

PivotRL: High Accuracy Agentic Post-Training at Low Compute Cost

Junkeun Yi^{*,1} Damon Mosk-Aoyama^{*,1} Baihe Huang^{*,2} Ritu Gala¹ Charles Wang¹
 Sugam Dipak Devare¹ Khushi Bhardwaj¹ Abhibha Gupta¹ Oleksii Kuchaiev¹
 Jiantao Jiao^{**,1,2} Jian Zhang^{**,1} Venkat Srinivasan^{**,†,1}

¹NVIDIA ²UC Berkeley

*Equal Contribution **Equal Advising [†]Corresponding Author: venkats@nvidia.com

Abstract

Post-training for long-horizon agentic tasks has a tension between compute efficiency and generalization. While supervised fine-tuning (SFT) is compute efficient, it often suffers from out-of-domain (OOD) degradation. Conversely, end-to-end reinforcement learning (E2E RL) preserves OOD capabilities, but incurs high compute costs due to many turns of on-policy rollout. We introduce **PivotRL**, a novel framework that operates on existing SFT trajectories to combine the compute efficiency of SFT with the OOD accuracy of E2E RL. PivotRL relies on two key mechanisms: first, it executes local, on-policy rollouts and filters for *pivots*: informative intermediate turns where sampled actions exhibit high variance in outcomes; second, it utilizes rewards for functional-equivalent actions rather than demanding strict string matching with the SFT data demonstration. We theoretically show that these mechanisms incentivize strong learning signals with high natural gradient norm, while preserving policy probability ordering on actions unrelated to training tasks. In comparison to standard SFT on identical data, we demonstrate that PivotRL achieves +4.17% higher in-domain accuracy on average across four agentic domains, and +10.04% higher OOD accuracy in non-agentic tasks. Notably, on agentic coding tasks, PivotRL achieves competitive accuracy with E2E RL with $4\times$ fewer training-time rollout turns.

1 Introduction

Long-horizon agentic tasks require many turns of Large Language Model (LLM) interaction with an environment. This includes tasks such as conversational tool use (Schick et al., 2023; Qin et al., 2023), agentic coding (Jimenez et al., 2024), terminal interaction (Xie et al., 2024), and web search (Yao et al., 2022; Wei et al., 2025). These tasks have emerged as the frontier in AI systems capable of executing complex, real-world workflows. However, post-training for these long-horizon agentic capabilities introduces a fundamental tension between training efficiency and generalization. While supervised fine-tuning (SFT) offers a compute-efficient mechanism for acquiring new capabilities, it frequently struggles to generalize beyond the training distribution and catastrophically degrades out-of-domain (OOD) performance (Chu et al., 2025; Luo et al., 2025b). In contrast, end-to-end reinforcement learning (E2E RL) typically yields higher in-domain accuracy while robustly retaining OOD capabilities (Ouyang et al., 2022; Chen et al., 2025). Nevertheless, E2E RL incurs high compute overheads, as each parameter update necessitates repeated, many turns of on-policy rollout with environmental interactions. This dichotomy naturally raises the following question:

Can we combine the data efficiency of SFT with the generalization capabilities of E2E RL, achieving both in-domain accuracy and OOD retention without incurring full-trajectory rollouts?

To address this challenge, a natural attempt would be to repurpose SFT trajectory demonstrations for RL. Specifically, one could sample on-policy rollouts conditioned on a randomly selected intermediate turn from an SFT trajectory, assigning a positive reward only if the generated action exactly matches the SFT data demonstration. While this mitigates the under-utilization of SFT data which only trains on a single demonstrated completion, our preliminary experiments reveal that it fails to improve OOD accuracy relative to standard SFT on the same data. We empirically trace this failure to two bottlenecks. First, randomly selected intermediate turns frequently provide negligible learning signals; under Group Relative Policy Optimization (GRPO) (Shao et al., 2024), sampled actions at such turns often uniformly succeed or fail, yielding a normalized advantage close to zero and thus producing no meaningful gradient update. Second, exact string matching with SFT data is excessively restrictive in generative action spaces, where numerous functionally equivalent actions validly diverge from the singular demonstrated completion. For instance, a tool call or search query may be perfectly appropriate despite not perfectly matching the demonstration.

Motivated by these insights, we introduce **PivotRL**, a novel framework for long-horizon agentic RL to overcome both bottlenecks simultaneously. PivotRL executes local, on-policy rollouts and filters for *pivots*: informative assistant turns where sampled actions exhibit mixed outcomes (i.e., both successes and failures). Consequently, training requires only brief, partial rollouts from these pivot states rather than exhaustive full trajectories. To effectively score these rollouts, we leverage domain-appropriate verifiers to assign rewards for functionally equivalent actions rather than strictly penalizing deviations from SFT data demonstrations (Shao et al., 2024; Rohatgi et al., 2025).

We substantiate our methodology with a lightweight theoretical analysis on our core design choices. First, we prove that the Fisher norm of the natural gradient of the statewise reward objective scales with the reward standard deviation. Consequently, the GRPO update along the KL path scales directly with this standard deviation, validating our strategy of filtering for reward-diverse pivots to maximize the local in-domain learning signals. Second, we show that functional reward-based RL shifts probability mass toward actions that are functionally equivalent to the expert demonstration, while preserving the conditional distributions on all other actions. This maintains the reference policy’s relative ordering of task-unrelated actions, thereby mitigating out-of-domain (OOD) degradation.

Empirically, we first show that under identical training data—the same prompts and expert trajectories—PivotRL yields larger in-domain improvements than SFT while exhibiting substantially less OOD regression. When training in the conversational tool use, agentic coding, terminal tool use and search domain, PivotRL achieves an average in-domain accuracy improvement of +14.11¹ over the base model, which significantly improves the +9.94 points attained by SFT. Crucially, PivotRL does not introduce OOD regressions (+0.21 change), whereas SFT suffers a severe regression of −9.48 in non-agentic domains including math, science QA and competitive coding. Second, we provide a direct comparison with E2E RL on SWE-Bench, a rigorous standard for evaluating long-horizon agentic capabilities with well-established E2E RL baselines (Jimenez et al., 2024; Wang et al., 2025b). On this benchmark, we demonstrate that PivotRL attains competitive accuracy with E2E RL, but only with a 4× fewer training-time rollout turns.

2 Preliminaries and motivating observations

Turn-level agentic training. Let τ be an interaction trajectory collected in agentic tasks. Decomposing τ at assistant decision boundaries gives $\tau = (s_0, a_0^*, \dots, s_T, a_T^*)$, where s_t is the full interaction history from the beginning of trajectory τ_i up to, but not including, the t -th assistant action, and a_t^* is the demonstrated assistant completion at that state. In turn-level agentic training, an *action* is the full assistant completion at a model-call boundary, instead of an individual token. The standard negative log-likelihood loss for SFT is given by

$$\mathcal{L}_{\text{sft}}(\theta) = -\mathbb{E}_{\tau \sim \mathcal{D}_{\text{sft}}(s_t, a_t^*) \sim \tau} [\log \pi_\theta(a_t^* | s_t)].$$

¹Throughout this paper, performance scores and differences are reported in percentage points.

Let π_0 denote the reference policy, E2E RL samples rollouts from π_{old} and optimizes the GRPO (Shao et al., 2024) objective given by:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{\substack{s \sim \mathcal{D}_{\pi_{\text{old}}} \\ a_t \sim \pi_{\text{old}}(\cdot | s)}} \left[\frac{1}{G} \sum_{i=1}^G \min(w_i(\theta) \hat{A}_i, \text{clip}(w_i(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_i) - D_{\text{KL}} \right]$$

where $D_{\text{KL}} = \beta \text{KL}(\pi_{\theta}(\cdot | s) \parallel \pi_0(\cdot | s))$ is the KL regularization, $w_i(\theta) = \frac{\pi_{\theta}(a_i | s)}{\pi_{\text{old}}(a_i | s)}$ is the importance sampling weight, $\mathcal{D}_{\pi_{\text{old}}}$ denotes the distribution of states s_t from the on-policy rollout trajectories, and

$$\hat{A}_i = \frac{r_i - \frac{1}{G} \sum_{j=1}^G r_j}{\text{std}(\{r_j\}_{j=1}^G) + \epsilon_{\text{std}}}. \quad (1)$$

represents the advantage normalized across the group of G sampled end-to-end rollouts.

Local RL from expert trajectories. To circumvent the computational overhead associated with full end-to-end trajectory generation, we consider a local RL paradigm. In this setting, instead of unrolling full interactions from the initial environment state until termination, we condition the policy on intermediate expert states s_t derived directly from the SFT dataset and then conduct targeted, turn-level rollouts from these states. Given expert trajectory dataset $\mathcal{D}_{\text{sft}} = \{\tau_i\}_{i=1}^N$, the simplest attempt to adapt it for local RL goes as follows: first sample a state s_t , then sample actions from $\pi_{\theta}(\cdot | s_t)$ and reward the action only when it exactly matches the demonstrated continuation:

$$r_{\text{strict}}(s, a) = \mathbf{1}[a = a^*(s)]. \quad (2)$$

This formulation represents the most direct translation of SFT demonstrations into a local RL framework: the interaction history is rigidly anchored to the expert trace, the subsequent action is sampled on-policy, and credit is sparsely assigned only for perfect replication of the demonstrated completion. However, empirical evaluations in τ^2 -Bench reveal that this naive local RL strategy yields merely marginal gains over standard behavior cloning on identical data, achieving an accuracy of 57.34 compared to 58.44 for same-data SFT. Consequently, this simplistic SFT-to-RL conversion proves inadequate for driving meaningful performance improvements.

Motivating observations. Through our preliminary experiments, we identify two bottlenecks in allocating rollout budgets and assigning local credit. *First, turns with uniformly successful or failed actions are uninformative under group-normalized RL.* Indeed, if a batch of rewards $\{r_i\}_{i=1}^G$ consists entirely of zeros or ones, then the normalized advantage in Eq. (1) evaluates to zero. Empirically on τ^2 -bench and SWE-Bench, 71% of randomly sampled turns yield a 0 learning signal, meaning they are uniformly solved or uniformly failed and contribute nothing to the gradient. *Second, exact-match local credit is too strict.* In generative action spaces, many tool calls, shell commands, or search steps are locally acceptable without matching the single demonstrated string exactly. These two bottlenecks map directly to the two ingredients of PivotRL: we first filter for *pivots* (informative turns that continue to produce mixed outcomes), and then replace exact-match local credit with a verifier that rewards locally acceptable actions.

