The Mirage of Optimizing Training Policies: Monotonic Inference Policies as the Real Objective for LLM Reinforcement Learning

Jing Liang[1,3*] **Hongyao Tang**[2*] **Yi Ma**[4*] **Yancheng He**[3†] **Weixun Wang**[3†] **Xiaoyang Li**[3]
Wenbo Su[3] **Jinyi Liu**[5] **Yan Zheng**[1,5†] **Jianye Hao**[2,5†] **Bo Zheng**[3]
[1]1School of New Media and Communication, Tianjin University
[2]School of Computer Software, Tianjin University
[3]Alibaba Group
[4]School of Computer and Information Technology, Shanxi University
[5]MemoraX AI

Abstract

Reinforcement learning (RL) has gained growing attention in large language model (LLM) post-training, yet RL training remains fragile and can suffer from instability or collapse. One vital cause is *training-inference mismatch*: LLM adopts separate inference and training engines for generation efficiency and training precision, which in practice exhibits inconsistent probabilities for the same trajectories on training and inference sides, even with synchronized model parameters. This naturally induces a special type of off-policyness ever existing and poisoning the training. Prior works have made various efforts in addressing the off-policyness to stabilize the training policies under the mismatch. In this paper, we point out the *objective misalignment* neglected by existing works that an effective update to the policy in the training engine doesn't necessarily ensures the improvement of the inference policy, i.e., the one used in deployment. To this end, we propose a new policy optimization objective for LLM RL, named **Monotonic Inference Policy Improvement (MIPI)**. Following this principle, we introduce **Monotonic Inference Policy Update (MIPU)**, a two-step LLM RL framework that constructs sampler-referenced candidate updates and selectively accepts synchronized candidates using an inference-side gap proxy. Experiments conducted on two model scales under high mismatch show that **MIPU** improves average reasoning performance and training stability.

1 Introduction

With the emergence of reasoning-oriented models such as DeepSeek-R1 (Guo et al., 2025), Reinforcement Learning (RL) has become an increasingly important paradigm for post-training large language models (LLMs), particularly for improving instruction following, alignment, and reasoning capabilities (Yu et al., 2025; Hu et al., 2025; Qwen-Team, 2025). Due to the massive scale of LLM, modern LLM RL pipelines often separate rollout generation from gradient computation: responses are sampled by an inference engine (e.g. vLLM (Kwon et al., 2023) or SGLang (Zheng et al., 2024)), whereas log-probabilities and updates are computed by a training engine (e.g. FSDP (Zhao et al., 2023) or Megatron (Shoeybi et al., 2019)). Even under synchronized parameters, implementation

[*]Equal contribution
[†]Corresponding authors
This work was conducted while Jing Liang and Weixun Wang were at Alibaba Group.

