Dataset Mismatch Matters in Group Relative Policy Optimization for Reinforcement Learning from Verifiable Rewards

Zhen-Yu Zhang
RIKEN-AIP
zhen-yu.zhang@riken.jp

Jiandong Zhang
Northeastern University
zhang.jiando@northeastern.edu

Huaxiu Yao
UNC-Chapel Hill
huaxiu@cs.unc.edu

Gang Niu
RIKEN-AIP
gang.niu.ml@gmail.com

Masashi Sugiyama
RIKEN-AIP
The University of Tokyo
sugi@k.u-tokyo.ac.jp

Abstract

Recently, *reinforcement learning from verifiable rewards* (RLVR) is practically very important, where the leading approach is *group relative policy optimization* (GRPO) that typically faces a trade-off between rollout efficiency and downstream performance. To improve efficiency while preserving performance, existing *selective rollout* methods mainly exploit *only source-side* training dynamics to optimize the discrete distribution over source prompts. However, in practice, the source training dataset is rarely identically distributed to the target test dataset. When such a mismatch exists, due to legal or privacy constraints, it is usually not acceptable to share target prompts themselves with the RLVR trainer. Nevertheless, it is still possible for the target data owner to provide limited aggregated feedback from a small held-out target set without revealing target prompts or rollout traces. In this paper, we propose *learning-to-reweight rollout* (L2RR) that explicitly incorporates target-side performance feedback into GRPO. Specifically, L2RR learns a discrete sampling distribution over source prompts by optimizing a bi-level objective that maximizes accuracy on a held-out target dataset, while simultaneously penalizing rollouts that provide little incremental learning information. Empirical studies on six math reasoning benchmarks and three model scales show that L2RR improves target-domain performance and increases rollout efficiency.

1 Introduction

Recent *large language models* (LLMs), such as OpenAI o1 [14] and DeepSeek R1 [8], have achieved strong reasoning performance across a variety of domains [12, 31], since they leverage increased inference-time computation for reasoning. A major catalyst behind this progress is *reinforcement learning with verifiable rewards* (RLVR), which optimizes policies from binary-valued verifiable feedback [17, 8]. *Group relative policy optimization* (GRPO) is a representative RLVR algorithm that trains via verifier-based rollouts, where a prompt is sampled, a group of responses is generated and scored, and the policy is updated using *group-normalized advantages* [33]. When group rewards collapse to the same value, the resulting advantages become near zero, so obtaining non-zero advantages often requires many costly rollouts. This creates a fundamental trade-off: fewer rollouts reduce cost but can degrade downstream performance, whereas better performance often requires more rollouts. Therefore, *improving rollout efficiency while preserving downstream performance* is critical for scaling RLVR training.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Existing methods improve rollout efficiency using training-side GRPO dynamics [43, 48]. Recent *selective rollout* methods leverage temporal regularities to either skip such prompts before expensive rollouts [48] or reuse previously successful rollouts to reduce repeated sampling [46]. These rollout-selection strategies rely on training-side information. Consequently, their efficiency gains may be limited by what the training dataset can reveal, and rollout-or-not decisions are tuned using training-side validation, in the hope that these choices transfer to downstream performance.

However, in practice, *the source training dataset is rarely identically distributed to the target test dataset*. Even in math reasoning, the training and test datasets can differ substantially in their difficulty or subdomains, with test problems often skewing toward harder or covering different topics [37]. As a result, source-side GRPO dynamics can yield rollout decisions that are misaligned with target performance, motivating the use of target information to learn selective rollout strategies.

Note that leveraging target-side information is non-trivial. In many real-world tasks, the model owner and the data owner are different entities, so target-side prompts and rollout traces can be inaccessible for RLVR training because of privacy or legal constraints [42, 1]. Without any target information, improvements beyond source-only training may be limited. Fortunately, in many real-world scenarios, although the trainer may not be allowed to access target prompts or target-side rollout traces, it is often still possible to access *limited target-side evaluation feedback* computed on a small held-out target dataset. Depending on the deployment protocol, this feedback may take the form of aggregated target-side gradients or directional derivatives, without directly sharing target prompts or rollout traces. For example, in hospitals, target prompts stay on-premise and training proceeds via fine-tuning with only model updates shared [1]; similarly, some benchmark-style workflows provide aggregate scores through a remote evaluator, without releasing test inputs during blind evaluation [9]. This motivates our core question: *can we leverage such limited target-side information to improve selective-rollout efficiency and target generalization?*

In this paper, we propose *learning-to-reweight rollout* (L2RR), a target-guided selective rollout scheme for GRPO that uses target-side evaluation feedback to learn how rollouts are allocated across the source prompt pool. Specifically, L2RR learns a discrete distribution for sampling over source prompts and alternates between GRPO policy updates and sampler updates. The GRPO policy updates are as usual. The sampler updates adjust prompt sampling probabilities using gradient-based feedback from the held-out target objective and the frequency of yielding near-zero group-normalized advantages. Benefiting from this design, L2RR jointly prioritizes target-relevant rollouts and avoids repeatedly spending resources on prompts that contribute little to learning progress, using only limited target-side information. We provide a theoretical analysis establishing an $\mathcal{O}(T^{-1/2})$ target-loss bound up to bias and tracking residuals under regularity assumptions and an appropriate choice of regularization weight, where T is the training horizon. Empirical studies on six reasoning benchmarks and three models spanning two parameter scales show that L2RR improves target performance while simultaneously increasing rollout efficiency.

Although *learning-to-reweight* [32][1] shares a similar idea and has been studied in supervised learning settings, extending this paradigm to RLVR is non-trivial. GRPO training dynamics are time-varying, and the advantage of a prompt can degenerate as group rewards collapse, so rollout effectiveness is required to be modeled using history-dependent information during training.

2 Preliminary

In this section, we first introduce the notation and problem setup, and then briefly review the GRPO training procedure, along with recent approaches for mitigating advantage collapse in GRPO.

Notations. Let $\mathcal{D}_{\mathrm{src}} = \{(\mathbf{x}_i^{\mathrm{S}}, a_i^{\mathrm{S}})\}_{i=1}^{N_S}$ be the source dataset and locally held validation set $\mathcal{D}_{\mathrm{tgt}} = \{(\mathbf{x}_i^{\mathrm{T}}, a_i^{\mathrm{T}})\}_{i=1}^{N_T}$ the target dataset, where $\mathbf{x}_i^{\mathrm{S}}$ and $\mathbf{x}_i^{\mathrm{T}}$ are prompts and a_i^{S} are the corresponding answers of $\mathbf{x}_i^{\mathrm{S}}$. We consider a training procedure of T rounds. For each prompt $\mathbf{x}$, at round $t \in \{1, \ldots, T\}$, the LLM policy $\pi_{\theta_t}(y \mid \mathbf{x})$ generates an output y. A verifier evaluates the output and returns a binary reward $r(\mathbf{x}, y, a) \in \{0, 1\}$, indicating whether the final answer matches the reference answer a.

Group relative policy optimization. In GRPO [33], at each round $t \in \{1, \ldots, T\}$, we sample a batch of prompts-answer pairs $\mathcal{B}_t$ of size B. For each sampled prompt $(\mathbf{x}^{\mathrm{S}}, a^{\mathrm{S}}) \in \mathcal{B}_t$, the current

[1] [35] further learned a weighting function that generalizes to new data, whereas our setting does not require this.

policy π_{θ_t} generates a group of G responses $\{o_g\}_{g=1}^{G}$. A verifier then assigns a binary reward to each response, yielding the reward set $R = \{r_g \mid r_g \in \{0,1\}\}_{g=1}^{G}$, where $r_g = r(\mathbf{x}^{\mathrm{S}}, o_g, a^{\mathrm{S}})$. Then, GRPO forms a group-relative advantage by normalizing rewards within the group:

$$A_g = \frac{r_g - \mathrm{mean}(R)}{\mathrm{std}(R) + \varepsilon},$$

where $\varepsilon > 0$ is a constant, $\mathrm{mean}(R)$ and $\mathrm{std}(R)$ denote the mean and standard deviation of R.

Following [43], we use the token-level GRPO loss and omit the Kullback-Leibler (KL) penalty term for each $(\mathbf{x}^{\mathrm{S}}, a^{\mathrm{S}})$, i.e.,

$$\ell\left(\theta_t, \theta_{t-1}; \mathbf{x}^{\mathrm{S}}, a^{\mathrm{S}}\right) = -\mathbb{E}_{\{o_g\}_{g=1}^{G} \sim \pi_{\theta_{t-1}}(\cdot|\mathbf{x}^{\mathrm{S}})} \frac{1}{\sum_{g=1}^{G}|o_g|} \sum_{g=1}^{G} \sum_{j=1}^{|o_g|} \min(\rho_{g,j} A_g,\, \bar{\rho}_{g,j} A_g)\,, \tag{1}$$

where $\bar{\rho}_{g,j} = \mathrm{clip}(\rho_{g,j},\, 1-\varepsilon_{\mathrm{low}},\, 1+\varepsilon_{\mathrm{high}})$, with $\mathrm{clip}(u,a,b) = \min\{\max\{u,a\},\, b\}$, and $\rho_{g,j} = \pi_{\theta_t}\big(o_{g,j} \mid \mathbf{x},\, o_{g,<j}\big) / \pi_{\theta_{t-1}}\big(o_{g,j} \mid \mathbf{x},\, o_{g,<j}\big)$.

Mitigating advantage collapse. In GRPO, the group-normalized advantages can collapse to zero when all responses in a group receive the same verifier reward, thereby eliminating the policy gradients. To mitigate this, [43] proposed dynamic sampling that filters out prompts with identical group rewards and resamples to fill the training batch, while [48] proposed a more sample-efficient method that maintains, for each $\mathbf{x}_i^{\mathrm{S}}$, a history of its rollout rewards. Let $n_i(t)$ denote the number of times $\mathbf{x}_i^{\mathrm{S}}$ has been sampled for rollout before round t. Its history is

$$\tau_{i,t} = \big((e_{i,1}, R_{i,1}), \ldots, (e_{i,n_i(t)}, R_{i,n_i(t)})\big),$$

where $e_{i,s}$ with $s \in \{1, \ldots, n_i(t)\}$ is the index of the training round in which $\mathbf{x}_i^{\mathrm{S}}$ is sampled for rollout for the s-th time, and $R_{i,s}$ denotes the set of verifier rewards from the responses generated on that occasion. An indicator $I_{i,s}$ and an online counter $z_{i,t}$ are then defined for each prompt $\mathbf{x}_i^{\mathrm{S}}$ as

$$I_{i,s} = \begin{cases} 1, & \text{if all rewards in } R_{i,s} \text{ are identical,} \\ 0, & \text{otherwise,} \end{cases}$$

and

$$z_{i,t} = \max\left\{ k \in \{0, \ldots, n_i(t)\} \;\middle|\; \prod_{s=n_i(t)-k+1}^{n_i(t)} I_{i,s} = 1 \right\},$$

where the empty product is defined as 1, so that $z_{i,t} = 0$ for prompts with no previous rollouts. The counter measures the number of most recent consecutive sampling events for $\mathbf{x}_i^{\mathrm{S}}$ that yield zero GRPO advantages in Eq. (1).

Following the filtering probability in [48], we assign each prompt $\mathbf{x}_i^{\mathrm{S}}$ a rollout-waste penalty that increases with persistently zero advantages:

$$1 - p_e^{z_{i,t}}, \tag{2}$$

where $p_e \in (0,1)$ controls how rapidly the penalty increases with zero-advantage rollouts.

3 Learning-to-Reweight Rollout (L2RR)

In this section, we present our proposed algorithm and theoretical analysis.

Formulation. Due to source-target datasets mismatch, a policy that is optimal for $\mathcal{D}_{\mathrm{src}}$ need not yield a policy that performs best on $\mathcal{D}_{\mathrm{tgt}}$. Inspired by importance reweighting for mitigating dataset mismatch [36, 32], we adapt this principle to RLVR to jointly improve target-domain generalization and rollout sample efficiency. Specifically, we maintain a time-varying sampling distribution $\mathbf{q}_t \in \Delta^{N_S}$ for $t \in [T]$, where $q_{t,i} > 0$ and $\sum_{i=1}^{N_S} q_{t,i} = 1$. At each round, we optimize the

$$L_{\mathrm{GRPO}}(\mathbf{q}_t) = \mathbb{E}_{(\mathbf{x}^{\mathrm{S}}, a^{\mathrm{S}}) \sim \mathbf{q}_t}\, \ell\left(\theta_t, \theta_{t-1}; \mathbf{x}^{\mathrm{S}}, a^{\mathrm{S}}\right).$$

The target owner evaluates the policy locally on the held-out target validation set, with the target objective defined as:

$$L_{\mathrm{tgt}}(\theta_t) = -\mathbb{E}_{\mathbf{x}\sim\mathcal{D}_{\mathrm{tgt}}}\mathbb{E}_{y\sim\pi_{\theta_t}(\cdot|\mathbf{x})}\big[r(\mathbf{x},y)\big].$$

Here, D_{tgt} is used to define the target-side local data, but its prompts are not exposed to the trainer.

This leads to the following objective function:

$$\begin{aligned}\min_{\mathbf{q}_1,\ldots,\mathbf{q}_T\in\Delta^{N_S}} \quad & \mathbb{E}[L_{\mathrm{tgt}}(\theta_{T+1})] \\ \text{s.t.} \quad & \mathcal{B}_t\sim\mathbf{q}_t, \quad \theta_{t+1}=\theta_t-\frac{\eta_\theta}{B}\sum_{i\in\mathcal{B}_t} g_{t,i}, \qquad t=1,\ldots,T,\end{aligned} \tag{3}$$

where θ_1 is the initial policy parameter, $g_{t,i}$ is the per-example GRPO gradient, and each $\mathbf{q}_t$ is chosen using the training history available before round t. The expectation is over prompt sampling and rollout generation.