3 PivotRL

PivotRL modifies the naive local-RL baseline from Section 2 in exactly two ways: it filters extracted turns so that online rollout budget is spent on informative states, and it replaces exact-match local credit with verifier-based reward. We now introduce the full training pipeline and then give two theoretical results that help explain these choices. Section 3.1 presents the method. Section 3.2 then studies the two components separately: Proposition 3.1 and Theorem 3.2 explain why turns with very different responses provide stronger local learning signal under group-normalized RL, while Theorem 3.3 shows that the verifier-based reward shifts probability mass toward acceptable actions while remaining conservative relative to the reference policy.

3.1 Method

In turn-level training, we extract assistant turns from each trajectory $\tau = (s_0, a_0^*, \dots, s_T, a_T^*)$ into a *pivot candidate* dataset:

$$\mathcal{D}_{\text{cand}} = \{(s_t, a_t^*)\}_{\tau \in \mathcal{D}_{\text{SFT}}, t=0, \dots, T}. \quad (3)$$

PivotRL performs three steps: (i) profile turns offline and retain only those likely to remain informative, (ii) sample local on-policy rollouts at the retained turns, and (iii) optimize a verifier-based GRPO-style objective. We summarize the full procedure in Algorithm 1. The first step addresses the uninformative-turn bottleneck from Section 2; the second addresses the overly strict local-credit bottleneck.

Offline turn selection. We estimate the informativeness of each extracted turn under a frozen reference policy π_0 , typically the policy used to initialize PivotRL. For a turn state s , we sample K local rollouts from $\pi_0(\cdot | s)$, score them with the verifier, and compute

$$\hat{\mu}(s) = \frac{1}{K} \sum_{k=1}^K r_{\text{func}}(s, a^{(k)}), \quad \hat{\sigma}^2(s) = \frac{1}{K} \sum_{k=1}^K (r_{\text{func}}(s, a^{(k)}) - \hat{\mu}(s))^2. \quad (4)$$

We then keep only turns with nonzero empirical reward variance and low reward mean,

$$\mathcal{D}_{\text{adv}} = \{(s, a^*) \in \mathcal{D}_{\text{cand}} : \hat{\sigma}^2(s) > 0, \hat{\mu}(s) < \lambda_{\text{diff}}\}. \quad (5)$$

The first filter removes turns that are already uniformly solved or uniformly failed under the reference policy; the second concentrates training on mixed-outcome turns that are still difficult. This filtered subset is thus called *pivot* used for PivotRL training. We write $\mathcal{D}_{\text{pivot}}$ for the retained training set, and unless otherwise stated use $\mathcal{D}_{\text{pivot}} = \mathcal{D}_{\text{adv}}$.

Verifier-based local reward. For a retained state s (present in $\mathcal{D}_{\text{adv}}$), let $\mathcal{M}(s) \subseteq \mathcal{A}(s)$ denote the set of locally acceptable actions under a domain-specific verifier. PivotRL assigns reward

$$r_{\text{func}}(s, a) = \mathbf{1}[a \in \mathcal{M}(s)]. \quad (6)$$

Relative to the strict local reward in Eq. (2), this verifier credits any action that is acceptable at the current turn, not only the single demonstrated completion. Depending on the domain, the verifier may be a normalized string/schema check, a task-specific equivalence rule, or a lightweight LLM judge.

Given a retained turn set $\mathcal{D}_{\text{pivot}}$ and rollout group size G , PivotRL samples $\{a_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}(\cdot | s)$ at each selected state and optimizes

$$\mathcal{J}_{\text{PivotRL}}(\theta) = \mathbb{E}_{\substack{s \sim \mathcal{D}_{\text{pivot}} \\ \{a_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}(\cdot | s)}} \left[\frac{1}{G} \sum_{i=1}^G \min(w_i(\theta) \hat{A}_i, \text{clip}(w_i(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_i) - D_{\text{KL}} \right], \quad (7)$$

where $D_{\text{KL}} = \beta \text{KL}(\pi_{\theta}(\cdot | s) \parallel \pi_0(\cdot | s))$, $w_i(\theta) = \frac{\pi_{\theta}(a_i | s)}{\pi_{\theta_{\text{old}}}(a_i | s)}$ and $\hat{A}_i$ is the group-normalized advantage from Eq. (1), computed using the local verifier rewards $\{r_{\text{func}}(s, a_i)\}_{i=1}^G$. Relative to end-to-end RL, the only online interaction during training is the short rollout needed to score each sampled turn-level action.

3.2 Theoretical analysis for PivotRL

We next analyze the two design choices from Section 3.1. We first formalize why mixed-outcome turns are the right states for group-normalized local RL. We then show that verifier-based reward yields a conservative KL-regularized update: it increases total mass on locally acceptable actions while preserving the reference ordering within and outside that acceptable set. Throughout this subsection, we assume finite action spaces, $\mathcal{M}(s) \neq \emptyset$, and $\pi_0(a | s) > 0$ for all s and a .

Algorithm 1 PivotRL

```

1: Require: expert trajectories  $\mathcal{D}_{\text{sft}}$ , reference policy  $\pi_0$ , initial policy  $\pi_\theta$ , verifier  $V$ , rollout
   group size  $G$ , difficulty threshold  $\lambda_{\text{diff}}$ 
2: Extract turns  $\mathcal{D}_{\text{cand}} = \{(s_t, a_t^*)\}$  from  $\mathcal{D}_{\text{sft}}$ 
3: Profile each  $s \in \mathcal{D}_{\text{cand}}$  with Eq. (4)
4: Set  $\mathcal{D}_{\text{pivot}} \leftarrow \mathcal{D}_{\text{adv}}$ 
5: for  $k = 1, \dots, K_{\text{train}}$  do
6:   Sample minibatch  $\{s_b\}_{b=1}^B \sim \mathcal{D}_{\text{pivot}}$ 
7:   for each  $s_b$  do
8:     Sample  $\{a_{b,i}\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}(\cdot \mid s_b)$ 
9:     Execute the short rollout needed to score each  $a_{b,i}$ 
10:     $r_{b,i} \leftarrow r_{\text{func}}(s_b, a_{b,i})$  for  $i = 1, \dots, G$ 
11:    Compute  $\hat{A}_{b,i} \leftarrow \frac{r_{b,i} - \frac{1}{G} \sum_{j=1}^G r_{b,j}}{\text{std}(\{r_{b,j}\}_{j=1}^G) + \epsilon_{\text{std}}}$ 
12:   end for
13:   Update  $\theta$  using the objective in Eq. (7)
14: end for

```

Proposition 3.1 (Only mixed-outcome turns produce nonzero group-normalized updates). *Let s be a fixed state and let $\{r_i\}_{i=1}^G$ be the binary rewards of a rollout group at s . If all rewards are identical, then the normalized advantages in Eq. (1) are zero for every i . Equivalently, only rollout groups with positive reward variance can contribute a nonzero group-normalized update.*

Proposition 3.1 is the direct reason to filter turns before RL: if a turn is uniformly easy or uniformly impossible under local sampling, then spending rollout budget on that turn does not change the policy under group-normalized training.

For any distribution π over $\mathcal{A}(s)$, define $T_\pi = \{v : \mathcal{A}(s) \rightarrow \mathbb{R} : \mathbb{E}_{a \sim \pi}[v(a)] = 0\}$, $\langle u, v \rangle_{F, \pi} = \mathbb{E}_{a \sim \pi}[u(a)v(a)]$, and let $\nabla^{\text{nat}} J_s(\pi) \in T_\pi$ denote the natural gradient of J_s under this Fisher geometry.

Theorem 3.2 (Reward variance determines the local GRPO signal). *Consider the statewise expected reward objective*

$$J_s(\pi) = \mathbb{E}_{a \sim \pi}[r(s, a)],$$

and the KL path $\pi_{s, \beta}(a) = \frac{\pi_0(a|s)e^{r(s, a)/\beta}}{\sum_{b \in \mathcal{A}(s)} \pi_0(b|s)e^{r(s, b)/\beta}}$. Define the population GRPO score

$$\gamma_{s, \beta} := -\mathbb{E}_{a \sim \pi_{s, \beta}} \left[\frac{r(s, a) - \mathbb{E}_{a' \sim \pi_{s, \beta}}[r(s, a')]}{\sqrt{\text{Var}_{a' \sim \pi_{s, \beta}}(r(s, a'))}} \partial_\beta \log \pi_{s, \beta}(a) \right]. \quad (8)$$

Then we have

$$\gamma_{s, \beta} = \frac{1}{\beta^2} \|\nabla^{\text{nat}} J_s(\pi_{s, \beta})\|_{F, \pi_{s, \beta}} = \frac{\sqrt{\text{Var}_{a \sim \pi_{s, \beta}}(r(s, a))}}{\beta^2}. \quad (9)$$

In particular, at fixed β , states with larger reward variance induce a larger natural-gradient norm and a larger population GRPO score.

Interpretation. Theorem 3.2 is a population, pathwise statement about the idealized KL-regularized statewise update $\pi_{s, \beta} \propto \pi_0 e^{r(s, \cdot)/\beta}$. It shows that, for group-normalized local RL, reward variance is not just a heuristic diagnostic: it is exactly the scale of the local natural-gradient signal along the KL path. For binary verifier rewards, larger variance means a more mixed success/failure turn, which is precisely why PivotRL filters toward mixed-outcome turns.

