portability across all checkpoints

The Codex checkpoint has the largest all-four $\mathrm{Pass@3}$ intersection; Dual E1 has the largest any-harness union (Table 6).

C.2 Harness-specific performance and transfer

The evaluation crosses 731 SWE-Bench Pro tasks, four harnesses, and ten checkpoints under the $\mathrm{Pass@3}$ evaluation budget per cell, totaling 87,720 final sessions. Figure 7 and Table 1 report absolute performance. Table 7 gives Base-relative changes and their averages across training and held-out harnesses.

Table 3: **Base-relative paired inference.** Each row compares 731 paired tasks. Changes and unadjusted pointwise 95% intervals are percentage points; p-values use Holm correction over 36 tests.

Recipe	Harness	Change	95% CI	Gain	Loss	Holm p	Sig.
Dual E1	Codex	$+13.54$	$[+10.94, +16.15]$	106	7	2.78×10^{-22}	yes
Dual E1	OpenCode	-1.23	$[-3.15, +0.68]$	22	31	1	no
Dual E1	Pi	$+0.68$	$[-1.09, +2.46]$	26	21	1	no
Dual E1	Claude Code	$+0.55$	$[-1.23, +2.33]$	24	20	1	no
Dual E3	Codex	$+13.13$	$[+10.40, +15.87]$	108	12	5.7×10^{-19}	yes
Dual E3	OpenCode	-3.15	$[-5.34, -0.96]$	24	47	0.214	no
Dual E3	Pi	-0.68	$[-2.60, +1.23]$	25	30	1	no
Dual E3	Claude Code	-0.96	$[-2.74, +0.82]$	19	26	1	no
Tri PPO	Codex	$+13.27$	$[+10.53, +16.01]$	106	9	1.26×10^{-20}	yes
Tri PPO	OpenCode	-2.60	$[-4.79, -0.55]$	21	40	0.49	no
Tri PPO	Pi	-1.78	$[-3.69, +0.14]$	19	32	1	no
Tri PPO	Claude Code	-2.33	$[-4.24, -0.41]$	17	34	0.53	no
Tri no-proj.	Codex	$+12.45$	$[+9.85, +15.18]$	101	10	1.36×10^{-18}	yes
Tri no-proj.	OpenCode	-2.05	$[-4.38, +0.14]$	27	42	1	no
Tri no-proj.	Pi	-0.55	$[-2.33, +1.23]$	22	26	1	no
Tri no-proj.	Claude Code	-2.46	$[-4.51, -0.41]$	19	37	0.512	no
Codex	Codex	$+15.32$	$[+12.59, +18.06]$	118	6	1.59×10^{-26}	yes
Codex	OpenCode	-0.68	$[-3.15, +1.64]$	36	41	1	no
Codex	Pi	-0.41	$[-2.19, +1.37]$	21	24	1	no
Codex	Claude Code	-0.55	$[-2.46, +1.37]$	22	26	1	no
OpenCode	Codex	$+1.92$	$[-0.27, +4.10]$	43	29	1	no
OpenCode	OpenCode	-2.19	$[-4.24, -0.14]$	21	37	1	no
OpenCode	Pi	-1.92	$[-3.83, +0.00]$	18	32	1	no
OpenCode	Claude Code	-1.92	$[-3.83, -0.14]$	18	32	1	no
Pi	Codex	$+10.26$	$[+7.66, +12.86]$	88	13	2.26×10^{-13}	yes
Pi	OpenCode	-1.37	$[-3.42, +0.68]$	24	34	1	no
Pi	Pi	-1.37	$[-3.28, +0.41]$	19	29	1	no
Pi	Claude Code	-0.68	$[-2.74, +1.37]$	26	31	1	no
Claude Code	Codex	-4.10	$[-6.57, -1.78]$	26	56	0.0317	yes
Claude Code	OpenCode	$+1.09$	$[-0.82, +3.01]$	29	21	1	no
Claude Code	Pi	$+1.37$	$[-0.55, +3.28]$	31	21	1	no
Claude Code	Claude Code	-1.78	$[-3.69, +0.14]$	19	32	1	no
Tri GRPO	Codex	$+12.45$	$[+9.99, +15.05]$	98	7	3.98×10^{-20}	yes
Tri GRPO	OpenCode	-6.02	$[-8.62, -3.56]$	22	66	8.35×10^{-5}	yes
Tri GRPO	Pi	-5.06	$[-7.39, -2.87]$	16	53	0.000255	yes
Tri GRPO	Claude Code	-3.69	$[-5.75, -1.50]$	18	45	0.0242	yes