Although we can upweight the sampling distribution on prompts whose rollouts most improve target held-out performance, those source prompts can still be either beyond the model's current capability or already solved so consistently that their GRPO advantages become degenerate. In both cases, repeatedly sampling them wastes rollout budget. To improve the sample efficiency, we use the rollout-waste penalty in Eq. (2), which penalizes the prompts that consecutively produce zero GRPO advantages. We quantify this history dependent rollout waste along this sampling rule by

$$L_{\mathrm{ws}}(\mathbf{q}_{1:T}) = \mathbb{E}\left[\frac{1}{TB}\sum_{t=1}^{T}\sum_{b=1}^{B}\Big(1-p_e^{\,z_{i_{t,b},t}}\Big)\right],$$

where $\mathbf{q}_{1:T}=(\mathbf{q}_1,\ldots,\mathbf{q}_T)$ and $i_{t,b}$ is the source-prompt index of the b-th sample in $\mathcal{B}_t$. Minimizing L_{ws} makes the sampler place less mass on prompts with large $z_{i,t}$, reducing the cost spent on uninformative rollouts.

Therefore, our learning objective is as follows:

$$\begin{aligned}\min_{\mathbf{q}_1,\ldots,\mathbf{q}_T\in\Delta^{N_S}} \quad & \mathbb{E}[L_{\mathrm{tgt}}(\theta_{T+1})] + \lambda L_{\mathrm{ws}}(\mathbf{q}_{1:T}) \\ \text{s.t.} \quad & \mathcal{B}_t\sim\mathbf{q}_t, \quad \theta_{t+1}=\theta_t-\frac{\eta_\theta}{B}\sum_{i\in\mathcal{B}_t} g_{t,i}, \qquad t=1,\ldots,T,\end{aligned} \tag{4}$$

where $\lambda\geq 0$ is the regularization hyperparameter.

Bi-level optimization task. The formulation in Eq. (4) motivates a bi-level optimization surrogate [2]. At iteration t, we first update the policy by one GRPO step on the batch $\mathcal{B}_t$ sampled from $\mathbf{q}_t$; following GRESO [48], prompts are drawn from $\mathbf{q}_t$ in sampling rounds (Table 4) until B prompts with non-degenerate rewards are collected, and all generated responses count toward the rollout cost:

$$\theta_{t+1} = \theta_t - \eta_\theta\cdot\frac{1}{B}\sum_{i\in\mathcal{B}_t} g_{t,i}, \tag{5}$$

where $g_{t,i}$ is the per-example GRPO gradient computed from the rollouts of prompt $\mathbf{x}_i^{\mathrm{S}}$ under π_{θ_t}.

After updating the policy, we update the sampling distribution $\mathbf{q}_t$. Since the dependence of $L_{\mathrm{tgt}}(\theta_{t+1})$ on $\mathbf{q}_t$ is implicit through discrete sampling, following the one-step unrolling surrogate used in importance weighting [32, 35], we form a differentiable look-ahead update on $\mathcal{B}_t$ with $\theta_{t+1}(\mathbf{q}_t) = \theta_t - \eta_\theta\sum_{i\in\mathcal{B}_t}\bar{q}_{t,i}\,g_{t,i}$, where $\bar{q}_{t,i}=q_{t,i}/\sum_{j\in\mathcal{B}_t}q_{t,j}$. Let $g_{t+1}^{\mathrm{tgt}} = \nabla_\theta L_{\mathrm{tgt}}(\theta)\big|_{\theta=\theta_{t+1}(\mathbf{q}_t)}$ and $\bar{g}_t = \sum_{i\in\mathcal{B}_t}\bar{q}_{t,i}\,g_{t,i}$. Treating $\mathcal{B}_t$, θ_t, and the per-example gradients as fixed, the one-step surrogate yields, for each $i\in\mathcal{B}_t$,

$$\widehat{g}_{t,i}^{\mathrm{tgt}} = \frac{\partial}{\partial q_{t,i}}L_{\mathrm{tgt}}\big(\theta_{t+1}(\mathbf{q}_t)\big) = -\eta_\theta\frac{\big\langle g_{t+1}^{\mathrm{tgt}},\, g_{t,i}-\bar{g}_t\big\rangle}{\sum_{j\in\mathcal{B}_t}q_{t,j}}. \tag{6}$$

The directional derivatives $\langle g_{t+1}^{\mathrm{tgt}}, g_{t,i}-\bar{g}_t\rangle$ can be computed by the target owner without sharing target prompts. We set $\widehat{g}_{t,i}^{\mathrm{tgt}}=0$ for $i\notin\mathcal{B}_t$. This interface provides target-data locality and restricted disclosure, while establishing formal privacy guarantees is left to future work.

For the vanishing-advantage regularizer, we use the online surrogate $L_{\mathrm{ws}}(\mathbf{q}_t) = \sum_{i=1}^{N_S} q_{t,i}\big(1 - p_e^{z_{i,t}}\big)$, whose coordinate gradient is $\widehat{g}_{t,i}^{\mathrm{ws}} = \lambda\big(1 - p_e^{z_{i,t}}\big)$ for $i = 1, \ldots, N_S$.

Combining $\widehat{g}_{t,i} = \widehat{g}_{t,i}^{\mathrm{tgt}} + \widehat{g}_{t,i}^{\mathrm{ws}}$, we update $\mathbf{q}_t$ via an exponentiated-gradient step:

$$q_{t+1,i} = \frac{q_{t,i} \exp\big(-\eta_q\, \widehat{g}_{t,i}\big)}{\sum_{j=1}^{N_S} q_{t,j} \exp\big(-\eta_q\, \widehat{g}_{t,j}\big)}. \tag{7}$$

From sampler optimization to rollout allocation. The sampler update relies on a target-guided gradient estimate, whose accuracy depends on the gradients $\{g_{t,i}\}_{\mathbf{x}_i \in \mathcal{B}_t}$ and is thus affected by the per-prompt rollout count $G_{i,t}$. In Section 4, our theoretical analysis makes this dependence explicit: lowering the variance of the target-guided gradient estimator tightens the target-loss bound, with other terms held fixed. Therefore, *the bound motivates our rollout allocation*: with a given per-round rollout budget, we allocate more rollouts to prompts that exert larger influence on the target-guided sampler gradient update, and fewer rollouts to prompts that repeatedly yield degenerate GRPO advantages.

We first define the time-varying rollout budget based on the history-dependent informativeness scores. For each prompt $\mathbf{x}_i \in \mathcal{B}_t$, let $s_{i,t} = p_e^{z_{i,t}}$, and set $\bar{s}_t = \frac{1}{B}\sum_{i \in \mathcal{B}_t} s_{i,t}$. The rollout budget is defined as $\mathrm{RO}_t = BG \cdot \mathrm{clip}\big(\bar{s}_t, \rho_{\min}, 1\big)$, where $\rho_{\min} \in [G_{\min}/G, 1]$ enforces a minimum rollout budget.

Then, at iteration t, we use the rollout budget RO_t and optimize $\{G_{i,t}\}_{i \in \mathcal{B}_t}$ to reduce the estimation error of the outer-gradient used to update $\mathbf{q}_t$. Assume that the variance of the per-prompt outer-gradient contribution decays inversely with the group size, e.g., $\mathrm{Var}\big(\widehat{g}_{t,i}^{\mathrm{tgt}}\big) \le \frac{c_{i,t}}{G_{i,t}}$, where $c_{i,t} \ge 0$ is an iteration-dependent variance proxy for rollout allocation. Under this model, we use a weighted allocation proxy to allocate more rollouts to prompts that contribute more to the sampler update and are not persistently degenerate. Specifically, we assign each prompt a weight $w_{i,t} = p_e^{z_{i,t}}\, u_{i,t}$, $u_{i,t} = \Big|\big\langle g_{t+1}^{\mathrm{tgt}},\, g_{t,i} - \bar{g}_t \big\rangle\Big|$, where $u_{i,t}$ measures the magnitude of the contribution of prompt $\mathbf{x}_i^{\mathrm{S}}$ to the target-guided sampler gradient, and $p_e^{z_{i,t}}$ downweights prompts with large waste counters.

This leads to the following rollout budget allocation problem:

$$\min_{\{G_{i,t}\}} \sum_{i \in \mathcal{B}_t} \frac{w_{i,t}\, c_{i,t}}{G_{i,t}}$$
$$\text{s.t.} \quad \sum_{i \in \mathcal{B}_t} G_{i,t} \le \mathrm{RO}_t, \quad G_{\min} \le G_{i,t} \le G.$$

For positive weights and proxies, the continuous optimum satisfies $G_{i,t} \propto \sqrt{w_{i,t} c_{i,t}}$ when no bounds are active. In our implementation, we set the proxy to $c_{i,t} = w_{i,t}^2$, so that $\sqrt{w_{i,t} c_{i,t}} = w_{i,t}^{3/2}$. Letting $\beta_{i,t} = \sqrt{w_{i,t}\, c_{i,t}}$ and normalizing within $\mathcal{B}_t$ as $\bar{\beta}_{i,t} = \beta_{i,t} / \sum_{j \in \mathcal{B}_t} \beta_{j,t}$, we allocate

$$G_{i,t} = \mathrm{clip}\big(\lfloor \mathrm{RO}_t \cdot \bar{\beta}_{i,t} \rfloor,\; G_{\min},\; G\big). \tag{8}$$

This allocation aims to reduce the variance of the target-guided gradient estimator used in the $\mathbf{q}_t$ update, motivated by the noise dependence of our regret bound in Theorem 1. We summarize our method in Algorithm 1.

4 Theoretical Analysis

In this section, we provide a standard online learning analysis for the sampler update in L2RR. The analysis views the sampling distribution update as an exponentiated-gradient step on the simplex and derives a sublinear regret bound for the linear losses induced by its feedback [10]. We also quantify the effect of the rollout-waste penalty and distinguish this regret guarantee from the conditional target-loss analysis in **Appendix D**.

Let $\widehat{\ell}_t(\mathbf{q}) = \langle \widehat{g}_t, \mathbf{q} \rangle$, where $\widehat{g}_t$ is the feedback in Eq. (7). We compare the sampler with the best fixed distribution on the same realized loss sequence:

$$\widehat{R}_T = \frac{1}{T}\left(\sum_{t=1}^{T} \widehat{\ell}_t(\mathbf{q}_t) - \min_{\mathbf{q} \in \Delta^{N_S}} \sum_{t=1}^{T} \widehat{\ell}_t(\mathbf{q})\right). \tag{9}$$

Algorithm 1 Learning-to-Reweight Rollouts (L2RR)

Input: Source dataset $\mathcal{D}_{\mathrm{src}}$, target-side feedback oracle $\mathcal{O}_{\mathrm{tgt}}$, weight λ, iterations T, batch size B, group size threshold $G_{\min}$ and G, learning rate η_θ and η_q, policy π_θ, verifier r

Initialize $\mathbf{q}_1$ as uniform distribution, θ_1, and $z_{i,1} = 0$.
for $t = 1, \ldots, T$ **do**
  Sample mini-batch $\mathcal{B}_t = \{\mathbf{x}^{\mathrm{S}}_{i_1}, \ldots, \mathbf{x}^{\mathrm{S}}_{i_B}\} \sim \mathbf{q}_t$
  Set the rollout number $G_{i,t}$ according to Eq. (8)
  Sample $G_{i,t}$ rollouts $\{o_{i,g}\}_{g=1}^{G_{i,t}}$ for $\mathbf{x}^{\mathrm{S}}_i \in \mathcal{B}_t$ and compute rewards $r_{i,g} = r(\mathbf{x}^{\mathrm{S}}_i, o_{i,g}, a^{\mathrm{S}})$
  Update policy θ_{t+1} according to Eq. (5)
  Update sampler $\mathbf{q}_{t+1}$ according to Eq. (7)
end for
Output policy $\pi_{\theta_{T+1}}$.

Theorem 1 (Sampler regret)**.** *Initialize* $\mathbf{q}_1$ *uniformly and assume* $T^{-1}\sum_{t=1}^{T}\mathbb{E}\|\widehat{g}_t\|_\infty^2 \le H^2$*, with* H *independent of* T*. For any fixed* $a > 0$*, use* $\eta_q = a/\sqrt{T}$ *throughout the run. Then*

$$\mathbb{E}\left[\widehat{R}_T\right] \le \frac{\log N_S/a + aH^2/2}{\sqrt{T}}. \tag{10}$$

Let $\widehat{R}_T^{\mathrm{tgt}}$ *denote the same regret with* $\widehat{\ell}_t^{\mathrm{tgt}}(\mathbf{q}) = \langle \widehat{g}_t^{\mathrm{tgt}}, \mathbf{q}\rangle$*. For a fixed per-run weight* $\lambda_T \ge 0$*,*

$$\mathbb{E}\left[\widehat{R}_T^{\mathrm{tgt}}\right] \le \frac{\log N_S/a + aH^2/2}{\sqrt{T}} + \lambda_T. \tag{11}$$

Thus the cumulative feedback regret is $\mathcal{O}(\sqrt{T})$ and its average is $\mathcal{O}(T^{-1/2})$. Choosing $\lambda_T \le c_\lambda/\sqrt{T}$, held fixed within each run, gives the same rates for the target-feedback losses. The second-moment term includes gradient-estimation noise, connecting the bound to rollout allocation without assuming that the weighted proxy minimizes that noise.

5 Experiments

In this section, we evaluate the proposed algorithm on multiple benchmarks with three different models. We also conduct ablation studies on our approach.

5.1 Experimental Setup

Models. We test L2RR on Qwen2.5-Math-1.5B [41], DeepSeek-R1-Distill-Qwen-1.5B [8], and Qwen2.5-Math-7B [41]. For Qwen2.5-Math-1.5B/7B models, we use 4096 as the context length, as it is the maximum context length for those two models. For DeepSeek-R1-Distill-Qwen-1.5B, we set the context length to 8192 (2048 prompt and 6144 response tokens), following the experimental settings in the work of [48].

Datasets. For source datasets, we consider: I). DAPO+MATH: we combine the DAPO dataset [43], which contains only integer solutions, with the MATH dataset [12], which contains LaTeX-formatted solutions. II). OPEN-R1 30k subset (R1): A 30,000-example subset of the OPEN-R1 math dataset[2].

For target datasets, we use six widely used mathematical reasoning benchmarks to evaluate the performance: Math500 [12], AIME24[3], AMC[4], Minerva Math [18], Gaokao [45], Olympiad Bench [11].