Theorem 3.3 (Functional reward shifts mass toward acceptable actions with minimal KL change). *Fix $\beta > 0$ and consider the regularized objective*

$$\mathcal{L}_{\text{func}, \beta}(\pi) = \mathbb{E}_{s \sim d} \left[-\mathbb{E}_{a \sim \pi(\cdot | s)}[r_{\text{func}}(s, a)] + \beta \text{KL}(\pi(\cdot | s) \parallel \pi_0(\cdot | s)) \right], \quad (10)$$

where $r_{\text{func}}(s, a) = \mathbf{1}[a \in \mathcal{M}(s)]$ and d is a fixed state distribution. For each state s , define $\rho(s) = \pi_0(\mathcal{M}(s) \mid s)$ and $q_\beta(s) = \frac{\rho(s)e^{1/\beta}}{(1-\rho(s)) + \rho(s)e^{1/\beta}}$. Then $\mathcal{L}_{\text{func}, \beta}$ has a unique minimizer π_β^* such that, for each state s (d -almost surely),

$$\pi_\beta^*(\mathcal{M}(s) \mid s) = q_\beta(s) \geq \rho(s), \quad (11)$$

with strict inequality whenever $0 < \rho(s) < 1$. Moreover, among all distributions satisfying Eq. (11), $\pi_\beta^*(\cdot \mid s)$ is the unique minimizer of

$$\text{KL}(\pi(\cdot \mid s) \parallel \pi_0(\cdot \mid s)),$$

and it preserves the reference ordering within both $\mathcal{M}(s)$ and its complement $\mathcal{M}(s)^c$:

$$\frac{\pi_\beta^*(a \mid s)}{\pi_\beta^*(b \mid s)} = \frac{\pi_0(a \mid s)}{\pi_0(b \mid s)} \quad \text{for } a, b \in \mathcal{M}(s), \quad \frac{\pi_\beta^*(a \mid s)}{\pi_\beta^*(b \mid s)} = \frac{\pi_0(a \mid s)}{\pi_0(b \mid s)} \quad \text{for } a, b \in \mathcal{M}(s)^c. \quad (12)$$

Interpretation. Theorem 3.3 isolates the effect of the functional reward, establishing that the minimizer policy of functional reward-based RL is the KL-projection of the reference policy onto the set of policies with a higher probability mass $q_\beta(s)$ on acceptable actions. From Eq. (12), functional reward-based RL preserves the conditional distribution on both (i) the set of acceptable actions and (ii) its complement. Since the action space of assistant turns is exponentially large, any given action is generally relevant to only a single task. Under this assumption, the complement of acceptable actions corresponds to the set of task-unrelated actions, and consequently, the relative ranking among task-unrelated actions is preserved, explaining PivotRL’s strong retention of OOD performance.

Remark 3.4 (Soft and imperfect verifiers). The binary acceptable-set model in Theorem 3.3 is a special case of a more general score-valued verifier. Proposition B.1 shows that for any verifier score $r(s, a) \in [0, 1]$, the statewise natural-gradient norm equals the standard deviation of $r(s, a)$, and the exact KL-regularized optimizer smoothly favors actions with higher verifier scores. In particular, actions with equal verifier scores preserve their reference-policy probability ratio, while the relative odds of actions with different scores change according to their score difference. Proposition B.2 further shows that if an implemented verifier $\hat{r}$ uniformly approximates an ideal verifier r within $\Delta(s)$ at state s , then their exact KL-regularized statewise optimizers differ in total variation by at most $\tanh(\Delta(s)/(2\beta))$. These propositions support the same qualitative mechanism under fuzzy or imperfect verifier scores, although they concern exact statewise policy-space optima and do not account for finite-sample estimation, function approximation, or optimization error.

Remark 3.5 (Synergy between pivot selection and functional RL). The same functional reward r_{func} is used both to profile candidate states and to compute the group-normalized advantages. The condition $\hat{\sigma}^2(s) > 0$ retains states whose profiled rollouts receive different rewards and can therefore produce a nonzero group-normalized update, while the condition $\hat{\mu}(s) < \lambda_{\text{diff}}$ concentrates the rollout budget on states that are not already reliably solved. Once a state is selected, the functional reward determines which sampled actions receive positive reinforcement. Our theory shows that reward variance determines the statewise natural-gradient magnitude, while the corresponding KL-regularized update exponentially shifts probability mass toward higher-scored actions. Thus, offline filtering determines where local RL is likely to provide an informative signal, whereas the verifier reward determines what the resulting update reinforces.

4 Experiments

PivotRL achieves larger in-domain gains than same-data SFT while nearly eliminating OOD degradation, as shown in Section 4.1. On SWE-Bench, where end-to-end RL (E2E RL) is the standard training paradigm, PivotRL attains comparable accuracy without multi-turn environment rollouts, as shown in Section 4.2. An ablation confirms that both pivot filtering and functional reward are necessary for the full gains, as shown in Section 4.3.

Setup. We train on four agentic domains separately—conversational tool use, software engineering, terminal control, and web browsing—and evaluate each resulting model on their

corresponding benchmarks: τ^2 -Bench (Barres et al., 2025), SWE-Bench Verified (Jimenez et al., 2024), Terminal-Bench (Team, 2025c), and BrowseComp (Wei et al., 2025). All experiments start from Qwen3-30B-A3B-Thinking-2507 (hereafter “Base”), use Nemo-RL (NVIDIA, 2025b) for optimization, and Nemo-Gym (NVIDIA, 2025a) for environment rollouts. For every SFT–PivotRL comparison, the base model, prompts, and expert trajectories are identical. Domain-specific data construction, verifier design, and hyperparameters appear in Appendix A.2.

4.1 In-domain and OOD accuracy

Benchmark	Base	SFT	PivotRL	Δ vs SFT
τ^2 -Bench	44.35	58.44	63.81	+5.37
SWE-Bench Verified	19.07	37.40	32.67	-4.73
Terminal-Bench	5.42	13.75	20.00	+6.25
BrowseComp	2.50	1.50	11.30	+9.80

Table 1: In-domain results. Base denotes Qwen3-30B-A3B-Thinking-2507; SFT and PivotRL use identical training data. PivotRL outperforms SFT on three of four benchmarks, with an average gain of +14.11 over Base versus +9.94 for SFT.

Table 1 reports in-domain accuracy for each training domain. PivotRL improves over SFT on τ^2 -Bench (+5.37), Terminal-Bench (+6.25), and BrowseComp (+9.80), achieving an average in-domain gain of +14.11 over Base compared to +9.94 for SFT. The more important result is OOD retention. Table 2 reports the average change on eight OOD benchmarks across the four training runs. SFT produces an average OOD change of -9.83 , with the worst case after terminal-domain training (AIME25 drops by -64.48 , MATH500 by -34.50). PivotRL stays near Base across all benchmarks, with an average change of $+0.21$ and no single benchmark dropping more than -3.12 . Table 3 provides the full per-domain breakdown. In every training domain, PivotRL (rl) preserves OOD performance while SFT (sft) causes broad regression—most dramatically after terminal-domain training, where SFT drops AIME25 from 86.04 to 21.56.

SWE-Bench Verified is the sole same-data setting in which PivotRL trails SFT. We attribute this gap primarily to the difficulty of constructing an informative local verifier for software engineering. This result identifies verifier quality as an important scaling axis for PivotRL; Appendix A.2 provides the exact verifier rules and a more detailed discussion.

4.2 Comparison to End-to-End RL on SWE-Bench

SWE-Bench is a natural comparison point because E2E RL is the standard training method for software-engineering agents: each GitHub issue requires a multi-turn tool-using trajectory, and the evaluation harness provides a binary success signal. To compare rollout cost, we count total training-time rollout turns: for PivotRL, each training sample is a single-turn rollout, so the total equals the number of training samples; for E2E RL, we sum the turns across all training trajectories.

All methods start from Base and are evaluated with the OpenHands harness (Wang et al., 2025b). Figure 1 plots accuracy against cumulative rollout turns and cumulative rollout time. To reach the same accuracy, PivotRL requires $\sim 4\times$ fewer training-time rollout turns and $\sim 5.5\times$ less wall-clock time on the same number of compute nodes.

4.3 Ablation study

We present ablation results in Table 4. To isolate the contribution of each PivotRL component, we remove one at a time on τ^2 -Bench. Removing filtering reduces accuracy from 63.81 to 59.68; removing functional reward yields 57.34. Pivots concentrate rollouts on states with nonzero advantage signal (Proposition 3.1), and functional reward ensures that correct but textually different actions receive credit.

OOD Benchmark	Base	Δ SFT	Δ PivotRL
IFBench	52.04	-11.46	+0.82
AIME25	86.04	-19.72	-1.20
MATH500	98.05	-8.51	+0.35
LiveCodeBench	66.52	-7.76	-0.17
Scicode	36.83	-11.39	+1.90
MMLU-Pro	80.84	-2.99	+0.31
MMLU-ProX	78.58	-7.16	+0.12
WMT24++	36.97	-9.61	-0.49
Average	66.62	-9.83	+0.21

Table 2: OOD performance change (Δ) relative to Base, averaged across the four training runs in Table 1. SFT degrades by -9.83 on average; PivotRL maintains near-zero change ($+0.21$).

Benchmark	Base	τ^2		SWE	
		sft	rl	sft	rl
Peak In-Domain	–	58.44 (+14.09)	63.81 (+19.46)	37.40 (+18.33)	32.67 (+13.60)
IFBench	52.04	44.42 (-7.62)	53.29 (+1.25)	38.80 (-13.24)	54.79 (+2.75)
AIME25	86.04	76.25 (-9.79)	85.63 (-0.41)	82.29 (-3.75)	85.00 (-1.04)
MATH500	98.05	98.00 (-0.05)	98.35 (+0.30)	98.30 (+0.25)	98.65 (+0.60)
LiveCodeBench	66.52	64.98 (-1.54)	66.96 (+0.44)	57.27 (-9.25)	66.96 (+0.44)
Scicode	36.83	31.95 (-4.88)	38.68 (+1.85)	24.85 (-11.98)	39.87 (+3.04)
MMLU-Pro	80.84	79.81 (-1.03)	81.31 (+0.47)	78.90 (-1.94)	81.13 (+0.29)
MMLU-ProX	78.58	74.31 (-4.27)	78.92 (+0.34)	75.58 (-3.00)	78.81 (+0.23)
WMT24++	36.97	34.25 (-2.72)	36.54 (-0.43)	35.69 (-1.28)	36.66 (-0.31)

Benchmark	Base	Terminal		BrowseComp	
		sft	rl	sft	rl
Peak In-Domain	–	13.75 (+8.33)	20.00 (+14.58)	1.50 (-1.00)	11.30 (+8.80)
IFBench	52.04	24.77 (-27.27)	49.47 (-2.57)	54.35 (+2.31)	53.89 (+1.85)
AIME25	86.04	21.56 (-64.48)	82.92 (-3.12)	85.21 (-0.83)	85.83 (-0.21)
MATH500	98.05	63.55 (-34.50)	98.00 (-0.05)	98.30 (+0.25)	98.40 (+0.35)
LiveCodeBench	66.52	45.15 (-21.37)	64.54 (-1.98)	67.62 (+1.10)	66.96 (+0.44)
Scicode	36.83	9.39 (-27.44)	39.13 (+2.30)	35.58 (-1.25)	37.28 (+0.45)
MMLU-Pro	80.84	71.57 (-9.27)	80.85 (+0.01)	81.13 (+0.29)	81.31 (+0.47)
MMLU-ProX	78.58	59.13 (-19.45)	78.49 (-0.09)	76.68 (-1.90)	78.60 (+0.02)
WMT24++	36.97	6.31 (-30.66)	36.48 (-0.49)	33.18 (-3.79)	36.26 (-0.71)

Table 3: Per-domain OOD breakdown across four single-domain training runs. Each cell shows score followed by Δ relative to Base. PivotRL (rl) consistently preserves OOD performance, whereas SFT (sft) causes broad regression—particularly after terminal-domain training.