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

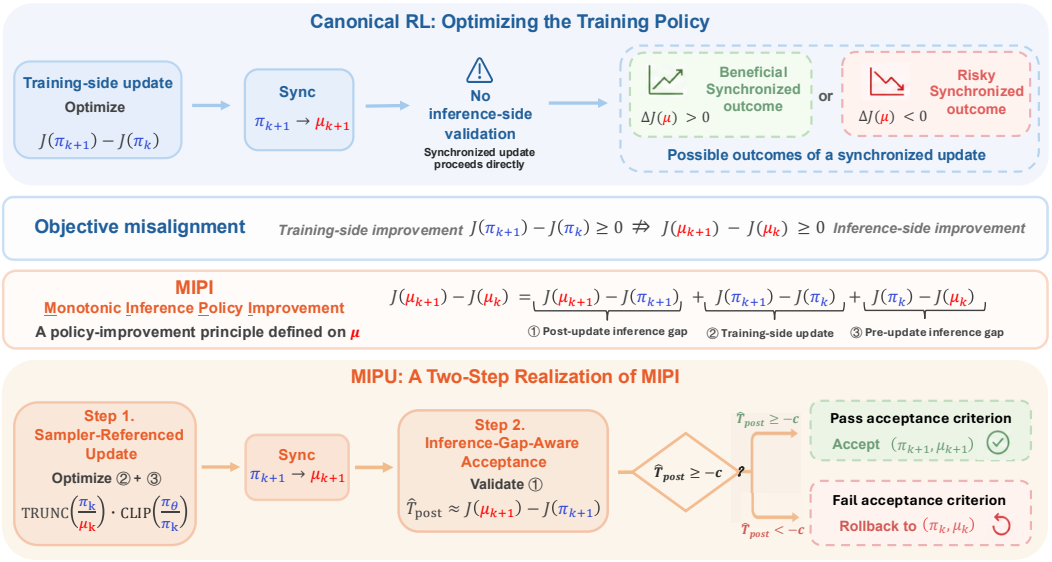


Figure 1: **Monotonic Inference Policy Update (MIPU) resolves the *Objective Misalignment* issue of LLM RL.** Canonical LLM RL accepts synchronized updates by a training-side objective, which does not necessarily imply improvement of the inference policy. Here, π and μ denote the training policy and inference policy respectively, c is a tolerance parameter accounting for proxy noise. To address this mismatch, we propose a new principle as monotonic improvement on the inference-policy trajectory (the **MIPI** principle). **MIPU** realizes this principle with two steps: **Step 1 optimizes Terms ②+③, while Step 2 estimates and validates Term ①, jointly covering all three terms in the MIPI decomposition.**

differences in precision, decoding, or serving backends can make the training policy π and inference policy μ assign different probabilities to the same trajectories. This inevitably creates the notorious *training-inference mismatch* issue (Yao et al., 2025; Liu et al., 2025).

Existing algorithmic and infrastructure remedies reduce this mismatch by correcting sampler-side ratios (Yao et al., 2025), filtering unstable samples (Liu et al., 2025), decaying learning rates (Zhang et al., 2026), or narrowing system-level discrepancies (Li et al., 2026a; Qi et al., 2025). These works aim to address the mismatch on the training side by improving the stability of the policy optimization process. However, due to the ever-existing training-inference mismatch, there is no guarantee that an effective improvement of the training policy necessarily has the same effect for the inference policy.

This suggests an *objective-level* view of training-inference mismatch. Since the policy used for rollout and deployment is induced by the inference engine, a model update should not be judged solely by whether it improves the training policy, i.e., the canonical optimization objective of LLM RL algorithms. Instead, the more natural question is:

Whether synchronizing this update actually produces a better inference policy ?

In the face of the objective misalignment revealed above, in this work we aim to break the canonicity of improving the training policy, and propose a new policy optimization principle for LLM RL, named **Monotonic Inference Policy Improvement (MIPI)**: the policy optimization objective should be established to improve the performance of the inference policy monotonically. Following this principle, we propose a two-step LLM RL framework, called **Monotonic Inference Policy Update (MIPU)**. As illustrated in Figure 1, the recipe of **MIPU** is to decompose the **MIPI** objective (i.e., $J(\mu_{k+1}) - J(\mu_k)$) into three terms, and optimize it via a sampler-referenced policy update step (i.e., the direct 'Step 1') and an inference-gap-aware acceptance step (i.e., the indirect 'Step 2'). For Step 1, the training engine performs a sampler-referenced policy update to generate a candidate model that theoretically ensures the halfway monotonicity $J(\pi_{k+1}) - J(\mu_k) \geq 0$. For Step 2, the synchronized candidate is evaluated through an inference-gap-aware acceptance criterion designed to promote $J(\mu_{k+1}) - J(\pi_{k+1}) \geq 0$.

In the experiments, we evaluate **MIPU** under FP8-quantized rollout, a high-mismatch setting where inference-side quantization amplifies the training-inference discrepancy. Across Qwen3-4B and

Qwen3-1.7B, **MIPU** achieves the best average performance and more stable training dynamics. Further analysis shows that Step 1 improves candidate updates, while Step 2 filters unreliable synchronized candidates using an inference-side criterion.

The contributions of this paper are summarized as follows:

- We propose a new learning objective for LLM RL, i.e., **MIPI**, which aligns with the purpose de facto and circumvents the training-inference mismatch issue.
- We propose a new LLM RL framework, i.e., **MIPU**, that realizes the **MIPI** principle in a two-step manner. **MIPU** allows flexible implementation of both the sampler-referenced policy update step and the inference-gap-aware update acceptance step.
- Experiments under FP8-quantized rollout show that **MIPU** improves both performance and training stability. And additional analysis verifies the complementary roles of Step 1 and Step 2 in candidate construction and inference-gap-aware acceptance.