Setups. Following [48], we implement L2RR using verl [34] and vLLM [16] for rollouts. We use Math-Verify[5] to verify the generated answers. Hyperparameters are listed in **Appendix A**. Each target benchmark is partitioned into fixed, disjoint 10% validation and 90% test splits shared across all methods and five training seeds. L2RR uses validation data for sampler learning; all methods

[2] https://huggingface.co/datasets/open-r1/OpenR1-Math-220k
[3] https://huggingface.co/datasets/Maxwell-Jia/AIME_2024
[4] https://huggingface.co/datasets/AI-MO/aimo-validation-amc
[5] https://github.com/huggingface/Math-Verify

Table 1: Performance (%) comparison across six math reasoning benchmarks. We train three models on source datasets DAPO+MATH (DM) and the Open R1 subset (OR1) and test on six benchmarks. # Rollout denotes cumulative source responses, in millions. RO ratio is the DAPO count divided by the method count. Accuracies are mean ± standard deviation over five training seeds.

Dataset	Method	Math500	AIME24	AMC	Gaokao	Miner.	Olymp.	Avg.	# Rollout	RO ratio
Qwen2.5-Math-1.5B										
DM	DAPO	76.3± 1.8	15.2± 3.1	60.9± 2.7	63.8± 2.3	31.6± 2.5	38.2± 1.2	47.7± 1.5	7.6M	1.0×
	GRESO	75.8± 1.8	14.1± 3.0	60.8± 2.8	65.5± 2.1	32.8± 2.8	38.0± 1.5	47.8± 1.6	3.3M	2.3×
	AR3PO	77.6± 1.3	15.7± 3.1	60.8± 2.5	65.3± 2.3	**35.4**± 2.5	38.6± 1.9	48.9± 1.5	2.5M	3.0×
	L2RR	**79.4**± 1.7	**17.2**± 2.4	**61.8**± 2.1	**66.8**± 2.1	35.2± 2.1	**41.4**± 0.9	**50.3**± 1.3	3.0M	2.5×
OR1	DAPO	76.5± 1.9	15.7± 2.7	49.5± 2.8	65.1± 2.3	30.2± 2.6	39.2± 1.7	46.0± 1.6	3.8M	1.0×
	GRESO	75.8± 2.1	18.5± 2.8	49.7± 2.5	64.9± 2.1	29.5± 2.5	39.0± 0.8	46.2± 1.5	1.6M	2.4×
	AR3PO	76.2± 1.8	18.3± 3.2	49.8± 2.5	64.7± 2.1	29.7± 2.5	39.2± 1.1	46.3± 1.5	1.4M	2.7×
	L2RR	**78.3**± 1.5	**19.8**± 2.7	**51.1**± 2.3	**66.2**± 1.7	**30.6**± 1.8	**41.5**± 1.1	**47.9**± 1.3	1.4M	2.7×
DeepSeek-R1-Distill-Qwen-1.5B										
DM	DAPO	85.9± 2.1	34.8± 3.3	70.5± 2.9	78.1± 2.3	34.4± 2.8	54.3± 1.8	59.7± 1.7	2.4M	1.0×
	GRESO	85.7± 2.3	34.3± 3.1	70.1± 3.1	77.5± 2.6	33.1± 2.6	53.1± 2.3	59.0± 1.8	1.6M	1.5×
	AR3PO	86.4± 1.9	34.6± 3.6	70.2± 3.5	77.9± 2.4	33.9± 2.6	53.9± 1.8	59.5± 1.8	1.7M	1.4×
	L2RR	**88.2**± 1.7	**36.7**± 2.9	**71.8**± 2.5	**79.5**± 2.3	**35.3**± 2.3	**56.8**± 1.7	**61.4**± 1.5	1.6M	1.5×
OR1	DAPO	84.8± 2.5	22.2± 2.9	**68.1**± 2.7	73.2± 2.7	33.4± 2.9	52.4± 1.9	55.7± 1.8	2.4M	1.0×
	GRESO	85.6± 2.3	24.1± 3.3	66.3± 2.8	74.4± 2.3	32.9± 2.5	54.3± 1.9	56.3± 1.7	1.5M	1.6×
	AR3PO	85.1± 2.2	23.7± 3.5	66.8± 2.5	74.2± 2.2	33.0± 2.5	53.8± 1.9	56.1± 1.7	1.6M	1.5×
	L2RR	**86.7**± 1.8	**25.9**± 2.8	67.7± 2.6	**76.1**± 2.1	**34.8**± 2.3	**57.4**± 1.4	**58.1**± 1.5	1.5M	1.6×
Qwen2.5-Math-7B										
DM	DAPO	82.3± 2.1	**33.3**± 3.1	77.8± 2.5	72.3± 2.1	34.9± 2.3	45.1± 1.7	57.6± 1.6	13.1M	1.0×
	GRESO	81.4± 2.1	30.6± 3.3	79.5± 2.5	70.5± 2.3	34.6± 2.4	45.4± 1.9	57.0± 1.6	6.3M	2.1×
	AR3PO	82.1± 2.2	31.5± 2.8	78.5± 3.3	71.9± 2.3	35.1± 2.7	44.7± 1.9	57.3± 1.7	6.6M	2.0×
	L2RR	**83.5**± 2.1	32.4± 2.7	**80.4**± 2.6	**72.8**± 2.2	**36.3**± 2.4	**47.5**± 1.9	**58.8**± 1.6	6.0M	2.2×
OR1	DAPO	82.2± 1.9	32.0± 2.7	64.3± 2.5	68.1± 2.5	34.7± 2.3	47.5± 1.8	54.8± 1.5	11.4M	1.0×
	GRESO	81.7± 2.3	33.1± 3.0	65.8± 2.4	68.0± 2.4	35.3± 2.5	47.3± 2.1	55.2± 1.7	3.4M	3.4×
	AR3PO	82.0± 2.6	33.5± 3.1	65.9± 2.8	67.5± 2.8	36.5± 2.8	46.9± 1.8	55.4± 1.8	3.6M	3.2×
	L2RR	**83.1**± 2.1	**35.0**± 2.6	**67.2**± 2.3	**69.3**± 2.3	**37.2**± 2.3	**48.9**± 1.7	**56.8**± 1.5	3.3M	3.5×

use it for checkpoint selection. Following [38, 48], we evaluate checkpoints on the validation splits every 50 steps and report test performance for the checkpoint with the best macro-average validation accuracy across the six benchmarks.

We also evaluate source-only baselines without target-validation-based checkpoint selection and Target-Mix baselines that directly incorporate target validation examples into policy training, requiring target-data access beyond our setting. Results are reported in **Appendix B**.

Contenders. We compare the proposed algorithm with three strong contenders that consider rollout efficiency. The decoupled clip and dynamic sampling policy optimization (DAPO) mitigates advantage collapse by dynamically resampling prompts to ensure non-degenerate verifier-reward variance in each GRPO group [43]. GRPO with efficient selective rollout (GRESO) reduces rollout cost via online pre-rollout filtering that predicts and skips prompts likely to produce degenerate near-zero GRPO updates [48]. Adaptive rollout and response reuse policy optimization (AR3PO) improves sampling efficiency by adaptive rollout allocation and response reuse, with mechanisms to control the mismatch between reused responses and the current policy [46].

5.2 Main Results

Table 1 reports the comparison results of the proposed algorithm with three state-of-the-art contenders across six math reasoning benchmarks, three models spanning two parameter scales, and two source training sets. We report the checkpoint with the best macro-average validation accuracy, following the setting in previous work [48, 46]. Across all settings, L2RR achieves the highest average performance while consistently reducing the rollout cost relative to DAPO. On Qwen2.5-Math-1.5B trained on DM, L2RR improves the average from 47.7% to 50.3% while reducing rollouts from 7.6M to 3.0M. These results suggest that leveraging limited target-side feedback to learn the source-prompt sampling distribution produces more cost-effective rollouts and improves generalization to the target domain.

GRESO and AR3PO reduce rollouts relative to DAPO by exploiting only source-side rollout statistics, with averages close to DAPO in most configurations. In contrast, L2RR more consistently improves

Table 2: Ablation study on key components of L2RR. We report performance (%) on six math reasoning benchmarks for *Qwen2.5-Math-1.5B* trained on the DAPO + MATH dataset (DM). # Rollout denotes cumulative source responses, in millions. RO ratio is the DAPO count divided by the method count.

Method	Math500	AIME24	AMC	Gaokao	Miner.	Olymp.	Avg.	# Rollout	RO ratio
L2RR full	**79.9**	**18.5**	**62.2**	**67.3**	**35.7**	**41.5**	**50.9**	3.0M	2.5×
L2RR w/o ws	78.4	16.7	61.8	66.0	34.0	40.0	49.5	2.7M	2.8×
L2RR w/o rollout allo	78.2	16.7	61.5	65.7	33.8	39.8	49.3	4.3M	1.8×
L2RR uniform q	77.9	15.7	61.8	65.2	33.0	39.5	48.9	3.2M	2.4×

Table 3: Effect of held-out target set size for sampler learning. Accuracy (%) on six reasoning benchmarks for Qwen2.5-Math-1.5B trained on DAPO + MATH. We vary the size of held-out target dataset to update $\mathbf{q}_t$.

# Target (held-out)	Math500	AIME24	AMC	Gaokao	Miner.	Olymp.	Avg.
4%	78.2	16.7	61.5	65.7	34.4	39.5	49.3
6%	78.5	17.6	61.8	66.0	34.7	39.8	49.7
8%	78.9	17.6	61.5	66.3	34.9	40.1	49.9
10%	79.9	18.5	62.2	67.3	35.7	41.5	50.9

average accuracy while maintaining comparable rollout budgets to these baselines, by exploiting a few target domain information. For example, on Qwen2.5-Math-7B trained on DM, L2RR reaches 58.8% average with 6.0M rollouts, outperforming GRESO and AR3PO. Overall, L2RR achieves a stronger accuracy and rollout trade-off by explicitly steering rollout selection, rather than relying solely on source-side dataset and training dynamic.

5.3 Ablation Studies

In this part, we use Qwen2.5-Math-1.5B trained on the DAPO + MATH dataset to test in detail how L2RR improves the performance and reduces rollout overhead in the training process.

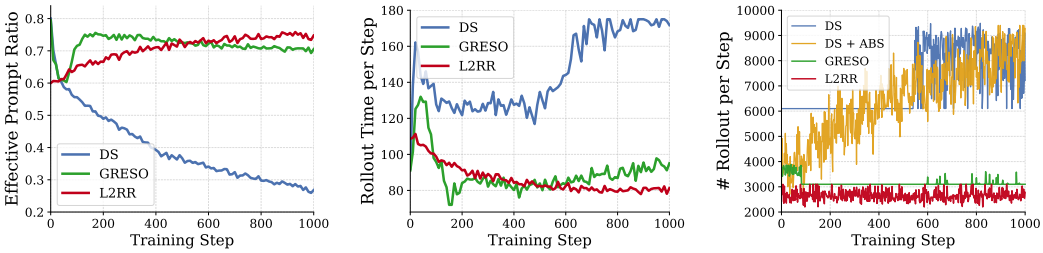
Contribution of each component. We first conduct an ablation study on the contribution of each component in our algorithm in Table 2. We compare the full L2RR with three ablated variants: L2RR W/O WS (setting $\lambda = 0$ to remove the rollout-waste regularizer), L2RR W/O ROLLOUT ALLOCATION (using a fixed group size $G_{i,t} \equiv G$), and L2RR UNIFORM q (fixing the sampler to be uniform and disabling distribution learning). Overall, the results show that each component plays an important role: removing the rollout-waste term lowers the average accuracy (50.9% $\rightarrow$ 49.5%) at a similar rollout count (2.7M vs. 3.0M), removing rollout allocation lowers the RO ratio (2.5× $\rightarrow$ 1.8×) by losing fine-grained budget reallocation within each batch, and fixing $\mathbf{q}$ to uniform leads to the most consistent drop in target accuracy, highlighting the importance of target-guided sampler learning for handling source–target mismatch. In particular, the full L2RR achieves the best average accuracy while maintaining favorable rollout counts, supporting the contribution of the combined components to target performance and rollout efficiency.

Number of held-out target examples. We further study how the amount of held-out target data affects L2RR. Keeping the source training setup fixed, we vary the size of the held-out target dataset used to compute the target-side evaluation objective for updating the sampler q_t. For each size, we uniformly sub-sample the held-out target set, run L2RR with the same training budget, and report the checkpoint with the best average accuracy across the six benchmarks. Table 3 shows that target accuracy increases monotonically with the held-out target size (49.3% at 4% to 50.9% at 10%); L2RR already benefits from a small held-out slice, and these single runs do not indicate saturation within the range studied.

Dynamics of training process. To better understand where the proposed algorithm gains efficiency, we analyze its training dynamics in Figure 1. Overall, L2RR improves both rollout efficiency and target accuracy by coupling target-guided sampler learning with waste-aware rollout allocation.

Figure 1(a) reports the effective-prompt ratio over training steps, where an "effective" prompt is one whose sampled group does not collapse to a degenerate outcome and thus can contribute non-trivial GRPO updates. After roughly the first 40 steps, L2RR keeps a much higher effective ratio than DS; GRESO is higher early on, but L2RR overtakes it after about 550 steps and keeps rising, suggesting that the learned sampling distribution increasingly prioritizes prompts that are both target-relevant and

Figure 1: Training-dynamics illustration for L2RR. During training, L2RR increases sampling on prompts that contribute more to improving held-out target accuracy and suppresses repeatedly wasteful prompts as indicated by the history-based counter. Qwen2.5-Math-1.5B trained on DM. DS denotes DAPO with dynamic sampling; DS + ABS adds the adaptive sampling batch size of GRESO [48].

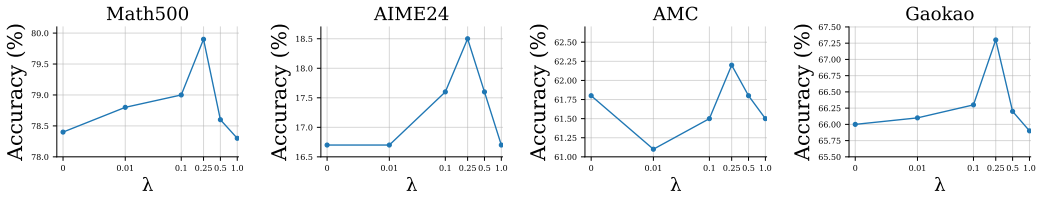


(a) Effective prompt ratio per training step.