Figures 2 and 3 show the training dynamics behind these gains. Under random sampling, per-batch reward variance collapses quickly, indicating that most sampled pivots no longer generate a useful advantage signal. The pivot sets preserve higher reward variance deeper into training and optimize to higher validation accuracy.

5 Related works

Agentic language models interleave natural language with grounded actions for complex environment interactions across tools, code, and web navigation (Yao et al., 2023; Schick et al., 2023; Xu et al., 2023; Qin et al., 2023; Shridhar et al., 2021; Wang et al., 2022; Jimenez et al., 2024; Wang et al., 2025b; Pan et al., 2025; Yao et al., 2022; Wei et al., 2025; Ma et al., 2024; Wang et al., 2024). Offline supervised fine-tuning (SFT) frequently suffers from catastrophic

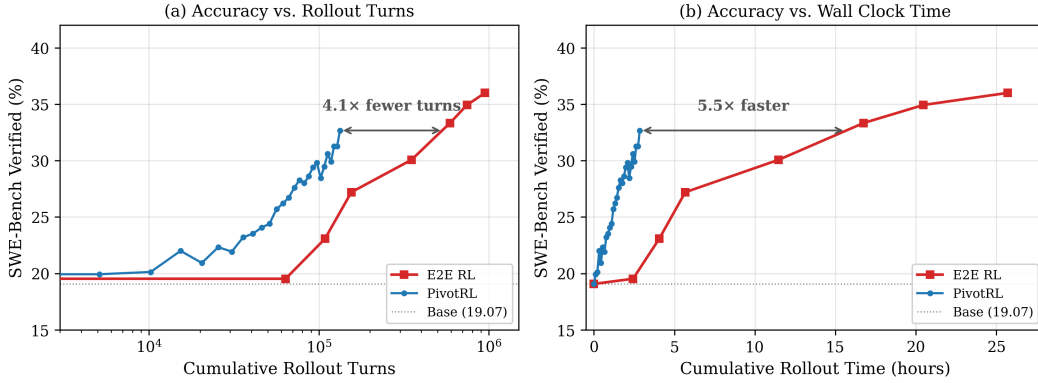


Figure 1: SWE-Bench accuracy vs. (a) cumulative training-time rollout turns and (b) cumulative training-time rollout wall-clock time, using the same number of compute nodes. PivotRL reaches comparable accuracy to E2E RL with $\sim 4\times$ fewer rollout turns and $\sim 5.5\times$ less wall-clock time.

Configuration	τ^2 -Bench
Full PivotRL ($\mathcal{D}_{\text{adv}}$ + functional reward)	63.81
– Pivot filtering ($\mathcal{D}_{\text{cand}}$ + functional reward)	59.68
– Functional reward ($\mathcal{D}_{\text{cand}}$ + strict reward)	57.34
Same-data SFT	58.44
Base	44.35

Table 4: Ablation on τ^2 -Bench. Both pivot filtering and functional reward are necessary; removing either degrades accuracy.

forgetting and degraded generalization (Luo et al., 2025b; Li et al., 2024; Imanov, 2026; Kirkpatrick et al., 2017; Li & Hoiem, 2016), while on-policy RL (Ouyang et al., 2022; Schulman et al., 2017; 2015; Wang et al., 2025a; Song et al., 2024; Zhang et al., 2025) mitigates forgetting by retaining broader capabilities (Shenfeld et al., 2025; Chen et al., 2025). RL with verifiable rewards (RLVR) (Shao et al., 2024; Xin et al., 2024; DeepSeek-AI, 2025a; Team, 2025a) with scalable policy optimization, hierarchical modeling, and multi-agent coordination (Ji et al., 2025; Rohatgi et al., 2025; Xue et al., 2025; Dong et al., 2025a; Yang et al., 2025; Yu et al., 2025; Luo et al., 2025a; Dong et al., 2025b; Su et al., 2025; Zhou et al., 2024; Prabhakar et al., 2025; Zhou et al., 2025; Xi et al., 2025) show success in various domains, while on-policy distillation mitigates off-policy distribution shifts (Lu & Lab, 2025; Zhao et al., 2026; Liu et al., 2025a; Hübotter et al., 2026). To transfer offline knowledge online, prior methods combine expert demonstrations with RL updates (Uchendu et al., 2023; Hester et al., 2018), utilize partial trace prefixes (Setlur et al., 2026), or repurpose SFT tokens as rollout rewards (Ming et al., 2026). Recently, COBALT (Chen et al., 2026) formulate multi-turn code generation as a one-step recoverable MDP (Jain et al., 2025a) and optimize single-step contextual-bandit updates from partial trajectories. Our work lies at this intersection: we target broader agentic domains by converting supervised trajectories into turn-level verifier rewards, enabling on-policy optimization without a separate reward model.

6 Conclusions and future work

PivotRL is a turn-level RL algorithm designed to tractably post-train LLMs for long-horizon agentic tasks. By leveraging local on-policy rollouts and verifier-based, expert-matching rewards, PivotRL achieves comprehensive action-space coverage and dense feedback signals. Our empirical evaluations across diverse agentic domains demonstrate PivotRL’s superior in-domain performance, OOD retention, and sample efficiency.

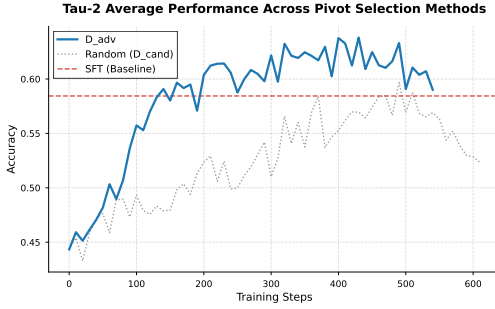


Figure 2: τ^2 -Bench accuracy over RL training. $\mathcal{D}_{adv}$ yields the strongest optimization and outperform same-data SFT.

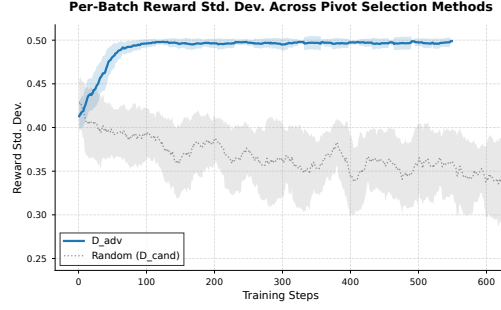


Figure 3: τ^2 -Bench reward standard deviation during RL training. $\mathcal{D}_{adv}$ sets maintain higher reward variance and therefore more informative local updates.

While our results establish the efficacy of this approach, they also highlight a key limitation that presents exciting avenues for future research. The primary limitation of PivotRL is its reliance on high-quality domain-specific verifiers for local credit. To address this limitation, we outline two concrete directions. First, we plan to replace the programmatic verifier with a turn-based action-value model trained from large-scale MCTS-style trajectory data. By maintaining the expert demonstration as the top-ranked action, this approach would provide a scalable, domain-agnostic reward signal without per-domain engineering, thus extending PivotRL to settings where we wish to preserve the algorithm’s full efficiency without the bottleneck of per-domain verifier engineering. Second, we plan to extend the rollout horizon to task completion and integrate our turn-based rewards with the E2E RL’s outcome-grounded signal. This would extend PivotRL to settings where outcome reward is also available and unlock efficiency trade-off between local & global verifier design, further bridging the gap between step-wise supervision and long-horizon autonomy.