2 Preliminaries

Reinforcement Learning for LLMs Reinforcement Learning (RL) (Sutton & Barto, 1998) is a general paradigm for sequential decision-making problems. RL has demonstrated its great ability of optimizing pre-trained LLMs (Trung et al., 2024; Jaech et al., 2024). In this paradigm, sequential token generation is modeled as a Markov Decision Process (MDP) $M = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where a **state** $s_t = (q, y_{1:t}) \in \mathcal{S}$ is the prompt with the output generated so far, and the **action** $a_t \in \mathcal{A}$ is the next token selected from the vocabulary $\mathcal{V}$. Hence, the **transition** $P(s_{t+1}|s_t, a_t)$ is deterministic in this context. An episode starts from a prompt s_0 (out of a predefined set $\mathcal{D}_0$) and terminates at an end-of-sequence token or by the maximum sequence length H. The **reward** $R(s_t, a_t)$ signal is issued by either a rule-based reward function or a learned reward model. In the scope of this paper, we consider the verifiable reward function. For any non-terminal timestep $t < T - 1$, $R(s_t, a_t)$ is 0; on completion, the terminal reward, denoted by $R(\tau)$ for the whole sequence, equals 1 if τ produces a correct and well-formatted answer and 0 otherwise.

The **policy** $\pi_\theta(a_t \mid s_t)$ in the MDP is the LLM itself, parameterized by θ, and it defines a probability distribution of next-token generation. We use $d_\tau^{\pi_\theta}$ to denote the distribution of the output sequence τ generated by π_θ and use $d_{s,a}^{\pi_\theta}, d_s^{\pi_\theta}$ for the state-action pairs (s, a) and the state respectively. The learning objective of an RL policy is to maximize the reward function, i.e., $\pi^* = \arg\max_{\pi_\theta} J(\pi_\theta)$, where $J(\pi) = \mathbb{E}_{\tau\sim\pi}[R(\tau)]$.

Monotonic Policy Improvement and Proximal Policy Gradient Optimizing RL policy according to the vanilla policy gradient $\nabla_\pi J(\pi)$ often turns out to be ineffective as the performance is sensitive to the step size. To this end, Schulman et al. (2015) revisit the *policy difference* (Kakade & Langford, 2002) for arbitrary two policies π and π' as defined below:

$$\Delta(\pi', \pi) := J(\pi') - J(\pi) = \mathbb{E}_{s,a\sim d^{\pi'}}\left[A^{\pi}(s, a)\right], \tag{1}$$

where $d^{\pi'}$ denotes the discounted state visitation distribution induced by policy π', and let $A^\pi(s, a)$ denote the advantage of taking action a at state s under policy π. Then, a natural idea is to ensure *monotonic policy improvement* during iterative policy optimization. Formally, given an old policy π_k, the goal of monotonic policy improvement is to find a new policy π_{k+1} such that $J(\pi_{k+1}) - J(\pi_k) \geq 0$, i.e., $\Delta(\pi_{k+1}, \pi_k) \geq 0$.

However, directly optimizing Equation 1 is infeasible because it depends on the visitation distribution of the updated policy, i.e., π_{k+1}. Schulman et al. (2015) propose Trust-Region Policy Optimization (TRPO) to approximate the exact objective difference by replacing the updated visitation distribution with the old-policy distribution and estimating with the importance sampling ratio:

$$\widetilde{\Delta}(\pi_{k+1}, \pi_k) := \mathbb{E}_{s\sim d^{\pi_k},\, a\sim\pi_k(\cdot|s)}\left[\frac{\pi_{k+1}(a \mid s)}{\pi_k(a \mid s)} A^{\pi_k}(s, a)\right]. \tag{2}$$

This local surrogate, denoted by $\tilde{\Delta}$, forms the basis of practical proximal policy optimization algorithms. TRPO maximizes such a surrogate subject to a KL-based trust-region constraint between the old and updated policies, while Proximal Policy Optimization (PPO) (Schulman et al., 2017) replaces the constrained optimization with a clipped probability-ratio objective that discourages excessively large policy updates. Based on PPO, Group Relative Policy Optimization (GRPO) (Shao et al., 2024) adopts a group-based advantage estimation which is free of learning the value function.

Training-Inference Mismatch in LLM RL As discussed above, modern LLM RL pipelines may induce different action distributions in the training and inference engines, even when they share synchronized model parameters. We model this training-inference mismatch by distinguishing the training policy π from the inference policy μ: at update step k, with the same parameters, π_k is the policy represented inside the training engine, whereas μ_k is the policy actually used for rollout generation and deployment in the inference engine. The training-inference mismatch manifests as $\pi \neq \mu$, even though they share the same model parameters. This mismatch thus results in new learning issues regarding non-stationarity and suboptimality different from the notorious ones in traditional RL (van Hasselt et al., 2018; Tang & Berseth, 2024; Wu et al., 2026). In the following of this paper, we will use the notations to explicitly highlight the distinction between training policies and inference policies. This distinction provides the basis for our method, where we revisit policy improvement from the perspective of the inference policy rather than the training-side surrogate.