(b) Rollout time per training step (seconds).

(c) Rollouts per step under dynamic rollout allocation.

Figure 2: Effect of the rollout-waste weight λ on target accuracy (%). Overall, moderate λ yields the best trade-off, while overly small or large λ can under-regularize wasted rollouts or over-penalize informative prompts and thus reduce accuracy.



less likely to yield repeatedly degenerate behavior, while the rollout-waste regularization discourages spending compute on persistently unproductive prompts.

Figure 1(b) shows rollout time per step. Since L2RR reduces repeated sampling of prompts that historically contribute little training gradients and reallocates effort toward prompts that better improve held-out target accuracy, its rollout time per step stays well below that of DS and, after about 600 steps, is also lower and more stable than that of GRESO, indicating improved utilization of the rollout budget. Accumulating the per-step times in Figure 1(b) over 1000 steps, L2RR needs the least total rollout time (about 24 hours, vs. about 25 hours for GRESO and 41 hours for DS).

Finally, Figure 1(c) visualizes the number of rollouts per step under variable rollout allocation. Unlike methods that use a fixed group size for every sampled prompt, L2RR dynamically assigns fewer responses to prompts with large waste counters and more responses to prompts that more strongly affect the target-guided sampler update, thereby reducing rollouts per step while maintaining or improving the downstream target accuracy. This behavior is consistent with our theoretical view that improving the accuracy of the target-driven sampler gradient reduces the effective outer-gradient noise, leading to more stable sampler updates and better target-domain generalization.

Effect of the rollout-waste weight λ. We also study how the rollout-waste regularizer weight λ in Eq. (4) affects target performance. Intuitively, increasing λ strengthens the preference for avoiding prompts that repeatedly induce degenerate GRPO updates, which can improve rollout efficiency but may also reduce exploration that the hard prompt may have positive advantages; setting λ too small makes the sampler update rely mainly on the target-guided term and can spend more rollouts on prompts that are unproductive.

Figure 2 reports accuracy trends on four representative benchmarks as λ varies over $\{0, 0.01, 0.1, 0.25, 0.5, 1.0\}$. Across datasets, we observe a consistent pattern: moderate values of λ tend to preserve (or improve) accuracy relative to $\lambda = 0$, whereas overly large λ can dampen learning on some datasets by aggressively down-weighting a broad set of prompts. Based on these, we adopt a moderate value of λ.

6 Conclusion

In this paper, we study source-target datasets mismatch in sample-efficient RLVR, where the source training distribution differs from the target evaluation distribution. We focus on a setting in which the trainer cannot access target prompts or rollout traces, but can obtain aggregated gradient-based feedback from a small held-out target validation split. We propose *learning-to-reweight rollouts* (L2RR), which learns a sampling distribution over source prompts to jointly improve target-domain generalization and rollout efficiency in GRPO training. Specifically, L2RR updates the sampling distribution using exponentiated-gradient steps guided by target-gradient alignment and a rollout-waste regularizer, and uses the resulting sampler to drive GRPO policy updates. Our theoretical analysis provides a sublinear regret bound for the sampler and a conditional target-loss bound for the policy, supporting the proposed sampling-distribution learning mechanism. Experiments on six widely-used RLVR benchmarks and three models show that L2RR improves target generalization while increasing rollout efficiency over strong contenders.

Acknowledgments

Zhen-Yu Zhang was supported by JSPS KAKENHI Grant Number JP25K21282. MS was supported by JST ASPIRE Grant Number JPMJAP2405. This research used computational resources of the supercomputer RIKYU provided by the RIKEN Center for Computational Science.

References

[1] Jiamu Bai, Daoyuan Chen, Bingchen Qian, Liuyi Yao, and Yaliang Li. Federated fine-tuning of large language models under heterogeneous tasks and client resources. *Advances in Neural Information Processing Systems (NeurIPS)*, 37:14457–14483, 2024.

[2] Benoît Colson, Patrice Marcotte, and Gilles Savard. An overview of bilevel optimization. *Annals of operations research*, 153(1):235–256, 2007.

[3] Wen-Bo Du, Tian Qin, Tian-Zuo Wang, and Zhi-Hua Zhou. Avoiding undesired future with minimal cost in non-stationary environments. *Advances in Neural Information Processing Systems (NeurIPS)*, 37:135741–135769, 2024.

[4] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research (JMLR)*, 12(7), 2011.

[5] Mehdi Fatemi, Banafsheh Rafiee, Mingjie Tang, and Kartik Talamadupula. Concise reasoning via reinforcement learning. *arXiv preprint arXiv:2504.05185*, 2025.

[6] Ilyas Fatkhullin and Niao He. Taming nonconvex stochastic mirror descent with general Bregman divergence. In *Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 3493–3501, 2024.

[7] Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 1568–1577, 2018.

[8] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[9] Isabelle Guyon, Lisheng Sun-Hosoya, Marc Boullé, Hugo Jair Escalante, Sergio Escalera, Zhengying Liu, Damir Jajetic, Bisakha Ray, Mehreen Saeed, Michèle Sebag, et al. Analysis of the AutoML challenge series. *Automated Machine Learning*, 177:177–219, 2019.

[10] Elad Hazan. *Introduction to Online Convex Optimization*. The MIT Press, 2 edition, 2022.

[11] Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings*

of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL), pages 3828–3850, 2024.

[12] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[13] Hamish Ivison, Muru Zhang, Faeze Brahman, Pang Wei Koh, and Pradeep Dasigi. Large-scale data selection for instruction tuning. *arXiv preprint arXiv:2503.01807*, 2025.

[14] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

[15] Prashant Khanduri, Siliang Zeng, Mingyi Hong, Hoi-To Wai, Zhaoran Wang, and Zhuoran Yang. A near-optimal algorithm for stochastic bilevel optimization via double-momentum. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:30271–30283, 2021.

[16] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626, 2023.

[17] Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

[18] Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:3843–3857, 2022.

[19] Shuozhe Li, Vaishnav Tadiparthi, Kwonjoon Lee, Nakul Agarwal, Hossein Nourkhiz Mahjoub, Ehsan Moradi Pari, Lizhang Chen, Amy Zhang, and Liu Leqi. Learning robust reasoning through guided adversarial self-play. *arXiv preprint arXiv:2602.00173*, 2026.

[20] Xuefeng Li, Haoyang Zou, and Pengfei Liu. Limr: Less is more for rl scaling. *arXiv preprint arXiv:2502.11886*, 2025.

[21] Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, Thomas Hubert, Peter Choy, Cyprien de Masson d'Autume, Igor Babuschkin, Xinyun Chen, Po-Sen Huang, Johannes Welbl, Sven Gowal, Alexey Cherepanov, James Molloy, Daniel J. Mankowitz, Esme Sutherland Robson, Pushmeet Kohli, Nando de Freitas, Koray Kavukcuoglu, and Oriol Vinyals. Competition-level code generation with AlphaCode. *Science*, 2022.

[22] Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation. *Advances in Neural Information Processing Systems (NeurIPS)*, 36:21558–21572, 2023.

[23] Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

[24] Dougal Maclaurin, David Duvenaud, and Ryan Adams. Gradient-based hyperparameter optimization through reversible learning. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pages 2113–2122, 2015.

[25] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 20286–20332, 2025.

[26] William Muldrew, Peter Hayes, Mingtian Zhang, and David Barber. Active preference learning for large language models. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, pages 36577–36590, 2024.

[27] Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.

[28] Khai Nguyen, Hai Nguyen, Tuan Pham, and Nhat Ho. Lightspeed geometric dataset distance via sliced optimal transport. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, pages 46162–46177, 2025.

[29] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:27730–27744, 2022.

[30] Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. MAUVE: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:4816–4828, 2021.

[31] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *Proceedings of the 1st Conference on Language Modeling*, 2024.

[32] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 4334–4343, 2018.

[33] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[34] Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the 20th European Conference on Computer Systems*, pages 1279–1297, 2025.

[35] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. *Advances in Neural Information Processing Systems (NeurIPS)*, 32:1917–1928, 2019.

[36] Masashi Sugiyama, Matthias Krauledat, and Klaus-Robert Müller. Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research (JMLR)*, 8(5):985–1005, 2007.

[37] Yiming Wang, Pei Zhang, Baosong Yang, Derek Wong, Zhuosheng Zhang, and Rui Wang. Embedding trajectory for out-of-distribution detection in mathematical reasoning. *Advances in Neural Information Processing Systems (NeurIPS)*, 37:42965–42999, 2024.

[38] Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Liyuan Liu, Baolin Peng, Hao Cheng, Xuehai He, Kuan Wang, Jianfeng Gao, et al. Reinforcement learning for reasoning in large language models with one training example. *arXiv preprint arXiv:2504.20571*, 2025.

[39] Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. Less: selecting influential data for targeted instruction tuning. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, pages 54104–54132, 2024.

[40] Wei Xiong, Jiarui Yao, Yuhui Xu, Bo Pang, Lei Wang, Doyen Sahoo, Junnan Li, Nan Jiang, Tong Zhang, Caiming Xiong, et al. A minimalist approach to llm reasoning: from rejection sampling to reinforce. *arXiv preprint arXiv:2504.11343*, 2025.

[41] An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

[42] Rui Ye, Rui Ge, Xinyu Zhu, Jingyi Chai, Du Yaxin, Yang Liu, Yanfeng Wang, and Siheng Chen. Fedllm-bench: Realistic benchmarks for federated learning of large language models. *Advances in Neural Information Processing Systems (NeurIPS)*, 37:111106–111130, 2024.

[43] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

[44] Yufeng Yuan, Yu Yue, Ruofei Zhu, Tiantian Fan, and Lin Yan. What's behind ppo's collapse in long-cot? value optimization holds the secret. *arXiv preprint arXiv:2503.01491*, 2025.

[45] Xiaotian Zhang, Chunyang Li, Yi Zong, Zhengyu Ying, Liang He, and Xipeng Qiu. Evaluating the performance of large language models on gaokao benchmark. *arXiv preprint arXiv:2305.12474*, 2023.

[46] Yuheng Zhang, Wenlin Yao, Changlong Yu, Yao Liu, Qingyu Yin, Bing Yin, Hyokun Yun, and Lihong Li. Improving sampling efficiency in rlvr through adaptive rollout and response reuse. *arXiv preprint arXiv:2509.25808*, 2025.

[47] Zheyu Zhang, Shuo Yang, and Gjergji Kasneci. Consolidating rewarded perturbations for llm post-training. *arXiv preprint arXiv:2605.31494*, 2026.

[48] Haizhong Zheng, Yang Zhou, Brian R Bartoldson, Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi Chen. Act only when it pays: Efficient reinforcement learning for llm reasoning via selective rollouts. *Advances in Neural Information Processing Systems (NeurIPS)*, 38:in press, 2025.

Appendix

A Detailed Experimental Setting

Table 4 lists the hyperparameters. Shared GRPO settings follow [48] and are identical for all methods; contenders use their published method-specific settings.

Table 4: Hyperparameters. Batch sizes count prompts.

Training	
Framework	verl (FSDP) with vLLM rollouts
Optimizer	AdamW, $\beta = (0.9, 0.999)$, weight decay 0.01
Learning rate η_θ	1×10^{-6}, constant
Training steps	1000
Batch size B	256 (DeepSeek-1.5B: 128)
Prompts per sampling round	384 (DeepSeek-1.5B: 192)
PPO mini-batch size	64 (4 updates per step; DeepSeek-1.5B: 2)
Group size G	8
Loss	token-level GRPO, $\varepsilon_{\text{low}} = \varepsilon_{\text{high}} = 0.2$, entropy coefficient 0.001, no KL
Max prompt / response length	Qwen2.5-Math: 1536 / 2560; DeepSeek-1.5B: 2048 / 6144
Rollout temperature	1.0
Reward	binary: 1 if correct, 0 otherwise (Math-Verify)
Evaluation	
Frequency	every 50 steps on the validation splits
Decoding	temperature 1.0, top-p 0.7, avg@4
Checkpoint selection	best macro-average validation accuracy
Validation / test split	10% / 90% of each benchmark
L2RR	
Rollout-waste weight λ	0.25
Sampler step size η_q	1.0
Waste decay p_e	0.5
Minimum group size $G_{\min}$	4
Initial sampler $\mathbf{q}_1$	uniform
Sampler floor	mixing with uniform, floor 0.02; cap $10\times$ uniform
Target-gradient batch	64 validation prompts $\times$ 8 rollouts, every step
Target-term normalization	RMS, clipped at 2.0
Compute	
Hardware	2 NVIDIA GB200 GPUs

B More Empirical Results

In this section, we provide detailed empirical results omitted from the main paper due to space limits.

B.1 Comparison with Target-Aware Baselines.

To further ensure a fair comparison, we allow DAPO, GRESO, and AR3PO to use the same held-out target data as L2RR. For each contender, we evaluate checkpoints every 50 steps during training and use the held-out target set for target-validation-based checkpoint selection. As shown in Table 12, L2RR consistently achieves the best average performance across different models and source datasets. Compared with the original source-only contenders, both Target-Mix and TargetVal variants generally improve target-domain performance, confirming that held-out target information is useful under source-target mismatch. However, the performance of Target-Mix and TargetVal is largely comparable, and both remain inferior to L2RR. This suggests that the gains of L2RR do not simply come from access to additional target data; rather, explicitly using target-side feedback to learn the source-prompt sampling distribution provides a more effective way to exploit limited target information than directly mixing target examples into training or using them only for model selection.

B.2 Comparison with Direct Target Training

To examine how target information is used, we compare L2RR with TARGET-ONLY, which performs GRPO policy training exclusively on the target validation split, and TARGET-MIX variants in Table 12, which mix this split with source data for training. Both baselines require direct access to target prompts and therefore exceed the target-feedback-only access assumed by L2RR.

Table 5 shows that training on the small target split alone yields substantially lower accuracy, while L2RR also outperforms the best Target-Mix variant in each configuration. These comparisons examine different uses of target information.

Table 5: Diagnostic comparison with direct target training. We report average test accuracy (%) across the six math reasoning benchmarks. Best Target-Mix is the highest average accuracy among the DAPO, GRESO, and AR3PO Target-Mix variants in Table 12. Target-only uses no source data, so its result is repeated across the DM and OR1 rows for each model.