Single-harness training produces held-out gains (Figure 2): the Pi checkpoint improves Codex by 10.26 points while changing Pi by -1.37 points. Figures 8 and 9 show training-versus-held-out averages and all Claude Code comparisons.

C.3 Sampling budget and paired turnover

For each checkpoint–harness pair, let c_z count successes among $n = 3$ attempts on task z. We use the $\mathrm{Pass@}k$ estimator of Chen et al. (2021):

$$\widehat{\mathrm{Pass@}k} = \frac{1}{731} \sum_{z} \left[1 - \frac{\binom{3-c_z}{k}}{\binom{3}{k}} \right]. \tag{10}$$

Equation 10 evaluates $\mathrm{Pass@}1$, $\mathrm{Pass@}2$, and $\mathrm{Pass@}3$ from the same $\mathrm{Pass@}3$ evaluation samples. Table 8 averages these estimates over 2,924 task–harness units for each of the ten checkpoints.

Table 4: **Direct recipe comparisons.** Duration compares independently Base-initialized Dual E1 (10 updates, left) and Dual E3 (30 updates, right). Projection compares Tri projection (left) with Tri no-projection (right). Differences are right minus left in percentage points over 731 paired tasks. Intervals are pointwise and unadjusted; p-values use the separate eight-test Holm family. Bootstrap intervals summarize four-harness means.

Comparison	harness	Left	Right	Difference	95% CI	Holm p
Duration	Codex	27.77	27.36	-0.41	$[-2.46, +1.64]$	1
Duration	OpenCode	27.09	25.17	-1.92	$[-3.97, +0.14]$	0.696
Duration	Pi	31.33	29.96	-1.37	$[-3.28, +0.55]$	1
Duration	Claude Code	31.87	30.37	-1.50	$[-3.42, +0.27]$	1
Duration	All Harnesses	29.51	28.21	-1.30	$[-2.36, -0.27]$	–
Projection	Codex	27.50	26.68	-0.82	$[-2.87, +1.23]$	1
Projection	OpenCode	25.72	26.27	$+0.55$	$[-1.64, +2.87]$	1
Projection	Pi	28.86	30.10	$+1.23$	$[-0.68, +3.15]$	1
Projection	Claude Code	29.00	28.86	-0.14	$[-1.92, +1.64]$	1
Projection	All Harnesses	27.77	27.98	$+0.21$	$[-0.79, +1.20]$	–

Table 5: **Harness heterogeneity and repository-cluster contrast.** Wald tests have three degrees of freedom and a separate nine-test Holm family. The contrast is Codex-minus-OpenCode Base-relative Pass@3 change, in percentage points; repository intervals are unadjusted and pointwise.

Checkpoint	Wald	Holm p	Contrast	Repository 95% CI
Dual E1	86.10	7.52×10^{-18}	$+14.77$	$[+6.68, +22.47]$
Dual E3	87.74	4.03×10^{-18}	$+16.28$	$[+7.24, +24.49]$
Tri PPO	109.25	1.27×10^{-22}	$+15.87$	$[+6.38, +24.59]$
Tri no-proj.	84.01	1.69×10^{-17}	$+14.50$	$[+4.87, +23.55]$
Codex	107.62	2.50×10^{-22}	$+16.01$	$[+5.71, +25.83]$
OpenCode	8.78	0.0324	$+4.10$	$[+0.60, +7.79]$
Pi	60.69	1.26×10^{-12}	$+11.63$	$[+4.48, +18.66]$
Claude Code	16.45	0.0018	-5.20	$[-9.23, -1.70]$
Tri GRPO	117.91	1.96×10^{-24}	$+18.47$	$[+7.22, +28.87]$