3 Related Works

Reinforcement learning for LLM post-training RL has become a prominent paradigm for LLM post-training, especially for improving reasoning, coding, and long-horizon tasking capabilities. A major recent direction is reinforcement learning with verifiable rewards (RLVR), where outcome rewards can be obtained from automatically checkable signals such as mathematical answers, code execution, or task success. Representative works include GRPO (Shao et al., 2024), DAPO (Yu et al., 2025), and GSPO (Zheng et al., 2025), which develop scalable RL recipes for mathematical and reasoning tasks. In parallel, recent studies have also explored off-policy or data-reuse variants of reinforced fine-tuning (Liang et al., 2026; Zhan et al., 2026), highlighting the importance of correcting distributional mismatch during LLM RL. More recently, RL has also been adopted in agentic model training, where models learn from environment feedback in tool-use, coding, and multi-turn interaction settings. Examples include GLM-4.5 (GLM-Team, 2025), Kimi K2.5 (Kimi-Team, 2026), and ROME (Wang et al., 2026). Together, these works show the broad potential of RL for improving LLM capabilities, while also highlighting the fragility of RL training under long-horizon rollouts, complex optimization dynamics, and system-level implementation details.

Training-inference mismatch in LLM RL Recent work has identified training-inference mismatch as an important source of instability in modern LLM RL systems. A common line of algorithmic approaches incorporates sampler-side information into the training update. For example, TIS (Yao et al., 2025) uses the training-to-sampler probability ratio as a clipped correction weight, while MIS (Liu et al., 2025) and its subsequent trust-region formulation TRM (Li et al., 2026b) filters tokens or sequences with extreme mismatch signals. Learning rate decay (Zhang et al., 2026) provides another optimization-side remedy by reducing the update magnitude when mismatch-related instability emerges. In parallel, infrastructure-level approaches study how system design choices affect training-inference mismatch. For example, Qi et al. (2025) show that numerical precision can be a direct source of mismatch and propose using FP16 rollout to reduce the discrepancy between training and inference engines. Li et al. (2026a) further investigate low-precision RL systems and analyze how quantized rollout affects reasoning-oriented RL training. These works reduce mismatch from the system side, while sampler-aware methods such as TIS and MIS correct or filter the training update from the algorithm side. Our work is complementary: instead of only reducing mismatch, we study its objective-level consequence and ask whether a synchronized update should be accepted as the next inference policy.

4 The New Objective: Monotonic Policy Improvement on the Inference Side

As introduced in Section 2, training-inference mismatch separates the training policy π from the inference policy μ. As illustrated in Figure 1, a training-side improvement can still lead to a harmful inference policy after synchronization. Thus, when $\pi \neq \mu$, improving the training policy does not necessarily imply improving the inference policy:

$$J(\pi_{k+1}) - J(\pi_k) \geq 0 \;\not\Rightarrow\; J(\mu_{k+1}) - J(\mu_k) \geq 0. \tag{3}$$

We now instantiate this mismatch in the GRPO objective. In GRPO training, the training-side improvement is implemented by the clipped surrogate

$$\mathcal{J}_{\mathrm{GRPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\,\{y_i\}_{i=1}^{G}\sim\mu_k(\cdot|x)}\left[\frac{1}{G}\sum_{i=1}^{G}\min\left(r_i(\theta)\hat{A}_i^{\mu_k},\,\mathrm{clip}\left(r_i(\theta),1-\epsilon,1+\epsilon\right)\hat{A}_i^{\mu_k}\right)\right], \tag{4}$$

where $r_i(\theta) = \frac{\pi_\theta(y_i|x)}{\pi_k(y_i|x)}$ is the training-side probability ratio, and optimizing Equation 4 yields the next training policy $\pi_{k+1} := \pi_{\theta^\star}$. The superscript in $\hat{A}_i^{\mu_k}$ emphasizes that the GRPO advantage is not an explicit estimate of the old training-policy advantage A^{π_k}. Instead, it is computed from the response group sampled by the inference policy μ_k: GRPO normalizes the reward of y_i using the mean and standard deviation of rewards within $\{y_j\}_{j=1}^{G} \sim \mu_k(\cdot \mid x)$.

Therefore, beyond the policy-reference mismatch discussed above, GRPO introduces an additional mismatch-induced bias through its advantage estimation. Unlike the standard policy surrogate in Equation 2, whose weighting term is the old-policy advantage A^{π_k}, GRPO weights the update by the group-relative statistic $\hat{A}_i^{\mu_k}$ induced by inference rollouts. When $\mu_k \neq \pi_k$, this statistic is no longer aligned with the advantage of the old training policy. Thus, under training-inference mismatch, the standard GRPO update is affected by both a ratio-level mismatch and an advantage-level bias.