Model	Source	Target-only	Best Target-Mix	L2RR
Qwen2.5-Math-1.5B	DM	39.3	50.4	**50.9**
	OR1	39.3	47.7	**48.7**
DeepSeek-R1-Distill-Qwen-1.5B	DM	50.1	61.6	**62.4**
	OR1	50.1	58.1	**59.1**
Qwen2.5-Math-7B	DM	47.4	58.7	**59.3**
	OR1	47.4	56.0	**56.8**

B.3 Comparison with Simple Target-Guided Baselines

We compare L2RR with two simple target-guided baselines using Qwen2.5-Math-1.5B trained on DM and OR1. Both baselines sample source prompts uniformly and use the same target-feedback interface and validation-based checkpoint-selection protocol as L2RR, with evaluation on the disjoint test splits.

Gradient-Similarity Thresholding. For each source prompt in the current batch, we compute the alignment score $s_{i,t} = \langle g_{t+1}^{\mathrm{tgt}}, g_{t,i} - \bar{g}_t \rangle$, where $\bar{g}_t = B^{-1} \sum_{j \in \mathcal{B}_t} g_{t,j}$ and the target gradient is evaluated at the same look-ahead checkpoint used by L2RR. Only prompts with positive scores are retained in the policy update, with equal weights.

Target-Loss Reweighting. Following the one-step reweighting strategy of [32], we form a virtual update $\theta_t'(\varepsilon) = \theta_t - \eta_\theta \sum_{j \in \mathcal{B}_t} \varepsilon_j g_{t,j}$ and assign weights proportional to $[-\partial L_{\mathrm{tgt}}(\theta_t'(\varepsilon))/\partial \varepsilon_i|_{\varepsilon=0}]_+$. We normalize the positive weights to sum to one and use them in the policy update, falling back to uniform weights when all weights are zero. Both baselines operate only on the current batch, without learning a persistent sampler or adaptively allocating rollouts.

Rollout budget. We keep the training steps and source batch size fixed and choose a fixed per-prompt group size for each baseline to match L2RR's cumulative source-rollout count. Group sizes are selected from the same feasible range based only on rollout counts. Rollouts discarded by Gradient-Similarity Thresholding are included in the count.

As shown in Table 6, L2RR improves over Gradient-Similarity Thresholding 1.0 percentage point on both sources and over Target-Loss Reweighting by 0.9/0.8 points on DM/OR1, respectively. These results support the combined design of persistent sampler learning, history-dependent rollout-waste control, and adaptive rollout allocation beyond using target feedback for within-batch filtering or reweighting alone.

B.4 Quantifying Source–Target Mismatch

We quantify source–target mismatch for the 12 pairs formed by the two source datasets and six target benchmarks using s-OTDD [28] and $1 - \mathrm{MAUVE}$ [30]. Both are oriented so that larger values indicate greater mismatch. We report $1 - \mathrm{MAUVE}$ because MAUVE itself measures distributional similarity.

For each pair, we sample $\min(512, N)$ prompts from each side, where N is the number of prompts in the target benchmark, and average the distance estimates over five independent source subsamples. Both metrics use the same prompt representations from the frozen sentence encoder

Table 6: Comparison with simple target-guided baselines. We report average test accuracy (%) across the six math reasoning benchmarks and cumulative source-rollout counts for Qwen2.5-Math-1.5B. All methods use the same target-feedback interface.

Source	Method	Avg. (%)	Total rollouts
DM	Gradient-Similarity Thresholding	49.9	3.0M
	Target-Loss Reweighting	50.0	3.0M
	L2RR	**50.9**	3.0M
OR1	Gradient-Similarity Thresholding	47.7	1.4M
	Target-Loss Reweighting	47.9	1.4M
	L2RR	**48.7**	1.4M

`all-mpnet-base-v2`, which is not updated during training. For s-OTDD, we assign difficulty labels using each prompt's empirical success rate over 16 responses sampled from a frozen pretrained model. We compute difficulty tertiles separately within each dataset to obtain easy, medium, and hard groups. The $1 - \mathrm{MAUVE}$ metric uses the same representations without difficulty labels. This analysis is entirely post hoc, and neither the distance estimates nor the difficulty labels are used for policy training, sampler updates, checkpoint selection, or hyperparameter tuning.

For each model and source–target pair, we define the gain Δ as L2RR's mean test accuracy minus the highest mean test accuracy among DAPO, GRESO, and AR3PO, using the five-run means in Table 1. Table 7 reports the distances and gains in percentage points.

Table 7: Source–target mismatch and L2RR gains. Distances are averaged over five source subsamples. Gains are computed from the five-run mean test accuracies in Table 1. Qwen-1.5B, DeepSeek-1.5B, and Qwen-7B denote Qwen2.5-Math-1.5B, DeepSeek-R1-Distill-Qwen-1.5B, and Qwen2.5-Math-7B, respectively.

Source	Target	s-OTDD	$1 - \mathrm{MAUVE}$	Δ Qwen-1.5B	Δ DeepSeek-1.5B	Δ Qwen-7B
DM	Math500	0.235	0.194	$+1.8$	$+1.8$	$+1.2$
	AIME24	0.369	0.303	$+1.5$	$+1.9$	-0.9
	AMC	0.204	0.175	$+0.9$	$+1.3$	$+0.9$
	Gaokao	0.290	0.238	$+1.3$	$+1.4$	$+0.5$
	Minerva	0.302	0.288	-0.2	$+0.9$	$+1.2$
	OlympiadBench	0.422	0.379	$+2.8$	$+2.5$	$+2.1$
OR1	Math500	0.298	0.248	$+1.8$	$+1.1$	$+0.9$
	AIME24	0.358	0.320	$+1.3$	$+1.8$	$+1.5$
	AMC	0.206	0.162	$+1.3$	-0.4	$+1.3$
	Gaokao	0.274	0.240	$+1.1$	$+1.7$	$+1.2$
	Minerva	0.311	0.286	$+0.4$	$+1.4$	$+0.7$
	OlympiadBench	0.422	0.382	$+2.3$	$+3.1$	$+1.4$

The relationship between mismatch and gain is model-dependent. Across the 12 pairs, the Pearson correlations with s-OTDD/$1 - \mathrm{MAUVE}$ are 0.48/0.37, 0.76/0.75, and 0.08/0.17 for Qwen-1.5B, DeepSeek-1.5B, and Qwen-7B, respectively. The association is strongest for DeepSeek-1.5B and weak for Qwen-7B. These descriptive correlations concern a small set of pairs sharing source datasets and target benchmarks, and should not be interpreted as a causal relationship or a guarantee of gains on new datasets.

B.5 Sensitivity to the Rollout-Waste Weight

We extend the rollout-waste weight ablation with a finer sweep over $\lambda \in \{0.10, 0.15, 0.20, 0.25, 0.30, 0.40, 0.50\}$, using Qwen2.5-Math-1.5B trained on DM. Table 8 reports accuracy on four target benchmarks and includes $\lambda = 0.01$ and 1.00 for comparison.

Table 8: Fine-grained sensitivity analysis of the rollout-waste weight λ. We report accuracy (%) for Qwen2.5-Math-1.5B trained on DM. Bold indicates the highest reported accuracy for each benchmark, including ties.

Benchmark / λ	0.01	0.10	0.15	0.20	0.25	0.30	0.40	0.50	1.00
Math500	78.8	79.0	79.1	79.3	**79.9**	79.5	78.7	78.6	78.3
AIME24	16.7	17.6	17.6	**18.5**	**18.5**	**18.5**	17.6	17.6	16.7
AMC	61.1	61.5	61.5	61.8	**62.2**	**62.2**	61.8	61.8	61.5
Gaokao	66.1	66.3	66.3	66.3	**67.3**	66.5	66.1	66.2	65.9

All four benchmarks peak near $\lambda = 0.25$, and moderate values in $[0.10, 0.50]$ stay within 1.3, 0.9, 0.7, and 1.2 percentage points of the peak on Math500, AIME24, AMC, and Gaokao, respectively, while the lowest values occur at $\lambda = 0.01$ or $\lambda = 1.00$. Each λ value is a single run, and all of these gaps are smaller than the five-seed standard deviations of L2RR in this setting (1.7–2.4 points; Table 1), so the sweep supports a broad moderate range around $\lambda = 0.25$ rather than a sharply tuned optimum.

B.6 Sensitivity to Target Validation Bias

We examine how biased target validation data affect L2RR when trained on DM and OR1. We consider difficulty bias on Math500 and Gaokao and subdomain bias on OlympiadBench. In each setting, the target validation set contains 10% of the benchmark examples, and the test examples are disjoint from the validation set.

Difficulty bias. For Math500, we group difficulty levels 1–2 as easy, level 3 as medium, and levels 4–5 as hard. For Gaokao, we use the official per-question score as a prespecified difficulty proxy, with higher-scoring questions treated as harder. Uniform validation samples examples uniformly from the benchmark. Easy-biased and Hard-biased validation use easy/medium/hard proportions of 70%/20%/10% and 10%/20%/70%, respectively. For each condition, we sample the test set from the remaining examples according to the original benchmark's empirical difficulty proportions.

Table 9: Sensitivity to difficulty-biased target validation data. Accuracy is reported in percent. Δ is the change in the average accuracy over Math500 and Gaokao relative to Uniform for the same source dataset, in percentage points.

Source	Validation set	Math500	Gaokao	Δ	Total rollouts
DM	Uniform	79.9	67.3	–	3.0M
	Easy-biased	80.0	67.6	$+0.2$	2.8M
	Hard-biased	79.8	66.7	-0.4	3.3M
OR1	Uniform	78.8	66.7	–	1.4M
	Easy-biased	78.9	65.8	-0.4	1.3M
	Hard-biased	78.1	66.6	-0.4	1.6M

Subdomain bias. We use OlympiadBench's Algebra, Geometry, Number Theory, and Combinatorics labels. Each biased validation set contains 70% examples from the designated subdomain and 30% from the other subdomains combined. For each condition, we sample the test set from the remaining examples according to the original benchmark's empirical subdomain proportions. Thus, these bias experiments preserve the original target group proportions in the test sampling, but do not use an identical fixed test set across conditions.

Table 10: Sensitivity to subdomain-biased target validation data. We report OlympiadBench accuracy (%), its change Δ relative to Uniform for the same source dataset in percentage points, and cumulative rollout counts.

Source	Validation set	OlympiadBench	Δ	Total rollouts
DM	Uniform	41.5	–	3.0M
	Algebra-biased	41.0	-0.5	3.1M
	Geometry-biased	41.8	$+0.3$	2.9M
	Number-Theory-biased	41.2	-0.3	3.1M
	Combinatorics-biased	41.0	-0.5	3.2M
OR1	Uniform	41.7	–	1.4M
	Algebra-biased	42.1	$+0.4$	1.4M
	Geometry-biased	41.3	-0.4	1.5M
	Number-Theory-biased	41.4	-0.3	1.4M
	Combinatorics-biased	41.7	0.0	1.5M

Tables 9 and 10 show relatively small accuracy changes under the tested bias conditions. The reported two-benchmark average changes range from -0.4 to $+0.2$ percentage points under difficulty bias, while OlympiadBench changes range from -0.5 to $+0.4$ points under subdomain bias. Rollout counts also vary with the validation composition. These observations are limited to the tested settings, and the comparisons may additionally reflect test-set sampling differences across conditions.

Table 11: Code generation with full and partial verifiers. We report test scores (%) on HumanEval+ and MBPP+, their average, and cumulative source-rollout counts for Qwen2.5-Coder-1.5B-Instruct trained on CodeContests. The verifier condition applies to training and target feedback, while final evaluation always uses the full test suites.

Verifier	Method	HumanEval+	MBPP+	Avg.	Total rollouts
Full unit tests	AR3PO	63.3	58.5	60.9	2.5M
	L2RR	**64.9**	**59.4**	**62.2**	2.7M
Random 50% of tests	AR3PO	59.8	55.6	57.7	2.5M
	L2RR	**61.9**	**56.9**	**59.4**	2.6M

B.7 Code Generation with Full and Partial Verifiers

To evaluate L2RR beyond mathematical reasoning, we train Qwen2.5-Coder-1.5B-Instruct on the training split of CodeContests [21], whose problems include executable test cases, and use HumanEval+ and MBPP+ [22] as target benchmarks. We reserve 10% of each target benchmark for sampler feedback and checkpoint selection and evaluate on the disjoint remaining 90%. Target validation examples are not included in L2RR's policy training, and AR3PO uses the same validation splits for checkpoint selection.

We compare two verifier conditions: full unit tests and a partial verifier that randomly selects 50% of the available tests at each call. In the partial-verifier condition, both training rewards and target feedback use the sampled test subsets. Final evaluation always uses the full test suites for the held-out test problems, regardless of the verifier used during training.

Table 11 shows that L2RR improves the average score over AR3PO by 1.3 percentage points with the full verifier and by 1.7 points with the partial verifier. L2RR uses 2.7M and 2.6M rollouts in these settings, respectively, compared with 2.5M for AR3PO. These results provide preliminary evidence that the L2RR training mechanism is applicable to code generation under the tested verifier conditions.

C Related Work

Reinforcement learning from verifiable rewards. RLVR has emerged as a powerful approach for improving the reasoning ability of LLMs, which leverages verifiers to provide rule based outcome rewards such as correctness in mathematics. Methods such as *proximal policy optimization* (PPO) [29] and GRPO [33] are representative algorithms. Encouraged by their success, a growing body of work has emerged to further improve them for LLM reasoning [44, 43, 23, 40, 46, 48]. For example, the decoupled clip and dynamic sampling policy optimization algorithm (DAPO) proposed by [43] introduces several techniques to improve GRPO, including dynamic sampling, which filters out zero-variance prompts and refills the training batch with effective training data through resampling. [46] proposed the adaptive rollout and response reuse policy optimization algorithm (AR3PO) that improves RLVR sampling efficiency via adaptive rollouts and response reuse, with corrections to reduce mismatch between the current policy and the behavior policy that generated the reused data. GRPO with efficient selective rollout method (GRESO) [48] reduces rollout cost by predicting and skipping prompts that are likely to stay uninformative, leveraging the empirical temporal consistency of zero-advantage prompts across epochs. Guided adversarial self-play (GASP) improves reasoning robustness to corrupted context through guided adversarial self-play [19]. Consolidating rewarded perturbations (CoRP) explores gradient-free LLM post-training by consolidating rewarded parameter perturbations into a single model [47].