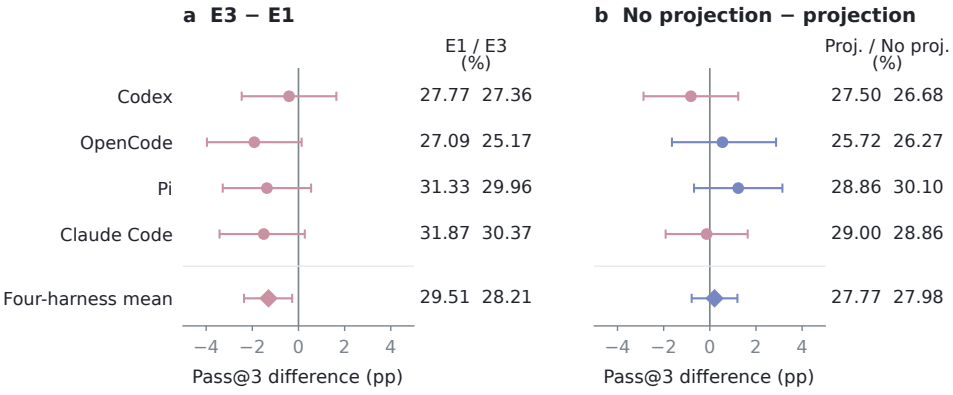


Figure 6: **Training-budget and reconstruction comparisons.** Dual E3 has lower Pass@3 than Dual E1 under each harness. Projection changes have mixed signs. Points show paired Pass@3 differences in percentage points; error bars are pointwise 95% paired-task bootstrap intervals and diamonds show four-harness means. Table 4 reports the paired inference.

Table 6: **Task portability for every checkpoint.** Minimum and range refer to harness-specific Pass@3, in percent and percentage points. All four and any harness count tasks in the intersection and union. Overlap is the fraction of newly solved Codex tasks already solved by Base under another harness.

Checkpoint	Minimum	Range	All four	Any harness	Overlap
Base	14.23	17.10	82	266	—
Dual E1	27.09	4.79	148	273	99/106
Dual E3	25.17	5.20	139	262	103/108
Tri PPO	25.72	3.28	146	256	103/106
Tri no-proj.	26.27	3.83	134	262	97/101
Codex	27.63	3.15	159	262	113/118
OpenCode	16.14	13.27	90	256	42/43
Pi	24.49	6.16	133	266	85/88
Claude Code	10.12	21.89	64	261	25/26
Tri GRPO	22.30	5.34	131	247	95/98

Pass@3 (%) 0 20 40 60

	Codex	OpenCode	Pi	Claude Code
Base	14.2	28.3	30.6	31.3
Dual E1	27.8	27.1	31.3	31.9
Dual E3	27.4	25.2	30.0	30.4
Tri PPO	27.5	25.7	28.9	29.0
Tri no-proj.	26.7	26.3	30.1	28.9
Codex	29.5	27.6	30.2	30.8
OpenCode	16.1	26.1	28.7	29.4
Pi	24.5	26.9	29.3	30.6
Claude Code	10.1	29.4	32.0	29.5
Tri GRPO	26.7	22.3	25.6	27.6

Figure 7: **Absolute** Pass@3 **on SWE-Bench Pro.** Cells give the percentage of 731 tasks solved under the Pass@3 evaluation budget for each checkpoint–harness pair. Shading uses the same 0–60% scale as Figure 13.

Table 7: **Base-relative** Pass@3 **changes.** (C, O, P, A) denote Codex, OpenCode, Pi, and Claude Code. The final two columns average changes over training and held-out harnesses, respectively. Each change compares 731 paired tasks; values are percentage points.