More importantly, these mismatches reveal a deeper objective-level misalignment. The standard GRPO surrogate optimizes a training-policy update, but the resulting synchronized inference policy is the one ultimately used for deployment. Under training-inference mismatch, the training-policy transition $\pi_k \to \pi_{k+1}$ may differ from the inference-policy transition $\mu_k \to \mu_{k+1}$. This motivates us to move beyond correcting individual terms in the training objective and instead ask *what policy improvement should mean when these two transitions differ*.

4.1 MIPI: Monotonic Inference Policy Improvement

To address this problem, we propose **Monotonic Inference Policy Improvement (MIPI)**, a policy improvement principle for LLM RL under training-inference mismatch.MIPI aims to align the update with the inference-policy transition rather than only with the training-side surrogate. Formally, MIPI takes the inference-policy improvement $J(\mu_{k+1}) - J(\mu_k)$ as the target quantity.

To make this principle explicit, we decompose the inference-policy improvement as follows:

$$J(\mu_{k+1}) - J(\mu_k) = \underbrace{J(\mu_{k+1}) - J(\pi_{k+1})}_{\text{① post-update inference gap}} + \underbrace{J(\pi_{k+1}) - J(\pi_k)}_{\text{② training-side update}} + \underbrace{J(\pi_k) - J(\mu_k)}_{\text{③ pre-update inference gap}}. \tag{5}$$

This decomposition provides the formal basis of **MIPI**. It separates the inference-policy improvement into a candidate construction part and a post-synchronization verification part. Accordingly, as illustrated in Figure 1, **MIPU** realizes **MIPI** through the following two steps:

MIPU: Two-Step Realization of MIPI

Step 1: Sampler-referenced policy update.

▷ The last two terms in Equation 5 combine into $J(\pi_{k+1}) - J(\mu_k)$. We approximate this quantity with a sampler-referenced surrogate to construct a candidate training policy relative to the rollout policy.

Step 2: Inference-gap-aware update acceptance.

▷ The first term $J(\mu_{k+1}) - J(\pi_{k+1})$ measures the post-update gap between the candidate training policy and its synchronized inference realization. We use this gap as an acceptance signal to decide whether the synchronized candidate should be accepted as the next inference policy. The rejected candidates are rolled back to the previous checkpoint.

4.2 Step 1: Sampler-Referenced Policy Update

Step 1 targets Terms ②+③ in Equation 5, which telescope to $J(\pi_{k+1}) - J(\mu_k)$. Equivalently, this grouped target can be written as $\Delta(\pi_{k+1}, \mu_k)$. Following Equation 2, viewing this target

relative to the sampler μ_k leads to the sampler-referenced weighting term $(\pi_\theta/\mu_k)A^{\mu_k}$. As discussed after Equation 4, GRPO uses the rollout-induced group-relative statistic $\hat{A}_i^{\mu_k}$, so this formulation keeps both the probability ratio and the advantage statistic aligned with the sampler.

By contrast, the standard GRPO surrogate in Equation 4 clips the training-side ratio π_θ/π_k, although the samples are generated by the inference policy μ_k.

Thus, the proximal constraint is centered at the old training policy π_k rather than at the sampler μ_k. Following the terminology of Yao et al. (2025), we refer to this direct sampler-referenced variant as PPO-IS, which instead clips the full trainer-to-sampler ratio $\rho_i(\theta) = \pi_\theta(y_i \mid x)/\mu_k(y_i \mid x)$. However, this full ratio conflates two different effects: the pre-update training-inference mismatch and the current training update:

$$\rho_i(\theta) = \underbrace{\frac{\pi_k(y_i \mid x)}{\mu_k(y_i \mid x)}}_{w_i^k:\ \text{pre-update mismatch}} \cdot \underbrace{\frac{\pi_\theta(y_i \mid x)}{\pi_k(y_i \mid x)}}_{r_i(\theta):\ \text{current update}}. \tag{6}$$

When w_i^k is already outside the clipping range, PPO-IS can over-constrain the update even when the current update ratio $r_i(\theta)$ remains close to one.