A related line of work studies data selection to improve the efficiency of supervised fine-tuning for LLMs [13, 39, 25], and parallel efforts target data efficiency in RL for LLMs [26, 20, 5]. These approaches often require substantial training on the full dataset to identify useful samples, so they provide limited end-to-end savings in rollout-heavy RL training. More broadly, [3] make an important advance in non-stationary decision-making through the first polynomial-time rehearsal-based method for their linear-Gaussian AUF formulation. Their work shares our emphasis on cost-aware decision-making, while focuses on minimizing action costs to avoid undesired outcomes rather than allocating rollouts for RLVR training.

Bi-level optimization for sample reweighting. A closely related line of research studied bi-level optimization for learning data weights: an inner loop trains model parameters on weighted training samples, while an outer loop updates the weights to minimize loss on a held-out development set. Early hypergradient methods differentiate through the training dynamics to optimize such outer objectives [24, 7]. Building on this paradigm, [32] proposed to learn per-example weights so that one gradient step on the reweighted training loss improves a clean validation objective, and [35] learned an explicit loss-to-weight mapping using a small meta set. Conceptually, these methods use limited target-side feedback to steer which training signals matter most; our setting is analogous in spirit, but differs technically because RLVR with GRPO induces history-dependent degeneracy, so we need to consider rollout efficiency in addition to validation performance.

On the theory side, [27] established the foundational stochastic mirror-descent analyses. [4] provided regret and convergence guarantees for stochastic first-order methods, which motivates analyzing our stochastic sampler update via standard gradient-mapping machinery. Recently, [15] established convergence and complexity guarantees for stochastic bilevel optimization under smoothness and strong-convexity-type conditions, under their stated assumptions; these results do not by themselves prove the validity of the present one-step RLVR surrogate.

D Proof of Theorem 1

We first prove the sampler's online regret bound, then examine the effects of estimation error and rollout allocation. Finally, we give a separate target-loss guarantee under additional assumptions.

D.1 Preliminary

Let $\mathcal{F}_t$ contain the training history before round t, including $\mathbf{q}_t$, θ_t, and the counters $z_{i,t}$. The feedback used in Eq. (7) is

$$\widehat{g}_t = \widehat{g}_t^{\mathrm{tgt}} + \lambda_T \psi_t, \qquad \psi_{t,i} = 1 - p_e^{z_{i,t}} \in [0,1],$$

where ψ_t is $\mathcal{F}_t$-measurable and λ_T is fixed within the run. The feedback includes batch sampling, rollout generation, and any target-side estimation randomness. Repeated prompt occurrences, if allowed, contribute to the same coordinate of $\widehat{g}_t^{\mathrm{tgt}}$. Starting from $q_{1,i} = 1/N_S$, exponentiated gradient gives

$$q_{t+1,i} = \frac{q_{t,i}\exp(-\eta_q \widehat{g}_{t,i})}{\sum_j q_{t,j}\exp(-\eta_q \widehat{g}_{t,j})}. \tag{12}$$

For finite feedback and $\eta_q > 0$, all coordinates remain strictly positive.

After the feedback is realized, define the linear losses $\widehat{\ell}_t(u) = \langle \widehat{g}_t, u\rangle$ and $\widehat{\ell}_t^{\mathrm{tgt}}(u) = \langle \widehat{g}_t^{\mathrm{tgt}}, u\rangle$ on Δ^{N_S}. Both average regrets, defined as in Eq. (9), compare against the best fixed distribution on the same realized sequence. This is external regret for feedback losses, not the difference between training trajectories induced by different samplers.

Define the conditional mean and centered error by

$$m_t = \mathbb{E}[\widehat{g}_t \mid \mathcal{F}_t], \qquad \xi_t = \widehat{g}_t - m_t,$$

and set

$$\sigma_{q,t}^2 = \mathbb{E}\|\xi_t\|_2^2, \qquad \bar{\sigma}_q^2 = \frac{1}{T}\sum_{t=1}^{T}\sigma_{q,t}^2. \tag{13}$$

Assumption 1 (Feedback second moment)**.** The feedback is square-integrable and satisfies

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\widehat{g}_t\|_\infty^2 \le H^2,$$

where H is independent of the horizon. This is a condition on the feedback, not a consequence of normalized sampling weights: their denominators can approach zero without an enforced sampler floor.

D.2 Main Proof

Lemma 1 (One-step entropy inequality)**.** *For any $u \in \Delta^{N_S}$, finite feedback, and $\eta_q > 0$,*

$$\langle \widehat{g}_t, \mathbf{q}_t - u \rangle \le \frac{\mathrm{KL}(u\|\mathbf{q}_t) - \mathrm{KL}(u\|\mathbf{q}_{t+1})}{\eta_q} + \frac{\eta_q}{2}\|\widehat{g}_t\|_\infty^2.$$

Proof. Let $h(q) = \sum_i q_i \log q_i$ and $p = \mathbf{q}_{t+1}$. The update is the minimizer of $\langle \widehat{g}_t, p\rangle + \eta_q^{-1}\mathrm{KL}(p\|\mathbf{q}_t)$. Its optimality condition and the Bregman three-point identity give

$$\eta_q\langle \widehat{g}_t, p - u\rangle \le \mathrm{KL}(u\|\mathbf{q}_t) - \mathrm{KL}(u\|p) - \mathrm{KL}(p\|\mathbf{q}_t).$$

Add $\eta_q\langle \widehat{g}_t, \mathbf{q}_t - p\rangle$ and use Pinsker's inequality and Hölder's inequality. Writing $d = \|\mathbf{q}_t - p\|_1$, the remaining terms satisfy

$$\eta_q\langle \widehat{g}_t, \mathbf{q}_t - p\rangle - \mathrm{KL}(p\|\mathbf{q}_t) \le \eta_q\|\widehat{g}_t\|_\infty d - \tfrac{1}{2}d^2 \le \tfrac{1}{2}\eta_q^2\|\widehat{g}_t\|_\infty^2.$$

Division by η_q proves the claim. Boundary comparators are allowed with the convention $0\log 0 = 0$. □

Theorem 2 (Formal statement of Theorem 1)**.** *For any realized finite feedback sequence, any constant $\eta_q > 0$, and uniform $\mathbf{q}_1$,*

$$\widehat{R}_T \le \frac{\log N_S}{\eta_q T} + \frac{\eta_q}{2T}\sum_{t=1}^{T}\|\widehat{g}_t\|_\infty^2, \qquad \widehat{R}_T^{\mathrm{tgt}} \le \widehat{R}_T + \lambda_T. \tag{14}$$

Under Assumption 1, setting $\eta_q = a/\sqrt{T}$ gives Eqs. (10)–(11)*. If $\lambda_T \le c_\lambda/\sqrt{T}$, then*

$$\mathbb{E}\widehat{R}_T^{\mathrm{tgt}} \le \frac{\log N_S/a + aH^2/2 + c_\lambda}{\sqrt{T}}.$$

Proof. Sum Lemma 1 and telescope the KL terms. For every $u \in \Delta^{N_S}$, uniform initialization gives $\mathrm{KL}(u\|\mathbf{q}_1) = \log N_S + \sum_i u_i \log u_i \le \log N_S$, and the final KL divergence is nonnegative. The bound holds simultaneously for all u, so it also holds for the best comparator chosen after the full feedback sequence is observed. Dividing by T proves the first inequality in Eq. (14).

For each comparator u,

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T}\langle \widehat{g}_t^{\mathrm{tgt}}, \mathbf{q}_t - u\rangle &= \frac{1}{T}\sum_{t=1}^{T}\langle \widehat{g}_t, \mathbf{q}_t - u\rangle + \frac{\lambda_T}{T}\sum_{t=1}^{T}\langle \psi_t, u - \mathbf{q}_t\rangle \\ &\le \widehat{R}_T + \lambda_T,\end{aligned}$$

since both $\langle \psi_t, u\rangle$ and $\langle \psi_t, \mathbf{q}_t\rangle$ lie in $[0,1]$. Maximizing over u proves the second inequality. Taking expectations, applying the second-moment assumption, and substituting the step size proves the rates. The argument is pathwise before expectations and allows feedback that depends on the sampler and training history; unbiasedness and PL conditions are not needed. □

How noise enters the regret bound. The martingale-difference property yields the exact decomposition

$$\mathbb{E}\|\widehat{g}_t\|_2^2 = \mathbb{E}\|m_t\|_2^2 + \sigma_{q,t}^2. \tag{15}$$

Since $\|\widehat{g}_t\|_\infty \le \|\widehat{g}_t\|_2$, one may replace H^2 in the expected regret bound by $T^{-1}\sum_t \mathbb{E}\|m_t\|_2^2 + \bar{\sigma}_q^2$. Reducing the variance tightens this upper bound when the other terms are held fixed. The regret is for the realized feedback losses; it does not require a vanishing bias or policy-tracking error.

D.3 Assumptions

For the separate target-loss analysis, we define the outer objective for a fixed sampler $\mathbf{q} \in \Delta^{N_S}$:

$$\Phi(\mathbf{q}) = \mathbb{E}[L_{\mathrm{tgt}}\big(\theta_{T+1}(\mathbf{q})\big)] + \lambda\, L_{\mathrm{ws}}(\mathbf{q}), \tag{16}$$

where $\theta_{T+1}(\mathbf{q})$ is a random parameter after T GRPO updates when sampling data from $\mathbf{q}$, and $L_{\mathrm{ws}}(\mathbf{q})$ is the expected history-based time-averaged rollout-waste functional. This defines a deterministic idealized fixed-sampler objective. Relating the coupled algorithm to it requires the assumptions below; the relation is not implied by the one-step surrogate.

Define the KL mirror step

$$\mathcal{M}_{\eta_q}(\mathbf{q}; v) = \arg\min_{u \in \Delta^{N_S}} \left\{ \langle v, u\rangle + \frac{1}{\eta_q}\mathrm{KL}(u\|\mathbf{q}) \right\},$$

and the corresponding KL-gradient mapping of Φ:

$$G_{\eta_q}(\mathbf{q}) = \frac{1}{\eta_q}\Big(\mathbf{q} - \mathcal{M}_{\eta_q}(\mathbf{q}; \nabla\Phi(\mathbf{q}))\Big). \tag{17}$$

Assumption 2. Assume Φ is continuously differentiable on a neighborhood of Δ^{N_S} and its gradient is L_q-Lipschitz on the simplex:

$$\|\nabla\Phi(\mathbf{q}) - \nabla\Phi(\mathbf{q}')\|_2 \le L_q\|\mathbf{q} - \mathbf{q}'\|_2, \qquad \forall \mathbf{q}, \mathbf{q}' \in \Delta^{N_S}.$$

Here $L_q > 0$ is an assumed smoothness constant, not a consequence of the normalized one-step weights. No fixed positive sampler floor is enforced by Eq. (12). Let $\Phi^{\inf} = \inf_{q \in \Delta^{N_S}} \Phi(q) > -\infty$. Define $M_\Phi = \sup_{\mathbf{q}\in\Delta^{N_S}} \|\nabla\Phi(\mathbf{q})\|_2$ and $K_\Phi = L_q + 3M_\Phi$. Compactness and the global Lipschitz-gradient condition imply $M_\Phi < \infty$.

Assumption 3. Let $\widehat{g}_t = (\widehat{g}_{t,i})_{i=1}^{N_S}$ be the stochastic gradient used in (12). In addition to the preceding online setup, assume

$$\mathbb{E}[\widehat{g}_t \mid \mathcal{F}_t] = \nabla\Phi(\mathbf{q}_t).$$

This identifies m_t with $\nabla\Phi(\mathbf{q}_t)$ for the target-loss analysis only; it is not required for Theorem 2.

Assumption 4. Define the meta objective

$$F_\lambda(q) = L_{\mathrm{tgt}}(\theta^\star(q)) + \lambda L_{\mathrm{ws}}(q),$$

where $\theta^\star(q)$ is a fixed, deterministic choice of idealized inner solution for sampler q. Write $f(q) = L_{\mathrm{tgt}}(\theta^\star(q))$ and $R(q) = L_{\mathrm{ws}}(q) \in [0,1]$, using the expected waste functional under constant sampling. This $R(q)$ is distinct from the instantaneous history-dependent linear penalty $\langle \psi_t, q\rangle$. Assume F_λ is differentiable and the mismatch between the two reference-objective gradients is bounded in second moment:

$$\mathbb{E}\left[\|\nabla\Phi(q_t) - \nabla F_\lambda(q_t)\|_2^2\right] \le \varepsilon^2, \qquad \forall t \in [T].$$

Assumption 5. Assume along the coupled trajectory $(\theta_t, \mathbf{q}_t)$,

$$\mathbb{E}\big[\|\theta_t - \theta^\star(\mathbf{q}_t)\|_2^2\big] \le \delta_t,$$

and L_{tgt} is Lipschitz in θ on the trajectory: for some $L_\theta > 0$,

$$|L_{\mathrm{tgt}}(\theta) - L_{\mathrm{tgt}}(\theta^\star(\mathbf{q}))| \le L_\theta\|\theta - \theta^\star(\mathbf{q})\|_2.$$

Assumption 6. Assume the minimum of F_λ on Δ^{N_S} is attained, and let $\mathbf{q}_\lambda^\star \in \arg\min_{\mathbf{q}\in\Delta^{N_S}} F_\lambda(\mathbf{q})$. Assume that on the region visited by $\{\mathbf{q}_t\}$, F_λ satisfies a PL inequality with respect to its KL-gradient mapping $G_{\eta_q}^{F_\lambda}$: there exists $\mu > 0$ such that

$$F_\lambda(\mathbf{q}) - F_\lambda(\mathbf{q}_\lambda^\star) \le \frac{1}{2\mu}\left\|G_{\eta_q}^{F_\lambda}(\mathbf{q})\right\|_2^2.$$

The mapping $G_{\eta_q}^{F_\lambda}$ is defined by replacing Φ with F_λ in Eq. (17).