Recipe	$\widehat{D}_C$	$\widehat{D}_O$	$\widehat{D}_P$	$\widehat{D}_A$	Training	Held-out
Dual E1	+13.54	−1.23	+0.68	+0.55	+6.16	+0.62
Dual E3	+13.13	−3.15	−0.68	−0.96	+4.99	−0.82
Tri PPO	+13.27	−2.60	−1.78	−2.33	+2.96	−2.33
Tri no-proj.	+12.45	−2.05	−0.55	−2.46	+3.28	−2.46
Codex	+15.32	−0.68	−0.41	−0.55	+15.32	−0.55
OpenCode	+1.92	−2.19	−1.92	−1.92	−2.19	−0.64
Pi	+10.26	−1.37	−1.37	−0.68	−1.37	+2.74
Claude Code	−4.10	+1.09	+1.37	−1.78	−1.78	−0.55
Tri GRPO	+12.45	−6.02	−5.06	−3.69	+0.46	−3.69

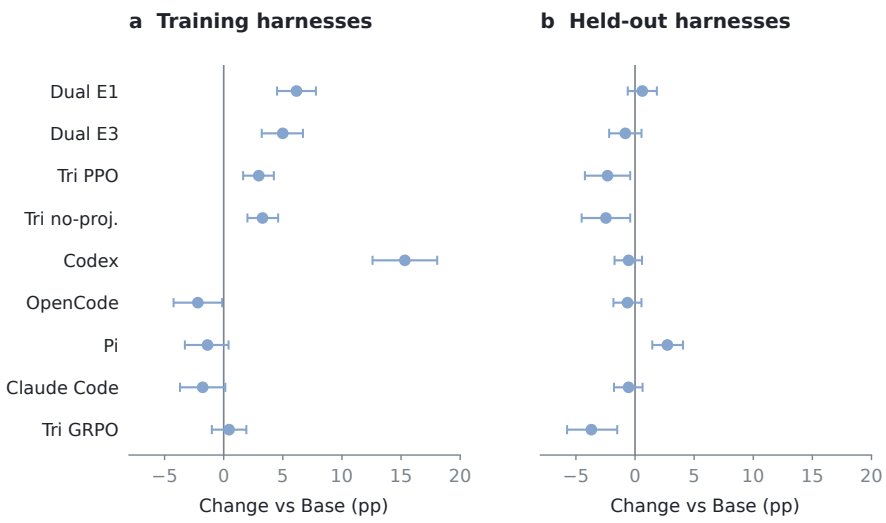


Figure 8: **Changes on training and held-out harnesses.** Points equally average each recipe's Pass@3 changes on training or held-out harnesses over 731 paired tasks. Error bars are pointwise 95% paired-task bootstrap intervals.

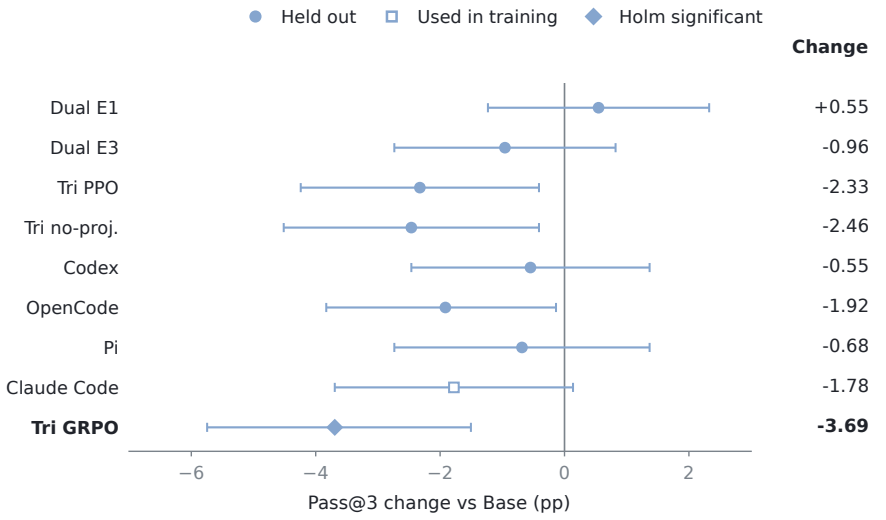


Figure 9: **Base-relative** Pass@3 **under Claude Code.** Each comparison uses 731 paired tasks; error bars are pointwise 95% paired-task bootstrap intervals. The open square marks the Claude Code-trained checkpoint; the remaining checkpoints show held-out transfer. The diamond marks the change significant after Holm correction across all 36 checkpoint–harness comparisons.