This factorization suggests separating the pre-update mismatch correction from the clipping of the current update. Following the same terminology, Vanilla-IS follows this idea by using the full correction weight w_i^k while clipping only $r_i(\theta)$, but the unbounded correction can introduce large variance. We therefore adopt the truncated variant TIS, which replaces w_i^k with $\bar{w}_i^k = \min(w_i^k, w_{\max})$ and still applies PPO clipping only to $r_i(\theta)$. Detailed comparisons among PPO-IS, Vanilla-IS, and TIS are provided in Appendix C.

Accordingly, Step 1 keeps a truncated sampler-referenced correction $\bar{w}_i^k$ while applying PPO-style clipping only to the current-update ratio $r_i(\theta)$. The Step-1 surrogate is

$$\mathcal{J}_{\mathrm{S1}}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\, \{y_i\}_{i=1}^{G}\sim\mu_k(\cdot|x)} \left[\frac{1}{G}\sum_{i=1}^{G} \bar{w}_i^k \min\left(r_i(\theta)\hat{A}_i^{\mu_k},\, \mathrm{clip}\left(r_i(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_i^{\mu_k} \right) \right]. \tag{7}$$

The truncated weight $\bar{w}_i^k$ retains the sampler-referenced correction while preventing large pre-update discrepancies between π_k and μ_k from dominating the gradient. The proximal constraint is still applied only to $r_i(\theta)$, the part of the ratio corresponding to the current training-side update $\pi_k \to \pi_\theta$. Optimizing Equation 7 yields a candidate training policy $\pi_{k+1} := \pi_{\theta^\star}$.

Step 1 provides a sampler-referenced proposal for the combined term $J(\pi_{k+1}) - J(\mu_k)$ in Equation 5. It does not, however, determine whether the synchronized inference policy μ_{k+1} realizes the gain proposed by the trainer. This post-update consistency is checked in Step 2.

4.3 Step 2: Inference-Gap-Aware Update Acceptance

After Step 1, the candidate training policy π_{k+1} is synchronized to the inference engine, producing μ_{k+1}. The remaining term in Equation 5 is the post-update inference gap $T_{\mathrm{post}} = J(\mu_{k+1}) - J(\pi_{k+1})$.

A negative value of T_{post} means that the inference policy underperforms its training-side counterpart after synchronization. In our proxy, this corresponds to an unfavorable mismatch pattern: the inference policy assigns insufficient probability to positive-advantage responses or excessive probability to negative-advantage responses relative to the training policy. Such updates are therefore unreliable, even if they appear beneficial on the training side.

By the performance difference identity, the direct form of the post-update gap $T_{\mathrm{post}} = \Delta(\mu_{k+1}, \pi_{k+1})$ requires the advantage under the candidate training policy π_{k+1}, which is not directly available in GRPO-style training. We therefore use the reverse identity, which anchors the advantage term at μ_{k+1} and makes it compatible with validation rollouts from the synchronized inference policy. This yields a validation-based proxy with an importance correction from μ_{k+1} to π_{k+1}:

$$\begin{aligned} T_{\mathrm{post}} &= -\Delta(\pi_{k+1}, \mu_{k+1}) = -\mathbb{E}_{s,a\sim d^{\pi_{k+1}}}\left[A^{\mu_{k+1}}(s,a)\right] \\ &\rightsquigarrow \widehat{T}_{\mathrm{post}} = -\mathbb{E}_{x\sim\mathcal{D}_{\mathrm{val}},\, y_i\sim\mu_{k+1}}\left[\rho_i \hat{A}_i^{\mu_{k+1}}\right]. \end{aligned} \tag{8}$$

Algorithm 1 Monotonic Inference Policy Update

[Input]: Base model π_{base}, group size G, clipping range ϵ, tolerance c
[Init]: Set trainer policy $\pi_{\theta_0} = \pi_{\text{base}}$, optimizer state Ω_0, and inference policy $\mu_0 = \text{Sync}(\pi_{\theta_0})$
for $k = 0, 1, 2, \ldots$ **do**
  Save checkpoint $(\theta_k, \Omega_k, \mu_k)$
  ▷ Step 1: sampler-referenced proposal
  Collect a training batch $\mathcal{B} \sim \mathcal{D}$ and collect G samples per prompt from μ_k
  Compute rewards, group-relative advantages $\hat{A}^{\mu_k}$, and mismatch weights $w^k = \pi_k / \mu_k$ on the training rollouts
  Update trainer parameters and optimizer state by maximizing the Step-1 surrogate in Equation 7
  Set $\pi_{k+1} \leftarrow \pi_{\theta^\star}$ and $\mu_{k+1} \leftarrow \text{Sync}(\pi_{k+1})$
  ▷ Step 2: monotonic inference acceptance
  Sample a validation batch $\mathcal{B}_{\text{val}} \sim \mathcal{D}_{\text{val}}$ and roll out G responses per prompt from μ_{k+1}
  Compute rewards, advantages $\hat{A}^{\mu_{k+1}}$, and ratios ρ_i on the validation rollouts
  Estimate the post-update inference gap $\widehat{T}_{\text{post}}$
  if $\widehat{T}_{\text{post}} < -c$ **then**
    Reject the update: restore $\theta_{k+1} \leftarrow \theta_k$, $\Omega_{k+1} \leftarrow \Omega_k$, and $\mu_{k+1} \leftarrow \mu_k$
  end if
end for