D.4 Technical Lemmas

To distinguish feedback regret from an idealized comparison, let v_t be an $\mathcal{F}_t$-measurable reference target-gradient vector and write

$$\mathbb{E}[\widehat{g}_t^{\mathrm{tgt}} \mid \mathcal{F}_t] = v_t + b_t, \qquad \mathbb{E}\|b_t\|_\infty \le \varepsilon_t.$$

Define the cumulative reference regret $R_T^{\mathrm{ref}} = \max_{u\in\Delta^{N_S}} \sum_t \langle v_t, \mathbf{q}_t - u\rangle$. The reference vectors are evaluated along the actual history; choosing $v_t = \nabla f(\mathbf{q}_t)$ does not make this regret in the values of a nonconvex f.

Proposition 1 (Reference-gradient regret)**.** *Under Assumption 1 and the preceding bias bound,*

$$\frac{\mathbb{E}R_T^{\mathrm{ref}}}{T} \le \frac{\log N_S/a + aH^2/2 + \sqrt{\bar\sigma_q^2}}{\sqrt{T}} + \lambda_T + \frac{2}{T}\sum_{t=1}^{T}\varepsilon_t. \tag{18}$$

Proof. Because the waste gradient is $\mathcal{F}_t$-measurable, $\widehat{g}_t = v_t + b_t + \lambda_T\psi_t + \xi_t$. For every u, decompose the reference regret into these four terms. Use $\|\mathbf{q}_t - u\|_1 \le 2$ for the bias and $\langle \psi_t, u - \mathbf{q}_t\rangle \le 1$ for the penalty. Taking the supremum gives

$$R_T^{\mathrm{ref}} \le T\widehat{R}_T + \lambda_T T + 2\sum_t \|b_t\|_\infty - \sum_t \langle \xi_t, \mathbf{q}_t\rangle + \left\|\sum_t \xi_t\right\|_\infty.$$

Adaptedness implies $\mathbb{E}\langle \xi_t, \mathbf{q}_t\rangle = 0$. Orthogonality of square-integrable martingale differences and Jensen's inequality give

$$\mathbb{E}\left\|\sum_t \xi_t\right\|_\infty \le \sqrt{\mathbb{E}\left\|\sum_t \xi_t\right\|_2^2} = \sqrt{\sum_t \sigma_{q,t}^2}.$$

Divide by T and apply the average-regret bound in Theorem 2. The last term accounts for the comparator being selected in hindsight, so no independence of that comparator is assumed. □

With uniformly bounded $\bar\sigma_q^2$, $\lambda_T \le c_\lambda/\sqrt{T}$, and $\sum_t \varepsilon_t = \mathcal{O}(\sqrt{T})$, reference-gradient regret is also $\mathcal{O}(\sqrt{T})$. A constant bias bound $\varepsilon_t \le \varepsilon$ instead contributes $2\varepsilon T$ to cumulative regret, or 2ε to average regret. Thus a constant residual in average regret is not a constant additive term in cumulative regret. For a fixed penalty weight, removing the penalty similarly costs at most $\lambda_T T$; no such removal is needed for the regularized feedback guarantee.

Lemma 2 (Noise-sensitive EG descent)**.** *Under Assumptions 2 and 3, with finite $\sigma_{q,t}^2$ as in Eq.* (13)*, the EG iterates satisfy, for $0 < \eta_q \le 1/L_q$,*

$$\min_{t\in[T]} \mathbb{E}\|G_{\eta_q}(\mathbf{q}_t)\|_2^2 \le \frac{2\Delta_\Phi}{\eta_q T} + K_\Phi \eta_q \bar\sigma_q^2, \qquad \Delta_\Phi = \Phi(\mathbf{q}_1) - \Phi^{\inf}. \tag{19}$$

Proof. We exploit the softmax form of EG before bounding the noise. This preserves cancellation of the first-order, conditionally mean-zero error. Nonconvex stochastic mirror-descent rates accommodating entropy geometry are also studied in [6]; the following proof is self-contained for this EG update.

Step 1: a smooth function in logit coordinates. Let $S(z) = \mathrm{softmax}(z)$, $p = S(z)$, and $J(z) = \mathrm{diag}(p) - pp^\top$. Since $0 \preceq J(z) \preceq I$, S is 1-Lipschitz in ℓ_2. For $p' = S(z')$,

$$\|J(z) - J(z')\|_{\mathrm{op}} \le \|p - p'\|_\infty + (\|p\|_2 + \|p'\|_2)\|p - p'\|_2 \le 3\|z - z'\|_2.$$

Consequently $\Psi(z) = \Phi(S(z))$ has a K_Φ-Lipschitz gradient: using $\nabla\Psi(z) = J(z)\nabla\Phi(S(z))$,

$$\|\nabla\Psi(z) - \nabla\Psi(z')\|_2 \le (L_q + 3M_\Phi)\|z - z'\|_2.$$

This bound is uniform over finite logits and does not require a fixed $q_{\min} > 0$.

Step 2: use conditional unbiasedness before Young-type bounds. Set $a_t = \nabla\Phi(\mathbf{q}_t)$, $e_t = \widehat{g}_t - a_t$, $z_t = \log \mathbf{q}_t - \eta_q a_t$, and $p_t = S(z_t) = \mathcal{M}_{\eta_q}(\mathbf{q}_t; a_t)$. The actual iterate is $\mathbf{q}_{t+1} = S(z_t - \eta_q e_t)$. Conditional on $\mathcal{F}_t$, z_t is fixed and $\mathbb{E}[e_t \mid \mathcal{F}_t] = 0$. Smoothness of Ψ therefore gives

$$\begin{aligned}\mathbb{E}[\Phi(\mathbf{q}_{t+1}) \mid \mathcal{F}_t] &\le \Phi(p_t) - \eta_q\langle\nabla\Psi(z_t), \mathbb{E}[e_t \mid \mathcal{F}_t]\rangle + \frac{K_\Phi\eta_q^2}{2}\mathbb{E}[\|e_t\|_2^2 \mid \mathcal{F}_t] \\ &= \Phi(p_t) + \frac{K_\Phi\eta_q^2}{2}\mathbb{E}[\|e_t\|_2^2 \mid \mathcal{F}_t].\end{aligned}$$

Step 3: exact-step descent and telescoping. Negative entropy $h(p) = \sum_i p_i \log p_i$ is 1-strongly convex in ℓ_2. The exact mirror-step optimality condition gives

$$\langle a_t, p_t - \mathbf{q}_t\rangle \le -\eta_q^{-1}\langle\nabla h(p_t) - \nabla h(\mathbf{q}_t), p_t - \mathbf{q}_t\rangle \le -\eta_q^{-1}\|p_t - \mathbf{q}_t\|_2^2.$$

By L_q-smoothness and $\eta_q \le 1/L_q$,

$$\Phi(p_t) \le \Phi(\mathbf{q}_t) - \frac{1}{2\eta_q}\|p_t - \mathbf{q}_t\|_2^2 = \Phi(\mathbf{q}_t) - \frac{\eta_q}{2}\|G_{\eta_q}(\mathbf{q}_t)\|_2^2.$$

Combining the last two bounds, taking expectations, and summing yields

$$\frac{\eta_q}{2}\sum_{t=1}^{T}\mathbb{E}\|G_{\eta_q}(\mathbf{q}_t)\|_2^2 \le \Delta_\Phi + \frac{K_\Phi\eta_q^2}{2}\sum_{t=1}^{T}\sigma_{q,t}^2.$$

Divide by $\eta_q T/2$ and take the minimum. The first-order noise cancellation uses conditional unbiasedness; the bound does not apply unchanged to an arbitrary biased feedback vector. □

Corollary 1 (Target-loss bound)**.** Under Assumptions 2, 3, 4, 5, and 6, with finite variances in Eq. (13), fix a horizon $T \ge 1$, choose $\eta_q = 1/(L_q\sqrt{T})$ and $0 \le \lambda = \lambda_T \le c_\lambda/\sqrt{T}$, and hold both values fixed throughout the run. Here $c_\lambda \ge 0$ is independent of T. Let $\bar{t} \in \arg\min_{t\in[T]}\mathbb{E}\|G_{\eta_q}(\mathbf{q}_t)\|_2^2$, $R = L_{\mathrm{ws}} \in [0,1]$, and $\mathbf{q}_{\lambda_T}^\star \in \arg\min_{\mathbf{q}} F_{\lambda_T}(\mathbf{q})$. Then

$$\begin{aligned}\mathbb{E}\big[L_{\mathrm{tgt}}(\theta_{\bar{t}}) - L_{\mathrm{tgt}}(\theta^\star(\mathbf{q}_{\lambda_T}^\star))\big] \le{}& \frac{2L_q\Delta_\Phi/\mu + K_\Phi\bar{\sigma}_q^2/(\mu L_q) + c_\lambda}{\sqrt{T}} \\ &+ \frac{\varepsilon^2}{\mu} + L_\theta\sqrt{\delta_{\bar{t}}}.\end{aligned} \tag{20}$$

Proof. For mirror steps from the same $\mathbf{q}$, strong convexity and the two optimality conditions imply

$$\|\mathcal{M}_{\eta_q}(\mathbf{q}; v) - \mathcal{M}_{\eta_q}(\mathbf{q}; v')\|_2 \le \eta_q\|v - v'\|_2.$$

Indeed, writing the two steps as p, p', addition of their variational inequalities yields $\|p - p'\|_2^2 \le \eta_q\langle v' - v, p - p'\rangle$. Assumption 4 therefore gives

$$\mathbb{E}\|G_{\eta_q}^{F_\lambda}(\mathbf{q}_{\bar{t}})\|_2^2 \le 2\mathbb{E}\|G_{\eta_q}(\mathbf{q}_{\bar{t}})\|_2^2 + 2\varepsilon^2.$$

Combine Lemma 2 with Assumption 6 to obtain

$$\mathbb{E}[F_\lambda(\mathbf{q}_{\bar{t}}) - F_\lambda(\mathbf{q}_\lambda^\star)] \le \frac{2\Delta_\Phi}{\mu\eta_q T} + \frac{K_\Phi\eta_q\bar{\sigma}_q^2 + \varepsilon^2}{\mu}.$$

For $f(\mathbf{q}) = L_{\mathrm{tgt}}(\theta^\star(\mathbf{q}))$, use the exact identity

$$f(\mathbf{q}) - f(\mathbf{q}_\lambda^\star) = F_\lambda(\mathbf{q}) - F_\lambda(\mathbf{q}_\lambda^\star) + \lambda[R(\mathbf{q}_\lambda^\star) - R(\mathbf{q})].$$

Taking expectations first gives, for any $0 < \eta_q \le 1/L_q$,

$$\begin{aligned}\mathbb{E}[f(\mathbf{q}_{\bar{t}}) - f(\mathbf{q}_\lambda^\star)] \le{}& \frac{2\Delta_\Phi}{\mu\eta_q T} + \frac{K_\Phi\eta_q\bar{\sigma}_q^2 + \varepsilon^2}{\mu} \\ &+ \lambda\big(R(\mathbf{q}_\lambda^\star) - \mathbb{E}R(\mathbf{q}_{\bar{t}})\big).\end{aligned} \tag{21}$$

Because $0 \le R \le 1$ and $0 \le \lambda = \lambda_T \le c_\lambda/\sqrt{T}$, $\lambda_T(R(\mathbf{q}_{\lambda_T}^\star) - \mathbb{E}R(\mathbf{q}_{\bar{t}})) \le c_\lambda/\sqrt{T}$. Finally, Assumption 5 and Cauchy–Schwarz give

$$\mathbb{E}[L_{\mathrm{tgt}}(\theta_{\bar{t}}) - f(\mathbf{q}_{\bar{t}})] \le L_\theta\mathbb{E}\|\theta_{\bar{t}} - \theta^\star(\mathbf{q}_{\bar{t}})\|_2 \le L_\theta\sqrt{\delta_{\bar{t}}}.$$

Substitute $\eta_q = 1/(L_q\sqrt{T})$ and add this bound to obtain Eq. (20). □

Rate across training horizons. For an asymptotic family of runs, assume the smoothness, gradient, PL, and initial-gap constants are controlled uniformly: for example $0 < L_- \le L_q \le L_+$, $\mu \ge \mu_0 > 0$, $M_\Phi \le M_0$, $\Delta_\Phi \le \Delta_0$, and $\bar{\sigma}_q^2 \le \sigma_0^2$, with L_θ uniformly bounded. The PL inequality must hold for the step sizes used in this family. Equation (20) then gives $\mathcal{O}(T^{-1/2}) + \mathcal{O}(\bar{\sigma}_q^2/\sqrt{T}) + \mathcal{O}(\varepsilon^2) + L_\theta\sqrt{\delta_{\bar{t}}}$. Thus the regularization transfer is absorbed into the convergence term, with no new constant noise or regularization floor. The comparator is the minimizer of F_{λ_T} for the same horizon.

Remark 1. Corollary 1 is a conditional optimization statement about an analytically selected iterate, separate from the online regret in Theorem 2. It does not identify the final or validation-selected checkpoint with that iterate. The fixed-sampler reference Φ, conditional unbiasedness, gradient mismatch, tracking, and the mapping-based PL condition are additional assumptions, not consequences of the one-step implementation or of the feedback regret theorem. The weight λ_T depends on the planned training horizon but is constant within each run. This theoretical parameter rule is not asserted for the reported experiments.

D.5 Rollout Allocation Proposition

The following statements connect the rollout model to the variance in Eqs. (15) and (20). They are not needed for the feedback regret theorem.

Assumption 7. At iteration t, the rollouts per prompt are $\{G_{i,t}\}_{i\in\mathcal{B}_t}$ and the batch-local budget is RO_t, with $\sum_{i\in\mathcal{B}_t} G_{i,t} \le \mathrm{RO}_t$ and $0 < G_{\min} \le G_{i,t} \le G$. Condition on $\mathcal{F}_t$ and the sampled batch $\mathcal{B}_t$. For the coordinate-variance and allocation statements below, assume distinct sampled indices. For repeated indices, the same argument requires bounds for the aggregated coordinate contributions. The allocation and variance proxies are measurable given this history and batch, before the evaluated rollouts are generated. Assume the target-driven coordinate estimator satisfies

$$\mathrm{Var}\big(\widehat{g}_{t,i}^{\mathrm{tgt}} \mid \mathcal{F}_t, \mathcal{B}_t\big) \le \frac{c_{i,t}}{G_{i,t}}, \qquad c_{i,t} \ge 0, \tag{22}$$

and assume $\lambda_T\psi_{t,i}$ is $\mathcal{F}_t$-measurable, thus it contributes no conditional variance given this history. The $c_{i,t}$ must bound the aggregated coordinate errors; setting $c = w^2$ heuristically does not establish this assumption.