References

- Anomaly. Opencode: The open source coding agent, 2026. URL <https://github.com/anomalyco/opencode>.
- Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2 -bench: Evaluating conversational agents in a dual-control environment, 2025. URL <https://arxiv.org/abs/2506.07982>.
- Howard Chen, Noam Razin, Karthik Narasimhan, and Danqi Chen. Retaining by doing: The role of on-policy data in mitigating forgetting, 2025. URL <https://arxiv.org/abs/2510.18874>.
- Ziru Chen, Dongdong Chen, Ruinan Jin, Yingbin Liang, Yujia Xie, and Huan Sun. Bridging online and offline rl: Contextual bandit learning for multi-turn code generation, 2026. URL <https://arxiv.org/abs/2602.03806>.
- Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training, 2025. URL <https://arxiv.org/abs/2501.17161>.
- DeepSeek-AI. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, September 2025a. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL <http://dx.doi.org/10.1038/s41586-025-09422-z>.
- DeepSeek-AI. Deepseek-v3.2: Pushing the frontier of open large language models, 2025b. URL <https://arxiv.org/abs/2512.02556>.
- Guanting Dong, Yifei Chen, Xiaoxi Li, Jiajie Jin, Hongjin Qian, Yutao Zhu, Hangyu Mao, Guorui Zhou, Zhicheng Dou, and Ji-Rong Wen. Tool-star: Empowering LLM-brained

- multi-tool reasoner via reinforcement learning, 2025a. URL <https://arxiv.org/abs/2505.16410>.
- Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. Agentic reinforced policy optimization, 2025b. URL <https://arxiv.org/abs/2507.19849>.
- Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, Gabriel Dulac-Arnold, John Agapiou, Joel Z. Leibo, and Audrunas Gruslys. Deep q-learning from demonstrations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence, AAAI’18/IAAI’18/EAAI’18*. AAAI Press, 2018. ISBN 978-1-57735-800-8.
- Jonas Hübötter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, and Andreas Krause. Reinforcement learning via self-distillation, 2026. URL <https://arxiv.org/abs/2601.20802>.
- Olaf Yunus Laitinen Imanov. Mechanistic analysis of catastrophic forgetting in large language models during continual fine-tuning, 2026. URL <https://arxiv.org/abs/2601.18699>.
- Arnav Kumar Jain, Gonzalo Gonzalez-Pumariega, Wayne Chen, Alexander M Rush, Wenting Zhao, and Sanjiban Choudhury. Multi-turn code generation through single-step rewards, 2025a. URL <https://arxiv.org/abs/2502.20380>.
- Naman Jain, Jaskirat Singh, Manish Shetty, Liang Zheng, Koushik Sen, and Ion Stoica. R2e-gym: Procedural environments and hybrid verifiers for scaling open-weights swe agents, 2025b. URL <https://arxiv.org/abs/2504.07164>.
- Yuxiang Ji, Ziyu Ma, Yong Wang, Guanhua Chen, Xiangxiang Chu, and Liaoni Wu. Tree search for LLM agent reinforcement learning, 2025. URL <https://arxiv.org/abs/2509.21240>.
- Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R Narasimhan. SWE-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=VTF8yNQm66>.
- James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114. URL <https://www.pnas.org/doi/abs/10.1073/pnas.1611835114>.
- Hongyu Li, Liang Ding, Meng Fang, and Dacheng Tao. Revisiting catastrophic forgetting in large language model tuning. *arXiv preprint arXiv:2406.04836*, 2024. doi: 10.48550/arXiv.2406.04836. URL <https://arxiv.org/abs/2406.04836>.
- Zhizhong Li and Derek Hoiem. Learning without forgetting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. URL <https://arxiv.org/abs/1606.09282>.
- Junnan Liu, Linhao Luo, Qianren Mao, Zhijun Chen, Zhuoran Li, Thuy-Trang Vu, and Gholamreza Haffari. Learning with interaction: Agentic distillation for large language model reasoning. In *International Conference on Learning Representations*, 2025a. URL <https://openreview.net/forum?id=zyp9QT5Gf1>.

- Weiwen Liu, Xu Huang, Xingshan Zeng, Xinlong Hao, Shuai Yu, Dexun Li, Shuai Wang, Weinan Gan, Zhengying Liu, Yuanqing Yu, Zezhong Wang, Yuxian Wang, Wu Ning, Yutai Hou, Bin Wang, Chuhan Wu, Xinzhi Wang, Yong Liu, Yasheng Wang, Duyu Tang, Dandan Tu, Lifeng Shang, Xin Jiang, Ruiming Tang, Defu Lian, Qun Liu, and Enhong Chen. Toolace: Winning the points of llm function calling, 2025b. URL <https://arxiv.org/abs/2409.00920>.
- Kevin Lu and Thinking Machines Lab. On-policy distillation. *Thinking Machines Lab: Connectionism*, 2025. URL <https://thinkingmachines.ai/blog/on-policy-distillation>.
- 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, 2025a. URL <https://arxiv.org/abs/2508.03680>.
- Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. An empirical study of catastrophic forgetting in large language models during continual fine-tuning, 2025b. URL <https://arxiv.org/abs/2308.08747>.
- Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. Agentboard: An analytical evaluation board of multi-turn LLM agents, 2024. URL <https://arxiv.org/abs/2401.13178>.
- Rui Ming, Haoyuan Wu, Shoubo Hu, Zhuolun He, and Bei Yu. One-token rollout: Guiding supervised fine-tuning of llms with policy gradient, 2026. URL <https://arxiv.org/abs/2509.26313>.
- MiniMax. Minimax-m2.5, 2026. URL <https://huggingface.co/MiniMaxAI/MiniMax-M2.5>.
- NVIDIA. Nemo gym: An open source library for scaling reinforcement learning environments for llm. <https://github.com/NVIDIA-NeMo/Gym>, 2025a. GitHub repository.
- NVIDIA. Nemo rl: A scalable and efficient post-training library. <https://github.com/NVIDIA-NeMo/RL>, 2025b. GitHub repository.
- OpenAI. Openai codex: The coding agent for software development, 2026. URL <https://developers.openai.com/codex>.
- Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL <https://arxiv.org/abs/2203.02155>.
- Jiayi Pan, Xingyao Wang, Graham Neubig, Navdeep Jaitly, Heng Ji, Alane Suhr, and Yizhe Zhang. Training software engineering agents and verifiers with swe-gym, 2025. URL <https://arxiv.org/abs/2412.21139>.
- Akshara Prabhakar, Zuxin Liu, Ming Zhu, Jianguo Zhang, Tulika Awalgaonkar, Shiyu Wang, Zhiwei Liu, Haolin Chen, Thai Hoang, Juan Carlos Niebles, Shelby Heinecke, Weiran Yao, Huan Wang, Silvio Savarese, and Caiming Xiong. Apigen-mt: Agentic pipeline for multi-turn data generation via simulated agent-human interplay, 2025. URL <https://arxiv.org/abs/2504.03601>.
- Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. Toolllm: Facilitating large language models to master 16000+ real-world apis, 2023. URL <https://arxiv.org/abs/2307.16789>.
- Dhruv Rohatgi, Abhishek Shetty, Donya Saless, Yuchen Li, Ankur Moitra, Andrej Risteski, and Dylan J. Foster. Taming imperfect process verifiers: A sampling perspective on backtracking, 2025. URL <https://arxiv.org/abs/2510.03149>.

- Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools, 2023. URL <https://arxiv.org/abs/2302.04761>.
- John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1889–1897, Lille, France, 07–09 Jul 2015. PMLR. URL <https://proceedings.mlr.press/v37/schulman15.html>.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL <https://arxiv.org/abs/1707.06347>.
- Amrith Setlur, Zijian Wang, Andrew Cohen, Paria Rashidinejad, and Sang Michael Xie. Reuse your flops: Scaling rl on hard problems by conditioning on very off-policy prefixes, 2026. URL <https://arxiv.org/abs/2601.18795>.
- 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.
- Idan Shenfeld, Jyothish Pari, and Pulkit Agrawal. RL’s razor: Why online reinforcement learning forgets less, 2025. URL <https://arxiv.org/abs/2509.04259>.
- Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning, 2021. URL <https://arxiv.org/abs/2010.03768>.
- Yifan Song, Da Yin, Xiang Yue, Jie Huang, Sujian Li, and Bill Yuchen Lin. Trial and error: Exploration-based trajectory optimization for LLM agents, 2024. URL <https://arxiv.org/abs/2403.02502>.
- Hongjin Su, Shizhe Diao, Ximing Lu, Mingjie Liu, Jiacheng Xu, Xin Dong, Yonggan Fu, Peter Belcak, Hanrong Ye, Hongxu Yin, Yi Dong, Evelina Bakhturina, Tao Yu, Yejin Choi, Jan Kautz, and Pavlo Molchanov. Toolorchestra: Elevating intelligence via efficient model and tool orchestration, 2025. URL <https://arxiv.org/abs/2511.21689>.
- GLM-4.5 Team. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models, 2025a. URL <https://arxiv.org/abs/2508.06471>.
- Kimi Team. Kimi k2: Open agentic intelligence, 2025b. URL <https://arxiv.org/abs/2507.20534>.
- The Terminal-Bench Team. Terminal-bench: A benchmark for ai agents in terminal environments, Apr 2025c. URL <https://github.com/laude-institute/terminal-bench>.
- Ikechukwu Uchendu, Ted Xiao, Yao Lu, Banghua Zhu, Mengyuan Yan, Joséphine Simon, Matthew Bennice, Chuyuan Fu, Cong Ma, Jiantao Jiao, Sergey Levine, and Karol Hausman. Jump-start reinforcement learning. In *International Conference on Machine Learning*, 2023.
- Hanlin Wang, Chak Tou Leong, Jiashuo Wang, Jian Wang, and Wenjie Li. Spa-rl: Reinforcing LLM agents via stepwise progress attribution, 2025a. URL <https://arxiv.org/abs/2505.20732>.
- Ruiyi Wang, Haofei Yu, Wenxin Zhang, Zhengyang Qi, Maarten Sap, Graham Neubig, Yonatan Bisk, and Hao Zhu. Sotopia- π : Interactive learning of socially intelligent language agents, 2024. URL <https://arxiv.org/abs/2403.08715>.
- Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. Science-world: Is your agent smarter than a 5th grader?, 2022. URL <https://arxiv.org/abs/2203.07540>.