Figure 10 counts newly solved and lost units among 2,924 task–harness pairs. The all-four intersection counts tasks solved by every harness under the Pass@3 evaluation budget; the any-harness union counts tasks solved by at least one harness.

Table 8: **SWE-Bench Pro performance by sampling budget.** The estimator in Equation 10 uses the Pass@3 evaluation samples in each of 2,924 task–harness units. Values are percentages.

Checkpoint	Pass@1	Pass@2	Pass@3
Base	19.66	23.94	26.13
Dual E1	21.93	26.99	29.51
Dual E3	21.07	25.85	28.21
Tri PPO	21.19	25.70	27.77
Tri no-proj.	21.06	25.72	27.98
Codex	23.24	27.42	29.55
OpenCode	18.87	22.94	25.10
Pi	20.44	25.26	27.84
Claude Code	18.78	23.03	25.27
Tri GRPO	19.07	23.35	25.55

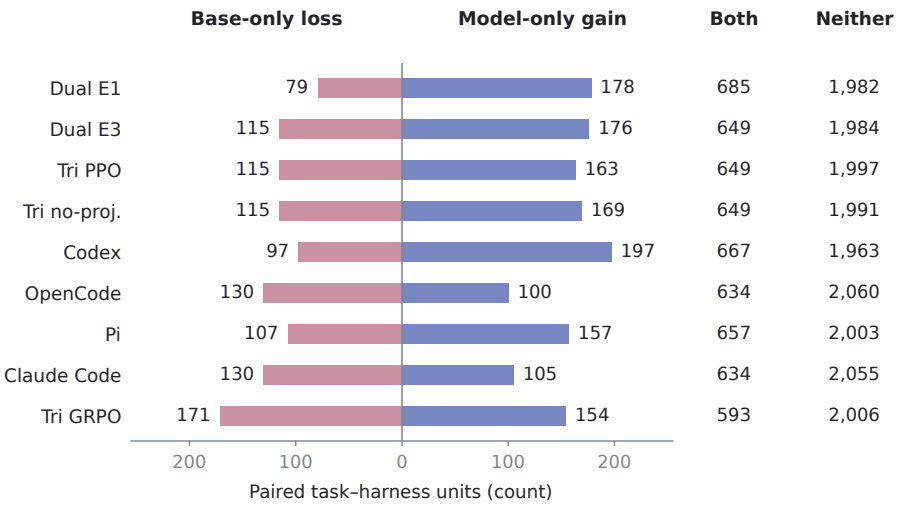


Figure 10: **Paired task–harness transitions relative to Base.** Each checkpoint compares 2,924 units. Bars show losses and gains from a common zero; numeric columns retain units solved by both or neither checkpoint.

D Execution Stages and Runtime Outcomes

D.1 SWE-Bench Pro execution-stage analysis

Among 87,720 final sessions, 82,594 complete, 76,230 of these deliver a nonempty patch, and 18,006 of those pass grading. Completion, delivery, and conditional grader-success rates respectively use all sessions, completed sessions, and completed sessions with nonempty patches as denominators.

Completion follows the session's final status; patch delivery follows the grader's patch classification. Pass metrics use grader outcomes for all sessions.

D.2 SWE-Bench Pro session reliability

Each checkpoint–harness pair contributes 2,193 final sessions to the runtime denominator (Table 9). Figure 11 reports runtime failure and grader success among completed sessions.

Table 9: **Final session outcomes.** Each checkpoint–harness row contains 2,193 sessions (731 tasks under the Pass@3 evaluation budget). COMPLETED, ERROR, TIMEOUT, and success conditioned on COMPLETED are percentages.