Lemma 3. *Under Assumption 7, define* $m_{t,\mathcal{B}} = \mathbb{E}[\widehat{g}_t \mid \mathcal{F}_t, \mathcal{B}_t]$ *and*

$$V_t = \mathbb{E}\big[\|m_{t,\mathcal{B}} - m_t\|_2^2\big] .$$

Then the total noise satisfies

$$\sigma_{q,t}^2 \le \mathbb{E}\left[\sum_{i\in\mathrm{supp}(\mathcal{B}_t)} \frac{c_{i,t}}{G_{i,t}}\right] + V_t.$$

Proof. The law of total conditional variance gives

$$\sigma_{q,t}^2 = \mathbb{E}\|\widehat{g}_t - m_{t,\mathcal{B}}\|_2^2 + \mathbb{E}\|m_{t,\mathcal{B}} - m_t\|_2^2.$$

Conditional on the batch and history, unsampled target-feedback coordinates are identically zero and the waste term is deterministic. The first term is therefore the sum of the conditional variances of the sampled coordinates, bounded by Assumption 7. The second term is V_t and includes batch-selection randomness over all coordinates. It cannot generally be omitted. □

The next lemma gives the optimum of the strictly positive, unconstrained continuous weighted proxy. Clipping or discretization is not covered by this lemma.

Lemma 4. *Fix iteration t and a realized batch $\mathcal{B}_t$. Let weights $w_{i,t} > 0$ and $c_{i,t} > 0$ be given for $i \in \mathcal{B}_t$, with $\mathrm{RO}_t > 0$. Consider the continuous relaxation (ignoring clipping and integer constraints)*

$$\min_{\{G_{i,t}>0\}} \sum_{i\in\mathcal{B}_t} \frac{w_{i,t}c_{i,t}}{G_{i,t}} \quad \textit{s.t.} \quad \sum_{i\in\mathcal{B}_t} G_{i,t} = \mathrm{RO}_t.$$

Then the unique optimizer is

$$G_{i,t}^\star = \mathrm{RO}_t \cdot \frac{\sqrt{w_{i,t}c_{i,t}}}{\sum_{j\in\mathcal{B}_t}\sqrt{w_{j,t}c_{j,t}}}, \qquad i \in \mathcal{B}_t,$$

and the optimal value equals

$$\sum_{i\in\mathcal{B}_t}\frac{w_{i,t}c_{i,t}}{G^\star_{i,t}}=\frac{\left(\sum_{i\in\mathcal{B}_t}\sqrt{w_{i,t}c_{i,t}}\right)^2}{\mathrm{RO}_t}.$$

Proof. Form the Lagrangian

$$\mathcal{L}(\{G_{i,t}\},\nu)=\sum_{i\in\mathcal{B}_t}\frac{w_{i,t}c_{i,t}}{G_{i,t}}+\nu\left(\sum_{i\in\mathcal{B}_t}G_{i,t}-\mathrm{RO}_t\right).$$

Stationarity for each $i\in\mathcal{B}_t$ gives

$$\frac{\partial\mathcal{L}}{\partial G_{i,t}}=-\frac{w_{i,t}c_{i,t}}{G_{i,t}^2}+\nu=0\quad\Longrightarrow\quad G_{i,t}=\sqrt{\frac{w_{i,t}c_{i,t}}{\nu}}.$$

Impose the budget constraint:

$$\sum_{i\in\mathcal{B}_t}\sqrt{\frac{w_{i,t}c_{i,t}}{\nu}}=\mathrm{RO}_t\quad\Longrightarrow\quad\sqrt{\nu}=\frac{\sum_{i\in\mathcal{B}_t}\sqrt{w_{i,t}c_{i,t}}}{\mathrm{RO}_t}.$$

Substituting back yields the stated optimizer. Strict convexity of $\sum_i w_{i,t}c_{i,t}/G_{i,t}$ for positive products establishes uniqueness. Finally,

$$\sum_{i\in\mathcal{B}_t}\frac{w_{i,t}c_{i,t}}{G^\star_{i,t}}=\sum_{i\in\mathcal{B}_t}\frac{w_{i,t}c_{i,t}}{\mathrm{RO}_t}\cdot\frac{\sum_{j\in\mathcal{B}_t}\sqrt{w_{j,t}c_{j,t}}}{\sqrt{w_{i,t}c_{i,t}}}=\frac{\left(\sum_{i\in\mathcal{B}_t}\sqrt{w_{i,t}c_{i,t}}\right)^2}{\mathrm{RO}_t}.$$

□

Proposition 2 (Continuous allocation and the unweighted variance term)**.** *Under Lemmas 3 and 4, if the continuous allocation $G^\star_{i,t}$ is used, with strictly positive weights and proxies and without clipping, then*

$$\sigma^2_{q,t}\le V_t+\mathbb{E}\left[\frac{\sum_j\sqrt{w_{j,t}c_{j,t}}}{\mathrm{RO}_t}\sum_i\sqrt{\frac{c_{i,t}}{w_{i,t}}}\right].$$

This is an identity-based substitution for the unweighted variance term, not the smaller minimum of the weighted proxy. It does not prove the implemented clipped allocation minimizes total noise.

Proof. Substitute $G^\star_{i,t}=\mathrm{RO}_t\sqrt{w_{i,t}c_{i,t}}/\sum_j\sqrt{w_{j,t}c_{j,t}}$ into $\sum_i c_{i,t}/G^\star_{i,t}$ and apply Lemma 3. □

Table 12: Performance (%) comparison across six math reasoning benchmarks. Three models are trained on DM or OR1. Base rows use target-validation checkpoint selection; +Target-Mix also trains on target validation examples; -TargetVal omits target-validation selection. Total RO denotes cumulative source responses in millions; RO ratio compares counts with DAPO, not wall-clock time. All entries are single runs; five-seed means of the base methods and L2RR are in Table 1.

Dataset	Method	Math500	AIME24	AMC	Gaokao	Miner.	Olymp.	Avg.	# Rollout	RO ratio
Qwen2.5-Math-1.5B										
DM	DAPO	77.3	16.7	61.8	64.2	31.8	38.7	48.4	7.6M	1.0×
	DAPO + Target-Mix	77.9	17.6	**62.2**	65.1	32.6	39.4	49.1	8.2M	0.9×
	DAPO - TargetVal	77.1	15.7	61.5	63.8	31.2	38.1	47.9	7.6M	1.0×
	GRESO	76.6	14.8	61.8	66.2	33.3	38.5	48.5	3.3M	2.3×
	GRESO + Target-Mix	77.4	15.7	**62.2**	**67.3**	34.5	39.4	49.4	3.8M	2.0×
	GRESO - TargetVal	76.1	13.9	61.1	65.6	32.4	37.8	47.8	3.3M	2.3×
	AR3PO	77.8	17.6	61.1	65.4	35.8	39.0	49.5	2.5M	3.0×
	AR3PO + Target-Mix	78.9	**18.5**	**62.2**	66.5	**36.2**	39.9	50.4	2.8M	2.7×
	AR3PO - TargetVal	76.9	16.7	60.8	64.8	35.1	38.2	48.8	2.5M	3.0×
	L2RR	**79.9**	**18.5**	**62.2**	**67.3**	35.7	**41.5**	**50.9**	3.0M	2.5×
OR1	DAPO	77.1	16.7	50.3	65.5	30.9	39.7	46.7	3.8M	1.0×
	DAPO + Target-Mix	77.8	17.6	51.4	66.4	**31.6**	40.6	47.6	4.0M	1.0×
	DAPO - TargetVal	76.4	15.7	48.6	60.0	30.2	39.3	45.0	3.8M	1.0×
	GRESO	76.1	20.4	50.7	65.1	30.0	39.2	46.9	1.6M	2.4×
	GRESO + Target-Mix	76.8	20.4	51.4	65.9	30.8	39.8	47.5	1.8M	2.1×
	GRESO - TargetVal	75.4	19.4	49.0	64.4	29.2	38.7	46.0	1.6M	2.4×
	AR3PO	76.5	20.4	51.0	65.0	30.2	39.6	47.1	1.4M	2.7×
	AR3PO + Target-Mix	77.1	20.4	51.4	65.9	30.9	40.3	47.7	1.7M	2.2×
	AR3PO - TargetVal	76.1	19.4	50.3	64.3	29.7	38.9	46.5	1.4M	2.7×
	L2RR	**78.8**	**21.3**	**52.4**	**66.7**	31.5	**41.7**	**48.7**	1.4M	2.7×
DeepSeek-R1-Distill-Qwen-1.5B										
DM	DAPO	87.9	37.0	71.6	78.7	35.3	55.9	61.1	2.4M	1.0×
	DAPO + Target-Mix	88.5	37.0	72.3	79.1	35.6	56.4	61.5	2.8M	0.9×
	DAPO - TargetVal	87.3	36.1	71.3	78.1	34.8	55.3	60.5	2.4M	1.0×
	GRESO	87.7	37.0	71.3	78.4	33.9	55.1	60.6	1.6M	1.5×
	GRESO + Target-Mix	88.6	**38.0**	72.0	78.9	34.5	56.0	61.3	1.8M	1.3×
	GRESO - TargetVal	87.2	37.0	70.9	77.9	33.6	54.6	60.2	1.6M	1.5×
	AR3PO	88.0	37.0	71.3	78.6	34.8	55.6	60.9	1.7M	1.4×
	AR3PO + Target-Mix	88.9	37.0	72.3	79.4	35.5	56.4	61.6	1.9M	1.3×
	AR3PO - TargetVal	87.4	36.1	71.3	78.1	34.3	55.2	60.4	1.7M	1.4×
	L2RR	**89.3**	**38.0**	**73.0**	**80.1**	**35.8**	**58.2**	**62.4**	1.6M	1.5×
OR1	DAPO	84.8	25.0	68.2	74.0	34.1	54.2	56.7	2.4M	1.0×
	DAPO + Target-Mix	85.3	25.0	**68.6**	74.9	34.8	54.8	57.2	2.8M	0.9×
	DAPO - TargetVal	84.2	24.1	67.9	73.4	33.5	53.6	56.1	2.4M	1.0×
	GRESO	85.9	26.9	66.9	75.2	33.6	55.5	57.3	1.5M	1.6×
	GRESO + Target-Mix	86.5	27.8	67.9	75.9	34.2	56.1	58.1	1.8M	1.3×
	GRESO - TargetVal	85.4	25.9	66.2	74.8	33.0	55.1	56.7	1.5M	1.6×
	AR3PO	85.4	26.9	67.2	75.0	33.7	55.3	57.3	1.6M	1.5×
	AR3PO + Target-Mix	86.2	27.8	68.2	75.9	34.4	56.2	58.1	1.9M	1.3×
	AR3PO - TargetVal	84.8	25.9	66.2	74.1	33.0	54.2	56.4	1.6M	1.5×
	L2RR	**86.8**	**28.7**	68.2	**76.7**	**35.1**	**59.1**	**59.1**	1.5M	1.6×
Qwen2.5-Math-7B										
DM	DAPO	82.9	34.3	79.1	71.7	35.4	43.6	57.8	13.1M	1.0×
	DAPO + Target-Mix	83.5	**35.2**	79.7	72.3	36.1	44.2	58.5	13.7M	1.0×
	DAPO - TargetVal	82.1	33.3	78.4	71.0	34.8	43.0	57.1	13.1M	1.0×
	GRESO	82.2	32.4	80.7	70.2	35.4	44.1	57.5	6.3M	2.1×
	GRESO + Target-Mix	83.0	33.3	**81.8**	71.1	36.2	44.9	58.4	6.7M	2.0×
	GRESO - TargetVal	81.6	31.5	80.1	69.3	34.8	43.4	56.8	6.3M	2.1×
	AR3PO	82.5	33.3	80.1	71.0	35.6	43.6	57.7	6.6M	2.0×
	AR3PO + Target-Mix	83.4	**35.2**	81.4	72.1	35.9	44.4	58.7	6.9M	1.9×
	AR3PO - TargetVal	82.0	32.4	79.4	70.1	34.7	42.8	56.9	6.6M	2.0×
	L2RR	**83.7**	34.3	**81.8**	**72.5**	**36.8**	**46.8**	**59.3**	6.0M	2.2×
OR1	DAPO	82.9	34.3	63.2	67.3	34.9	46.3	54.8	11.4M	1.0×
	DAPO + Target-Mix	82.9	35.2	63.9	67.6	35.6	46.9	55.4	11.8M	1.0×
	DAPO - TargetVal	82.1	33.3	62.2	66.5	33.6	45.8	53.9	11.4M	1.0×
	GRESO	82.3	35.2	64.5	66.8	36.5	45.7	55.2	3.4M	3.4×
	GRESO + Target-Mix	83.1	**36.1**	65.5	67.5	37.1	46.2	55.9	3.7M	3.1×
	GRESO - TargetVal	83.4	34.3	63.5	65.8	35.8	45.0	54.6	3.4M	3.4×
	AR3PO	82.4	**36.1**	64.5	66.5	36.6	45.7	55.3	3.6M	3.2×
	AR3PO + Target-Mix	83.2	**36.1**	65.5	67.3	37.2	46.5	56.0	3.8M	3.0×
	AR3PO - TargetVal	81.6	35.2	63.5	65.7	35.9	45.0	54.5	3.6M	3.2×
	L2RR	**83.6**	**36.1**	**65.9**	**67.9**	**37.6**	**49.7**	**56.8**	3.3M	3.5×

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We discussed the paper's contributions and scope

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations in the work.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: We provided all details.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provided all details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provided all details.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We provided all details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provided all details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provided all details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.

- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We checked it carefully.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discussed it in appendix.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We discussed it.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [N/A]

    Justification: NA

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: NA

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: NA

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: NA

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: NA

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.