- Xingyao Wang, Boxuan Li, Yufan Song, Frank F. Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, Hoang H. Tran, Fuqiang Li, Ren Ma, Mingzhang Zheng, Bill Qian, Yanjun Shao, Niklas Muennighoff, Yizhe Zhang, Binyuan Hui, Junyang Lin, Robert Brennan, Hao Peng, Heng Ji, and Graham Neubig. Openhands: An open platform for ai software developers as generalist agents, 2025b. URL <https://arxiv.org/abs/2407.16741>.
- Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. Browsecomp: A simple yet challenging benchmark for browsing agents, 2025. URL <https://arxiv.org/abs/2504.12516>.
- Zhiheng Xi, Jixuan Huang, Chenyang Liao, Baodai Huang, Honglin Guo, Jiaqi Liu, Rui Zheng, Junjie Ye, Jiazheng Zhang, Wenxiang Chen, Wei He, Yiwen Ding, Guanyu Li, Zehui Chen, Zhengyin Du, Xuesong Yao, Yufei Xu, Jiecao Chen, Tao Gui, Zuxuan Wu, Qi Zhang, Xuanjing Huang, and Yu-Gang Jiang. Agentgym-rl: Training LLM agents for long-horizon decision making through multi-turn reinforcement learning, 2025. URL <https://arxiv.org/abs/2509.08755>.
- Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. Osvorld: Benchmarking multimodal agents for open-ended tasks in real computer environments, 2024. URL <https://arxiv.org/abs/2404.07972>.
- Huajian Xin, Z. Z. Ren, Junxiao Song, Zhihong Shao, Wanbiao Zhao, Haocheng Wang, Bo Liu, Liyue Zhang, Xuan Lu, Qiushi Du, Wenjun Gao, Qihao Zhu, Dejian Yang, Zhibin Gou, Z. F. Wu, Fuli Luo, and Chong Ruan. Deepseek-prover-v1.5: Harnessing proof assistant feedback for reinforcement learning and monte-carlo tree search, 2024. URL <https://arxiv.org/abs/2408.08152>.
- Qiantong Xu, Fenglu Hong, Bo Li, Changran Hu, Zhengyu Chen, and Jian Zhang. On the tool manipulation capability of open-source large language models, 2023. URL <https://arxiv.org/abs/2305.16504>.
- Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru Li, Xiaosen Zheng, Zejun Ma, and Bo An. Simpletir: End-to-end reinforcement learning for multi-turn tool-integrated reasoning, 2025. URL <https://arxiv.org/abs/2509.02479>.
- Shihui Yang, Chengfeng Dou, Peidong Guo, Kai Lu, Qiang Ju, Fei Deng, and Rihui Xin. Dcpo: Dynamic clipping policy optimization, 2025. URL <https://arxiv.org/abs/2509.02333>.
- Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents, 2022. URL <https://arxiv.org/abs/2207.01206>.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023. URL <https://arxiv.org/abs/2210.03629>.
- Qiyang Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL <https://arxiv.org/abs/2503.14476>.
- Zijing Zhang, Ziyang Chen, Mingxiao Li, Zhaopeng Tu, and Xiaolong Li. Rlvmr: Reinforcement learning with verifiable meta-reasoning rewards for robust long-horizon agents, 2025. URL <https://arxiv.org/abs/2507.22844>.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models, 2026.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. Archer: Training language model agents via hierarchical multi-turn rl, 2024. URL <https://arxiv.org/abs/2402.19446>.

Yifei Zhou, Song Jiang, Yuandong Tian, Jason Weston, Sergey Levine, Sainbayar Sukhbaatar, and Xian Li. Sweet-rl: Training multi-turn LLM agents on collaborative reasoning tasks, 2025. URL <https://arxiv.org/abs/2503.15478>.

A Experiment details

This appendix provides domain-specific training procedures, terminology conventions, and detailed ablation analyses for the experiments in Section 4.

A.1 Terminology conventions

Throughout the experiments, we call every extracted assistant turn a *pivot candidate* and reserve the term *pivot* for the offline-filtered subset used for PivotRL training. An *action* is the full assistant completion at a model-call boundary, not an individual token. Depending on the benchmark, that completion may be a conversational tool-use turn, a coding-agent tool call, a bash command, or a search/browsing step. Natural-language-only assistant turns are also actions when the benchmark includes them.

In the single-domain experiments, OOD change denotes the score difference relative to the base model on benchmarks outside the training domain.

For pivot selection, we use two subsets: *random* ($\mathcal{D}_{\text{cand}}$), which uses all pivot candidates, and *low-reward-mean* ($\mathcal{D}_{\text{adv}}$), which requires both nonzero reward variance and a gap between the demonstrated action and the reference-policy mean.

A.2 Domain-specific training details

In all four domains, pivot candidates are extracted once from the demonstration traces, profiled offline, and filtered before RL training; training then samples only from the retained set. We detail the domain-specific action definitions, data construction, and local verifiers below.

τ^2 -Bench. We curate an SFT trajectory dataset of 281,774 trajectories across 838 domains using a synthetic data-generation pipeline similar to ToolAce (Liu et al., 2025b), Kimi-K2 (Team, 2025b), and DeepSeek-V3.2 (DeepSeek-AI, 2025b). We train on top of Qwen3-30B-A3B-Thinking-2507. Every assistant turn in the trace is first treated as a pivot candidate. In this domain, an action is the full assistant turn at the model-call boundary, which may contain natural language, tool calls, or both. We then evaluate random and low-reward-mean pivot sets as described in Section 3.1.

Terminal-Bench. We collect resolved trajectories from Qwen/Qwen3-Coder-480B-A35B-Instruct and moonshotai/Kimi-K2-Instruct using the Terminus-1 and Terminus-2 agents. Every assistant bash action is treated as a pivot candidate, and an action in this domain is the next bash command produced at the model-call boundary. We start from the low-reward-mean set $\mathcal{D}_{\text{adv}}$, with an additional command-deduplication step to improve diversity. The final dataset contains approximately 20,000 samples. The local verifier is deterministic and string-only: it asks whether the sampled command is locally interchangeable with the demonstrated command via output-schema validation and normalized string similarity check.

SWE-Bench Verified. We use an internal trajectory dataset generated with OpenHands (Wang et al., 2025b), OpenCode (Anomaly, 2026), and Codex (OpenAI, 2026) on tasks from SWE-Gym (Pan et al., 2025) and R2E-Gym (Jain et al., 2025b), using MiniMaxAI/MiniMax-M2.5 (MiniMax, 2026). We evaluate mean@3 with the OpenHands harness. Every non-error tool-call action is treated as a pivot candidate, and an action in this domain is the next assistant tool call in the coding trace. Filtering to $\mathcal{D}_{\text{adv}}$ yields a final training set of 87,718 samples. The local verifier requires the correct tool identity and argument schema including paths, IDs, single-token values, numbers, and booleans. This is a deliberately coarse local signal: it checks whether the model selected the correct next *kind* of operation at the pivot (e.g., search, open, edit, or run), without attempting to score tool arguments or patch quality at every turn. Final task success is determined only by the full SWE-Bench evaluation harness.

Domain	Gym configuration	Verifier code	Pivot dataset
τ^2 -Bench	config	code	data
Terminal-Bench	config	code	–
SWE-Bench	config	code	data
BrowseComp	config	code	–

Table 5: Released configurations, verifier implementations, and datasets.

This verifier captures meaningful information about the local action: for example, it distinguishes whether the agent should search, inspect a file, edit code, or execute a command, while also checking the associated schema and exact short arguments. However, it does not execute the resulting patch, test whether the edit resolves the issue, or semantically evaluate alternative edit strategies at each intermediate turn. The permissive handling of long strings reduces false negatives but also weakens the local credit signal. We therefore view SWE-Bench as the clearest demonstration of the importance of verifier quality.

For the E2E RL comparison in Section 4.2, PivotRL trains with a batch size of 1024 (64 prompts $\times$ 16 generations) at 1 turn per sample, reaching 32.67% accuracy at step 130 with $\sim 133\text{K}$ cumulative rollout turns. The E2E RL baseline trains with a batch size of 512 (16 prompts $\times$ 32 generations) at 12–25 turns per trajectory, reaching the same 32.67% accuracy at step ~ 72 with $\sim 542\text{K}$ cumulative rollout turns. At this matched accuracy, PivotRL requires $\sim 4\times$ fewer rollout turns and $\sim 5.5\times$ less wall-clock time (Figure 1). Both methods use the same number of compute nodes.

BrowseComp. We start from a multi-hop question-answer dataset and use an online search engine together with DeepSeek-V3.2 to generate browsing trajectories. Each search-related assistant step at a model-call boundary is treated as a pivot candidate; in this domain, an action is the next browsing step, such as issuing a search query, opening a result, or taking the next evidence-gathering action. The final dataset contains 13,215 samples.

A.2.1 Released artifacts

We release the domain-specific NeMo Gym configurations, verifier implementations, and available pivot datasets.

A.3 Detailed ablation analyses

A.3.1 Effect of turn selection strategy on τ^2 -Bench

We compare two pivot selection strategies on τ^2 -Bench:

- random pivots ($\mathcal{D}_{\text{cand}}$): no filtering,
- low-reward-mean ($\mathcal{D}_{\text{adv}}$): mixed-outcome pivots further filtered for a large gap between the demonstrated action’s reward and the reference-policy mean, given by Eq. (5).

Table 6 shows a monotonic benefit from more selective filtering: random pivots already improve over SFT (59.68 vs. 58.44), and low-reward-mean pivots yield the best result (63.81).

A.3.2 Additional results on a larger model

To evaluate whether the gains are specific to the Qwen3-30B base model used in the controlled experiments, we additionally compare SFT and PivotRL on Nemotron 3 Super 120B-A12B. Table 7 shows consistent gains for PivotRL across conversational tool use, terminal control, and web browsing.

These results provide corroborating evidence that the empirical advantage of PivotRL transfers beyond the main Qwen3-30B experiments.

	τ^2 -Airline	τ^2 -Retail	τ^2 -Telecom	τ^2 -Average
gpt-oss-120b-high	55.33	58.19	67.20	60.24
gpt-oss-20b-high	43.33	50.20	52.63	48.72
Qwen3-235B-A22B-Thinking-2507	50.00	66.08	46.78	54.29
Qwen3-30B-A3B-Thinking-2507	50.00	54.39	28.65	44.35
Qwen + SFT	51.33	58.19	65.79	58.44
Qwen + Random Pivots ($\mathcal{D}_{\text{cand}}$)	58.00	63.74	57.31	59.68
Qwen + Low-Reward-Mean Pivots ($\mathcal{D}_{\text{adv}}$)	58.67	69.01	63.74	63.81

Table 6: Effect of pivot selection on τ^2 -Bench (best checkpoint over training). Random turns improve over the base model and low-reward-mean pivots perform even stronger.

Benchmark	SFT	PivotRL
τ^2 -Bench	48.00	64.00
Terminal-Bench	23.33	34.17
BrowseComp	13.03	25.04

Table 7: Additional results on Nemotron 3 Super 120B-A12B. All entries are accuracy in percent.

A.3.3 Sensitivity to profiling hyperparameters

We evaluate sensitivity to the difficulty threshold λ_{diff} and the number K of profiling samples. Table 8 reports the best average τ^2 -Bench accuracy over training. PivotRL outperforms the same-data SFT baseline for every tested setting, while the main configuration achieves the strongest result.