Checkpoint	Harness	Completed	Error	Timeout	Success | Completed
Base	Codex	98.81	1.19	0.00	7.25
Base	OpenCode	90.10	3.51	6.38	24.34
Base	Pi	93.66	2.51	3.83	26.29
Base	Claude Code	97.40	0.91	1.69	25.56
Dual E1	Codex	97.22	2.19	0.59	19.23
Dual E1	OpenCode	84.50	3.88	11.63	24.55
Dual E1	Pi	92.66	2.96	4.38	25.94
Dual E1	Claude Code	97.17	0.87	1.96	24.92
Dual E3	Codex	95.39	3.88	0.73	21.27
Dual E3	OpenCode	82.63	6.20	11.17	22.13
Dual E3	Pi	92.89	5.70	1.41	25.18
Dual E3	Claude Code	97.63	0.55	1.82	22.79
Tri PPO	Codex	96.81	2.28	0.91	21.53
Tri PPO	OpenCode	84.04	3.56	12.40	23.22
Tri PPO	Pi	95.26	3.24	1.50	23.55
Tri PPO	Claude Code	98.31	0.78	0.91	22.36
Tri no-proj.	Codex	97.04	1.60	1.37	19.83
Tri no-proj.	OpenCode	86.50	2.05	11.45	21.72
Tri no-proj.	Pi	96.81	1.46	1.73	24.73
Tri no-proj.	Claude Code	97.81	0.87	1.32	22.75
Codex	Codex	94.98	2.33	2.69	25.20
Codex	OpenCode	84.22	2.46	13.32	24.53
Codex	Pi	93.94	2.42	3.65	25.24
Codex	Claude Code	97.45	1.00	1.55	25.27
OpenCode	Codex	98.91	0.91	0.18	9.96
OpenCode	OpenCode	89.65	1.60	8.76	22.74
OpenCode	Pi	96.17	1.55	2.28	24.09
OpenCode	Claude Code	98.50	0.87	0.64	22.41
Pi	Codex	96.49	3.42	0.09	15.69
Pi	OpenCode	86.46	6.25	7.30	23.68
Pi	Pi	93.75	5.11	1.14	23.98
Pi	Claude Code	97.45	1.00	1.55	24.29
Claude Code	Codex	99.77	0.18	0.05	5.03
Claude Code	OpenCode	94.48	2.37	3.15	24.18
Claude Code	Pi	96.72	1.37	1.92	25.41
Claude Code	Claude Code	98.08	1.05	0.87	23.11
Tri GRPO	Codex	95.99	3.56	0.46	21.09
Tri GRPO	OpenCode	87.05	5.65	7.30	18.65
Tri GRPO	Pi	94.57	5.20	0.23	20.49
Tri GRPO	Claude Code	99.00	0.82	0.18	20.64

Figure 12 compares performance with the average of harness-specific median session durations. Tri GRPO has the shortest duration; Dual E1 and Codex have the highest four-harness mean Pass@3.

E TASK STRATA

The difficulty proxy combines five reference-patch and issue properties within each of 11 repositories. Inputs comprise patch line changes, patch file counts, test-patch line changes, fail-to-pass test counts, and combined problem-statement and requirements length. Within-repository percentile ranks use average ranks for ties and respective weights of 0.45, 0.20, 0.15, 0.10, and 0.10. Weighted scores are ranked within each repository. Percentiles at or below 0.40 define Easy, those above 0.85 define Hard, and the remainder define Medium. The resulting Easy, Medium, and Hard strata contain 288, 329, and 114 tasks, respectively.

a Runtime failure

Rate (%) 0 10 20

	Codex	OpenCode	Pi	Claude Code
Base	1.2	9.9	6.3	2.6
Dual E1	2.8	15.5	7.3	2.8
Dual E3	4.6	17.4	7.1	2.4
Tri PPO	3.2	16.0	4.7	1.7
Tri no-proj.	3.0	13.5	3.2	2.2
Codex	5.0	15.8	6.1	2.6
OpenCode	1.1	10.4	3.8	1.5
Pi	3.5	13.5	6.2	2.6
Claude Code	0.2	5.5	3.3	1.9
Tri GRPO	4.0	13.0	5.4	1.0

b Success | completed

Rate (%) 0 20 40

	Codex	OpenCode	Pi	Claude Code
Base	7.2	24.3	26.3	25.6
Dual E1	19.2	24.6	25.9	24.9
Dual E3	21.3	22.1	25.2	22.8
Tri PPO	21.5	23.2	23.6	22.4
Tri no-proj.	19.8	21.7	24.7	22.8
Codex	25.2	24.5	25.2	25.3
OpenCode	10.0	22.7	24.1	22.4
Pi	15.7	23.7	24.0	24.3
Claude Code	5.0	24.2	25.4	23.1
Tri GRPO	21.1	18.6	20.5	20.6