Here, $\hat{A}_i^{\mu_{k+1}}$ is the group-relative advantage computed from validation responses sampled by μ_{k+1}. The symbol $\rightsquigarrow$ emphasizes that $\widehat{T}_{\text{post}}$ is a stable proxy motivated by the reverse identity, rather than an exact estimator of T_{post}. Since the reverse form takes expectation under π_{k+1} while validation responses are sampled from μ_{k+1}, we introduce an importance weight. For each completion y_i, we use the length-normalized sequence ratio

$$\rho_i = \exp\left(\frac{1}{T_i}\sum_{t=1}^{T_i} \log \frac{\pi_{k+1}(y_{i,t} \mid x, y_{i,<t})}{\mu_{k+1}(y_{i,t} \mid x, y_{i,<t})}\right). \tag{9}$$

This length-normalized ratio serves as a more stable importance weight for approximating the reverse-form expectation, avoiding the high variance of the full sequence-level importance sampling ratio.

The proxy in Equation 8 serves as a risk signal. Since $\widehat{T}_{\text{post}} = -\mathbb{E}_{\mu_{k+1}}[\rho_i \hat{A}_i^{\mu_{k+1}}]$, a more negative value indicates that ρ_i is positively associated with the validation advantage statistic. In other words, compared with the synchronized inference policy, the candidate training policy assigns relatively higher probability to positive-advantage responses or lower probability to negative-advantage responses. This suggests that the training-side proposal may not be realized by the inference engine. It does not prove that the inference policy is worse than the previous one, but marks the update as unreliable under training-inference mismatch.

Therefore, we use the post-update gap proxy as an acceptance test: $\widehat{T}_{\text{post}} \geq -c$, where $c \geq 0$ is a tolerance parameter for noise in the proxy. This rule tolerates mild negative proxy values but rejects candidates with a clear negative post-update gap signal. If passing the test, we accept the candidate and set $\pi_k \leftarrow \pi_{k+1}$ and $\mu_k \leftarrow \mu_{k+1}$. Otherwise, we reject the candidate and roll back both the trainer state and the inference engine to the previous checkpoint. The implementation details are provided in Section A.2.

Overall, Step 1 turns the trainer update into a sampler-referenced candidate, and Step 2 filters candidates with a large post-update inference gap. Algorithm 1 summarizes the full **MIPU** procedure, including checkpointing, synchronization, validation, and rollback. This mechanism does not provide a formal monotonic-improvement guarantee, but it reduces the risk of accumulating updates whose gains are not realized by the inference policy.

5 Experiments

We evaluate **MIPU** under FP8-quantized rollout, where quantized inference amplifies training-inference mismatch. Our experiments are organized around three research questions:

- **RQ1:** Does **MIPU** improve performance and training stability under high training-inference mismatch? (Section 5.2)
- **RQ2:** Do Step 1 and Step 2 play complementary roles in **MIPU**? (Section 5.3)
- **RQ3:** Is Step 2 effective because of its inference-gap signal, or simply because it rejects more updates? (Section 5.4)

5.1 Experimental Setup

Training. We train Qwen3-1.7B and Qwen3-4B under the *FP8-quantized rollout*, where rollout generation uses FP8-quantized inference and therefore induces a high training-inference mismatch. All analyses beyond the main results are conducted on Qwen3-4B. The detailed hyperparameter choices are provided in Section A.1. For training data, we use DAPO-Math-17 (Yu et al., 2025) and DeepMath-103K (He et al., 2025b) as our training corpora. Qwen3-1.7B is trained on `5759` examples filtered from DAPO, while Qwen3-4B is trained on `1491` examples filtered from DeepMath. The filtering retains problems that the base model can solve with non-trivial but non-saturated success rates, ensuring that the training set provides meaningful reward variation for RL. Throughout the paper, training steps in figures count attempted updates, including accepted and rejected updates.