A.3.4 Profiling staleness and optional re-profiling

The main experiments use one-time fixed profiling under the reference policy. As training progresses, initially informative pivots may become easier and their reward variance may decrease. Periodic re-profiling is an optional extension that can refresh the pivot set, but it is not included in the main fixed-profiling efficiency comparison. If used, every additional re-profiling stage should be counted as extra inference and verifier cost; a full additional pass requires at most $K|\mathcal{D}_{\text{cand}}|$ local single-turn generations.

In separate internal TerminalBench-2.0 experiments, performance saturated at approximately 48%, and a later re-profiling stage improved performance to approximately 51%. We do not include this result in the main fixed-profiling efficiency claim because they induce additional inference & verifier cost to the training phase and this tradeoff complicates our comparison. Instead, it suggests that dynamic sampling, online hard-example mining, or periodic re-profiling can be useful orthogonal extensions when the initial pivot set becomes stale, provided their additional cost is reported explicitly.

B Omitted proofs

B.1 Proof of Theorem 3.2

Proof. Fix s and $\beta > 0$. Write for simplicity

$$\pi = \pi_{s,\beta}, \quad q = \mathbb{E}_{a \sim \pi}[r(s, a)], \quad \sigma^2 = \text{Var}_{a \sim \pi}(r(s, a)).$$

We first compute the natural gradient of J_s at π . Let $v \in T_\pi$, and consider the exponential perturbation

$$\pi_t(a) = \frac{\pi(a)e^{tv(a)}}{\sum_{b \in \mathcal{A}(s)} \pi(b)e^{tv(b)}}.$$

Ablation	Setting	Accuracy (%)
Difficulty threshold λ_{diff}	SFT baseline	58.44
	40%	61.89
	60% (main)	63.81
	80%	60.56
Profiling samples K	SFT baseline	58.44
	4	61.93
	8	62.07
	16 (main)	63.81

Table 8: Sensitivity to the pivot-selection threshold and profiling sample count. Accuracy is the best average τ^2 -Bench performance over training.

For any $v \in T_\pi$, we have

$$\left. \frac{d}{dt} \right|_{t=0} \log \pi_t(a) = v(a) - \sum_{b \in \mathcal{A}(s)} \pi(b)v(b) = v(a).$$

Thus

$$\begin{aligned} DJ_s(\pi)[v] &= \left. \frac{d}{dt} \right|_{t=0} \mathbb{E}_{a \sim \pi_t} [r(s, a)] \\ &= \left. \frac{d}{dt} \right|_{t=0} \sum_{a \in \mathcal{A}(s)} \pi_t(a) r(s, a) \\ &= \sum_{a \in \mathcal{A}(s)} \pi(a) r(s, a) v(a) \\ &= \sum_{a \in \mathcal{A}(s)} \pi(a) (r(s, a) - q) v(a) \\ &= \langle r(s, \cdot) - q, v \rangle_{F, \pi}. \end{aligned}$$

Since this holds for every $v \in T_\pi$, the Riesz representative is

$$\nabla^{\text{nat}} J_s(\pi) = r(s, \cdot) - q.$$

We then have

$$\begin{aligned} \|\nabla^{\text{nat}} J_s(\pi)\|_{F, \pi}^2 &= \mathbb{E}_{a \sim \pi} [(r(s, a) - q)^2] \\ &= \text{Var}_{a \sim \pi}(r(s, a)) \\ &= \sigma^2 \end{aligned} \tag{13}$$

Next, we compute the β -derivative of the exponential-tilted path. Define

$$Z_s(\beta) = \sum_{b \in \mathcal{A}(s)} \pi_0(b | s) e^{r(s, b)/\beta}.$$

Then

$$\log \pi_{s, \beta}(a) = \log \pi_0(a | s) + \frac{r(s, a)}{\beta} - \log Z_s(\beta).$$

Differentiating with respect to β gives

$$\partial_\beta \log \pi_{s, \beta}(a) = -\frac{r(s, a)}{\beta^2} - \frac{Z'_s(\beta)}{Z_s(\beta)}.$$

Now

$$\begin{aligned}
Z'_s(\beta) &= \sum_{b \in \mathcal{A}(s)} \pi_0(b | s) e^{r(s,b)/\beta} \left(-\frac{r(s,b)}{\beta^2} \right) \\
&= -\frac{1}{\beta^2} \sum_{b \in \mathcal{A}(s)} \pi_0(b | s) e^{r(s,b)/\beta} r(s,b) \\
&= -\frac{Z_s(\beta)}{\beta^2} \mathbb{E}_{b \sim \pi_{s,\beta}}[r(s,b)] \\
&= -\frac{q}{\beta^2} Z_s(\beta).
\end{aligned}$$

Therefore

$$\begin{aligned}
\partial_\beta \log \pi_{s,\beta}(a) &= -\frac{1}{\beta^2} (r(s,a) - q) \\
&= -\frac{1}{\beta^2} \nabla^{\text{nat}} J_s(\pi_{s,\beta})(a).
\end{aligned}$$

If $\sigma = 0$, then $r(s,a) = q$ for π -almost every a , so

$$\nabla^{\text{nat}} J_s(\pi) = 0, \text{ and } \frac{1}{\beta^2} \|\nabla^{\text{nat}} J_s(\pi)\|_{F,\pi} = 0.$$

Now assume $\sigma > 0$. Then

$$\begin{aligned}
\gamma_{s,\beta} &= -\mathbb{E}_{a \sim \pi} \left[\frac{r(s,a) - q}{\sigma} \partial_\beta \log \pi(a) \right] \\
&= -\mathbb{E}_{a \sim \pi} \left[\frac{r(s,a) - q}{\sigma} \left(-\frac{1}{\beta^2} (r(s,a) - q) \right) \right] \\
&= \frac{1}{\beta^2 \sigma} \mathbb{E}_{a \sim \pi} [(r(s,a) - q)^2] \\
&= \frac{\sigma}{\beta^2}.
\end{aligned}$$

Combining this with Eq. (13), we have

$$\gamma_{s,\beta} = \frac{1}{\beta^2} \|\nabla^{\text{nat}} J_s(\pi_{s,\beta})\|_{F,\pi_{s,\beta}} = \frac{\sigma}{\beta^2}.$$

This proves the theorem. □

B.2 Proof of Theorem 3.3

Proof. Since

$$\mathcal{L}_{\text{func},\beta}(\pi) = \mathbb{E}_{s \sim d}[\ell_s(\pi)]$$

has no coupling across states, it suffices to solve the pointwise minimization problem for each fixed state s . The claim for the population objective then follows immediately d -almost surely.

Fix a state s , and write $M = \mathcal{M}(s)$, $\rho = \pi_0(M | s)$, $\pi_0(a) = \pi_0(a | s)$.

We first handle the boundary cases. If $\rho = 0$, then $M = \emptyset$ because $\pi_0(a) > 0$ for every $a \in \mathcal{A}(s)$. Hence $r_{\text{func}}(s,a) \equiv 0$, so $\ell_s(\pi) = \beta \text{KL}(\pi \| \pi_0)$, which is uniquely minimized at $\pi = \pi_0$. If $\rho = 1$, then $M = \mathcal{A}(s)$, so $r_{\text{func}}(s,a) \equiv 1$, and $\ell_s(\pi) = -1 + \beta \text{KL}(\pi \| \pi_0)$, which is again uniquely minimized at $\pi = \pi_0$.

It remains to consider the nondegenerate case $0 < \rho < 1$. Let p be any distribution on $\mathcal{A}(s)$, and let

$$q = p(M) = \sum_{a \in M} p(a).$$

For $q \in (0, 1)$, define the conditional distributions

$$p_+(a) = \frac{p(a)}{q}, \quad a \in M, \quad p_-(a) = \frac{p(a)}{1-q}, \quad a \notin M,$$

and similarly

$$\pi_{0,+}(a) = \frac{\pi_0(a)}{\rho}, \quad a \in M, \quad \pi_{0,-}(a) = \frac{\pi_0(a)}{1-\rho}, \quad a \notin M.$$

Using the chain rule for KL divergence, we obtain

$$\begin{aligned} \text{KL}(p \parallel \pi_0) &= \sum_{a \in M} p(a) \log \frac{p(a)}{\pi_0(a)} + \sum_{a \notin M} p(a) \log \frac{p(a)}{\pi_0(a)} \\ &= q \text{KL}(p_+ \parallel \pi_{0,+}) + (1-q) \text{KL}(p_- \parallel \pi_{0,-}) + q \log \frac{q}{\rho} + (1-q) \log \frac{1-q}{1-\rho}. \end{aligned}$$

Since $\mathbb{E}_{a \sim p}[r_{\text{func}}(s, a)] = q$, the pointwise objective becomes

$$\ell_s(p) = -q + \beta \left[q \text{KL}(p_+ \parallel \pi_{0,+}) + (1-q) \text{KL}(p_- \parallel \pi_{0,-}) + q \log \frac{q}{\rho} + (1-q) \log \frac{1-q}{1-\rho} \right].$$

For fixed q , the last two terms are constants, while the first two KL terms are nonnegative and vanish if and only if

$$p_+ = \pi_{0,+}, \quad p_- = \pi_{0,-}.$$

Hence, for fixed q , the unique minimizer is

$$\bar{p}_q(a) = \begin{cases} \frac{q}{\rho} \pi_0(a), & a \in M, \\ \frac{1-q}{1-\rho} \pi_0(a), & a \notin M. \end{cases}$$

Substituting $p = \bar{p}_q$ eliminates the conditional KL terms and yields the scalar problem

$$\phi_s(q) = -q + \beta \left[q \log \frac{q}{\rho} + (1-q) \log \frac{1-q}{1-\rho} \right], \quad q \in [0, 1],$$

with the usual convention $0 \log 0 = 0$.