Figure 11: **Runtime reliability and conditional grader success.** Each checkpoint–harness cell starts from 2,193 sessions. Runtime failure is final ERROR plus TIMEOUT; conditional grader success is the success rate among COMPLETED sessions. Panels share checkpoint rows and use separate color scales (0–20% and 0–40%).

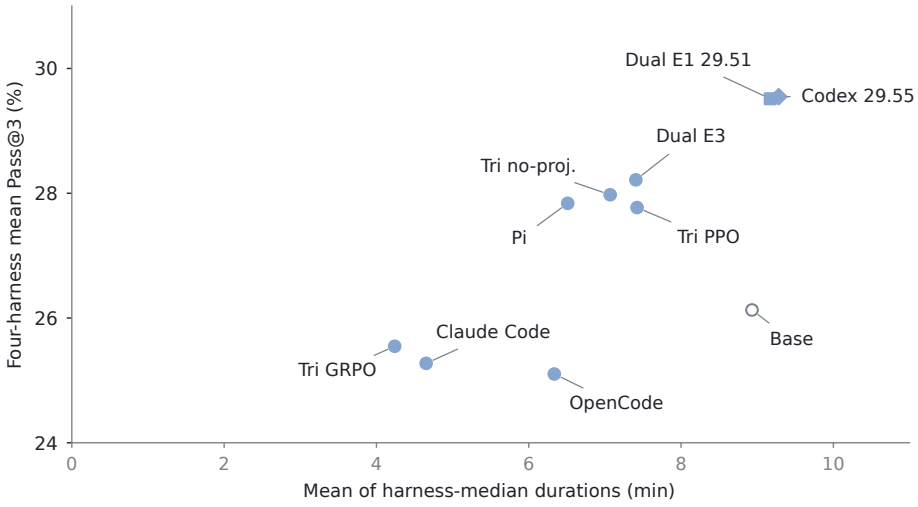


Figure 12: **Performance and session duration.** Each point identifies one checkpoint. The horizontal coordinate averages the four harness-specific median session durations; the vertical coordinate equally averages harness-specific Pass@3.

The language strata contain 280 Go, 266 Python, 165 JavaScript, and 20 TypeScript tasks. All 300 Go and TypeScript tasks have zero successes across ten checkpoints and four harnesses under the Pass@3 evaluation budget.

F Supplementary SWE-bench Verified Evaluation

This evaluation used 45 tasks from the 500-task SWE-bench Verified dataset, selected for estimated experienced-developer repair effort exceeding one hour (OpenAI, 2024). Table 10 reports the results.