For $q \in (0, 1)$,

$$\phi'_s(q) = -1 + \beta \log \frac{q(1-\rho)}{(1-q)\rho},$$

$$\phi''_s(q) = \beta \left(\frac{1}{q} + \frac{1}{1-q} \right) > 0.$$

Therefore ϕ_s is strictly convex on $(0, 1)$ and has at most one stationary point there. Solving

$\phi'_s(q) = 0$ gives $q = \frac{\rho e^{1/\beta}}{(1-\rho) + \rho e^{1/\beta}}$. Hence the unique minimizer of ℓ_s is $\pi_\beta^*(\cdot \mid s) = \bar{p}_{q_\beta(s)}$, namely

$$\pi_\beta^*(a \mid s) = \begin{cases} \frac{q_\beta(s)}{\rho(s)} \pi_0(a \mid s), & a \in \mathcal{M}(s), \\ \frac{1-q_\beta(s)}{1-\rho(s)} \pi_0(a \mid s), & a \notin \mathcal{M}(s). \end{cases} \quad (14)$$

Among all distributions with total acceptable mass $q_\beta(s)$, the unique KL minimizer is precisely the block-rescaled distribution above.

From Eq. (14), the ordering-preservation statements are immediate. If $a, b \in \mathcal{M}(s)$, then

$$\frac{\pi_\beta^*(a | s)}{\pi_\beta^*(b | s)} = \frac{\pi_0(a | s)}{\pi_0(b | s)}.$$

Likewise, if $a, b \notin \mathcal{M}(s)$, then

$$\frac{\pi_\beta^*(a | s)}{\pi_\beta^*(b | s)} = \frac{\pi_0(a | s)}{\pi_0(b | s)}.$$

This completes the proof. $\square$

B.3 Extensions to soft and imperfect verifiers

Proposition B.1 (Extension to soft verifier scores). *Assume that $\mathcal{A}(s)$ is finite and that $\pi_0(a | s) > 0$ for every state s and action $a \in \mathcal{A}(s)$. Let $r(s, a) \in [0, 1]$ be a possibly non-binary verifier score. For a fixed state s and a full-support distribution π over $\mathcal{A}(s)$, define*

$$J_s(\pi) = \mathbb{E}_{a \sim \pi}[r(s, a)].$$

The natural gradient under the Fisher geometry defined in Section 3.2 satisfies

$$\nabla^{\text{nat}} J_s(\pi) = r(s, \cdot) - \mathbb{E}_{a \sim \pi}[r(s, a)],$$

$$\|\nabla^{\text{nat}} J_s(\pi)\|_{F, \pi}^2 = \text{Var}_{a \sim \pi}(r(s, a)).$$

For $\beta > 0$ and a fixed state distribution d , consider the regularized objective

$$L_{r, \beta}(\pi) = \mathbb{E}_{s \sim d} \left[-\mathbb{E}_{a \sim \pi(\cdot | s)}[r(s, a)] + \beta \text{KL}(\pi(\cdot | s) \| \pi_0(\cdot | s)) \right].$$

This objective has a unique minimizer $\pi_{r, \beta}^*$, characterized d -almost surely by

$$\pi_{r, \beta}^*(a | s) = \frac{\pi_0(a | s) \exp(r(s, a) / \beta)}{\sum_{b \in \mathcal{A}(s)} \pi_0(b | s) \exp(r(s, b) / \beta)}.$$

Consequently, for every $a, b \in \mathcal{A}(s)$,

$$\frac{\pi_{r, \beta}^*(a | s)}{\pi_{r, \beta}^*(b | s)} = \frac{\pi_0(a | s)}{\pi_0(b | s)} \exp\left(\frac{r(s, a) - r(s, b)}{\beta}\right).$$

When $r(s, a) = r_{\text{func}}(s, a)$, the minimizer $\pi_{r, \beta}^*$ reduces to the binary functional-reward minimizer π_β^* in Theorem 3.3.

Proof. Fix a state s and a full-support distribution π , and let $q := \mathbb{E}_{a \sim \pi}[r(s, a)]$. For $v \in T_\pi$, consider the same exponential perturbation used in the proof of Theorem 3.2,

$$\pi_t(a) = \frac{\pi(a) \exp(tv(a))}{\sum_{b \in \mathcal{A}(s)} \pi(b) \exp(tv(b))}.$$

Because $v \in T_\pi$, differentiating at $t = 0$ yields

$$\left. \frac{d}{dt} \right|_{t=0} \log \pi_t(a) = v(a).$$

It follows that

$$\begin{aligned} DJ_s(\pi)[v] &= \left. \frac{d}{dt} \right|_{t=0} \mathbb{E}_{a \sim \pi_t}[r(s, a)] \\ &= \sum_{a \in \mathcal{A}(s)} \pi(a) r(s, a) v(a) \\ &= \sum_{a \in \mathcal{A}(s)} \pi(a) (r(s, a) - q) v(a) \\ &= \langle r(s, \cdot) - q, v \rangle_{F, \pi}. \end{aligned}$$

Since this identity holds for every $v \in T_\pi$, the Riesz representative is $\nabla^{\text{nat}} J_s(\pi) = r(s, \cdot) - q$. Therefore,

$$\|\nabla^{\text{nat}} J_s(\pi)\|_{F,\pi}^2 = \text{Var}_{a \sim \pi}(r(s, a)).$$

The objective $L_{r,\beta}$ has no coupling across states, so it suffices to solve the pointwise minimization problem for each fixed state s . Suppress the conditioning on s , write $\pi_0(a) = \pi_0(a | s)$, and let p be any distribution on $\mathcal{A}(s)$. The pointwise objective is

$$\ell_{s,r}(p) = - \sum_{a \in \mathcal{A}(s)} p(a) r(s, a) + \beta \sum_{a \in \mathcal{A}(s)} p(a) \log \frac{p(a)}{\pi_0(a)}.$$

The function $\ell_{s,r}$ is strictly convex in p . Introducing a Lagrange multiplier λ for the constraint $\sum_a p(a) = 1$, we have the first-order condition $0 = -r(s, a) + \beta \left(\log \frac{p(a)}{\pi_0(a)} + 1 \right) + \lambda$. Rearranging and normalizing over $\mathcal{A}(s)$ yields

$$p(a) = \frac{\pi_0(a) \exp(r(s, a)/\beta)}{\sum_{b \in \mathcal{A}(s)} \pi_0(b) \exp(r(s, b)/\beta)}.$$

This distribution is feasible and has full support, and strict convexity makes it the unique pointwise minimizer. Taking the ratio of the minimizing probabilities of a and b gives

$$\frac{p(a)}{p(b)} = \frac{\pi_0(a)}{\pi_0(b)} \exp \left(\frac{r(s, a) - r(s, b)}{\beta} \right).$$

The characterization of $\pi_{r,\beta}^*$ follows pointwise for d -almost every state. Finally, substituting $r = r_{\text{func}}$ recovers the two-block rescaling characterized by Theorem 3.3. $\square$

Proposition B.2 (Robustness to verifier perturbations). *Under the assumptions of Proposition B.1, let r and $\hat{r}$ be two verifier scores and define*

$$\Delta(s) = \max_{a \in \mathcal{A}(s)} |\hat{r}(s, a) - r(s, a)|.$$

Let $\pi_{r,\beta}^*$ and $\pi_{\hat{r},\beta}^*$ denote the corresponding minimizers of $L_{r,\beta}$ and $L_{\hat{r},\beta}$. Then, for every state s ,

$$\left\| \pi_{\hat{r},\beta}^*(\cdot | s) - \pi_{r,\beta}^*(\cdot | s) \right\|_{\text{TV}} \leq \tanh \left(\frac{\Delta(s)}{2\beta} \right),$$

where $\|p - q\|_{\text{TV}} = \frac{1}{2} \sum_{a \in \mathcal{A}(s)} |p(a) - q(a)|$. Moreover, for every $a, b \in \mathcal{A}(s)$,

$$\left| \log \frac{\pi_{\hat{r},\beta}^*(a | s) / \pi_{\hat{r},\beta}^*(b | s)}{\pi_{r,\beta}^*(a | s) / \pi_{r,\beta}^*(b | s)} \right| \leq \frac{2\Delta(s)}{\beta}.$$

Proof. Fix a state s and suppress the conditioning on s . Write $p(a) := \pi_{r,\beta}^*(a | s)$, $\hat{p}(a) := \pi_{\hat{r},\beta}^*(a | s)$, $\delta(a) := \frac{\hat{r}(s,a) - r(s,a)}{\beta}$. By Proposition B.1, we have

$$\frac{\hat{p}(a)}{p(a)} = \frac{\exp(\delta(a))}{\mathbb{E}_{b \sim p}[\exp(\delta(b))]}.$$

Define the likelihood ratio $h(a) = \hat{p}(a)/p(a)$ and let $m := \min_{a \in \mathcal{A}(s)} h(a)$, $M := \max_{a \in \mathcal{A}(s)} h(a)$. Because $|\delta(a)| \leq \Delta(s)/\beta$ for every action,

$$\frac{M}{m} \leq \exp \left(\frac{2\Delta(s)}{\beta} \right) =: \kappa.$$

If $M = m$, then $h(a) = 1$ for every action and the total-variation bound is immediate. Suppose that $M > m$. Because $\mathbb{E}_{a \sim p}[h(a)] = 1$, we have $m \leq 1 \leq M$ and

$$\begin{aligned} \|\hat{p} - p\|_{\text{TV}} &= \mathbb{E}_{a \sim p}[(h(a) - 1)_+] \\ &\leq \frac{M-1}{M-m} \mathbb{E}_{a \sim p}[h(a) - m] \\ &= \frac{(M-1)(1-m)}{M-m} \end{aligned}$$

where the inequality is due to $(h-1)_+ \leq \frac{M-1}{M-m}(h-m)$ for every $h \in [m, M]$.

Under the constraints $m \leq 1 \leq M$ and $M/m \leq \kappa$, the right-hand side is maximized at $m = \kappa^{-1/2}$ and $M = \kappa^{1/2}$. Therefore,

$$\|\hat{p} - p\|_{\text{TV}} \leq \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} = \tanh\left(\frac{\Delta(s)}{2\beta}\right).$$

The pairwise-odds bound follows directly from taking absolute values and applying the definition of $\Delta(s)$ to

$$\log \frac{\hat{p}(a)/\hat{p}(b)}{p(a)/p(b)} = \frac{\hat{r}(s, a) - r(s, a) - \hat{r}(s, b) + r(s, b)}{\beta}.$$

□