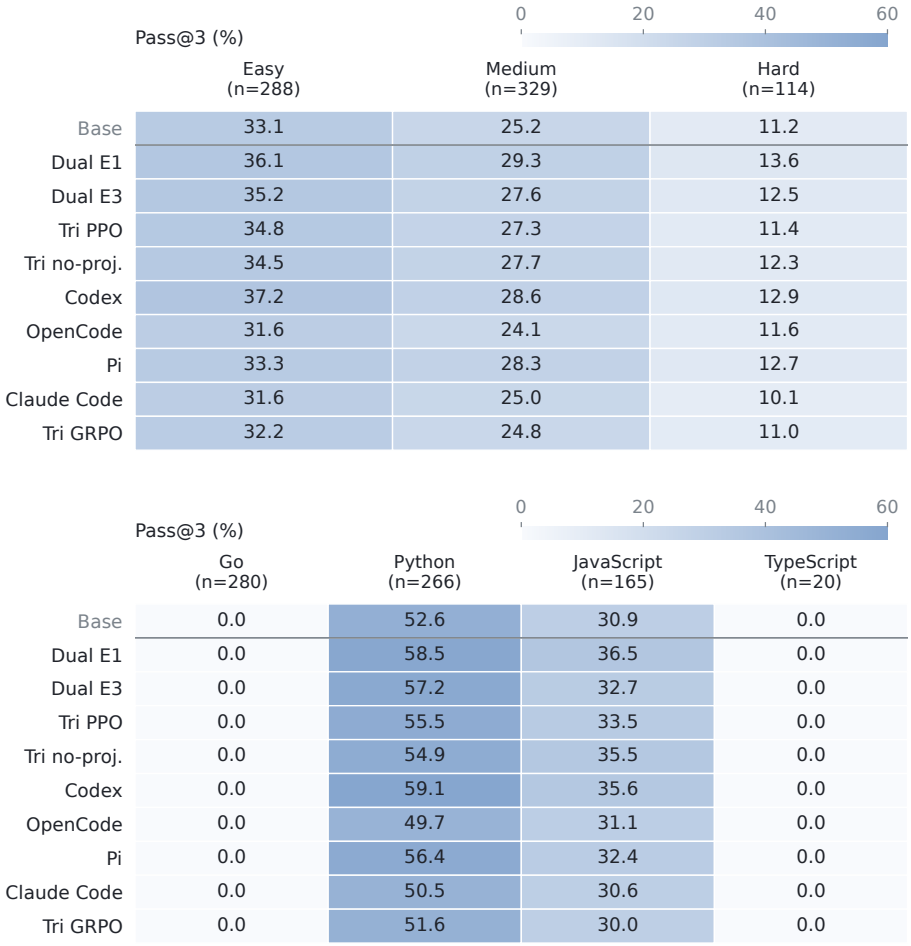


Figure 13: **Task-stratum results.** Top: difficulty proxy. Bottom: repository language. Each cell averages task-level Pass@3 over four harnesses within the indicated stratum. Axes show task counts. Both panels use the same 0–60% color scale.

Table 10: **Supplementary results on 45 SWE-bench Verified tasks.** Each checkpoint–harness pair uses the same 45 tasks and three evaluation runs. Pass@1 is the per-attempt success rate; Pass@3 is the fraction of tasks solved at least once. Four-harness means give each harness equal weight. Values are percentages.

Checkpoint	Codex		OpenCode		Pi		Claude Code		Four-harness mean	
	Pass@1	Pass@3	Pass@1	Pass@3	Pass@1	Pass@3	Pass@1	Pass@3	Pass@1	Pass@3
Base	24.44	35.56	29.63	42.22	31.11	44.44	29.63	44.44	28.70	41.67
Dual E1	31.11	44.44	27.41	35.56	28.15	40.00	30.37	46.67	29.26	41.67
Dual E3	21.48	31.11	27.41	42.22	22.96	35.56	28.15	42.22	25.00	37.78
Tri PPO	22.22	31.11	25.19	31.11	24.44	33.33	25.93	37.78	24.44	33.33
Tri no-proj.	26.67	40.00	24.44	35.56	28.89	42.22	23.70	37.78	25.93	38.89
Codex	22.96	35.56	28.15	40.00	28.15	40.00	28.15	37.78	26.85	38.33
OpenCode	27.41	40.00	27.41	42.22	25.19	33.33	26.67	40.00	26.67	38.89
Pi	26.67	42.22	29.63	44.44	23.70	31.11	31.11	44.44	27.78	40.56
Claude Code	28.89	42.22	25.19	37.78	25.93	33.33	21.48	31.11	25.37	36.11
Tri GRPO	18.52	26.67	22.22	28.89	27.41	42.22	20.00	26.67	22.04	31.11

The Pi checkpoint increased Codex Pass@3 from 16/45 to 19/45 tasks, while its Pi Pass@3 fell from 20/45 to 14/45. Dual E1 had the highest trained four-harness mean Pass@3, matching Base at 41.67%; all eight other trained checkpoints were lower. Every trained checkpoint's worst-harness Pass@3 remained at or below Base's 35.56%.

The Codex checkpoint matched Base's Codex Pass@3 at 35.56%, while Codex Pass@1 fell from 24.44% to 22.96%. Both metrics were lower under the other three harnesses.