Evaluation. We evaluate all methods on five mathematical reasoning benchmarks: MATH-500 (Hendrycks et al., 2021), AIME24 (MAA, 2024), AMC23 (MAA, 2023), Minerva (Lewkowycz et al., 2022), and OlympiadBench (He et al., 2024). We report pass@1 accuracy for all benchmarks. For the small benchmarks AIME24 and AMC23, we use avg@16 to reduce evaluation variance and improve evaluation reliability. All reported scores are computed using the same evaluation protocol across methods. For diagnostics, we report the inference-training K3-KL (denoted as *Mismatch-K3* in the following figures), computed on rollout tokens sampled from μ. Specifically, with $d_t = \log \pi(y_t \mid s_t) - \log \mu(y_t \mid s_t)$, we average $\exp(d_t) - d_t - 1$, which estimates $D_{\mathrm{KL}}(\mu \| \pi)$.

Baselines. Our work focuses on stabilizing the RL objective under training-inference mismatch, which is orthogonal to system-level infrastructure optimization. We therefore do not compare with infrastructure methods. For the main comparison, we compare against the standard GRPO baseline and two optimization-side stabilization baselines, MIS and LR-decay. Since **MIPU** instantiates Step 1 with a TIS-style correction, we analyze TIS separately as the Step 1-only variant in Sec. 5.3. The implementation details are in Appendix A.

5.2 Main Results

Table 1: **Pass@1 accuracy (%) on five mathematical benchmarks under FP8-quantized rollout.** Bold denotes the best result for each model and benchmark, including the average. *Stable* indicates whether the method avoids collapse or sharp degradation during continued training.

Model	Method	MATH	AIME	Olympiad	Minerva	AMC23	Avg.	*Stable*
Qwen3-4B	Baseline	89.34	42.00	64.89	43.39	82.50	64.42	✗
	MIS	90.95	38.44	62.50	44.12	81.09	63.42	✗
	LR-decay	90.34	**44.00**	67.26	43.75	82.97	65.66	✗
	Ours	**91.15**	43.56	**67.86**	**45.96**	**85.00**	**66.71**	✓
Qwen3-1.7B	Baseline	83.10	**25.33**	56.55	31.68	57.66	50.86	✗
	MIS	81.29	24.67	58.33	**34.19**	60.16	51.73	✗
	LR-decay	82.09	26.00	58.93	28.68	**65.47**	52.23	✗
	Ours	**86.52**	24.67	**59.52**	33.82	65.31	**53.97**	✓

Table 1 reports the main results under FP8-quantized rollout, the high-mismatch setting where quantized inference amplifies the gap between the training policy π and the inference policy μ. On both model scales, **MIPU** achieves the best average performance, reaching 66.71% on Qwen3-4B and

53.97% on Qwen3-1.7B. For Qwen3-4B, the gains are most visible on AMC23 (85.00%) and Minerva (45.96%); for Qwen3-1.7B, the gains are most visible on MATH-500 (86.52%) and OlympiadBench (59.52%). Beyond peak accuracy, the Stable column in Table 1 shows that **MIPU** remains stable until the end of training, while several baselines suffer from collapse or sharp degradation.

It is worth noting that some baselines can achieve competitive intermediate performance, but their gains are not sustained under continued training. This distinction is important in high-mismatch RL, where a transient peak may be followed by rapid degradation once mismatch-driven errors accumulate. As shown in Figure 2, GRPO, MIS, and LR-decay degrade after reaching intermediate peaks on Qwen3-4B under FP8-quantized rollout, whereas **MIPU** maintains a stable trajectory after reaching high performance. Together with Table 1, these results demonstrate improvements in both peak accuracy and long-horizon training stability.

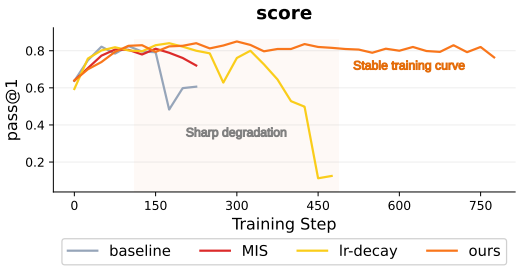


Figure 2: **Training performance on Qwen3-4B under FP8-quantized rollout.**

Beyond the main FP8 setting, MIPU improves over the corresponding GRPO on code-generation tasks, a Llama-family model, FP16 and INT8 rollouts, and the SGLang backend (Appendix B).

5.3 Ablation Studies

We isolate the contributions of the two stages on Qwen3-4B under FP8-quantized rollout by comparing GRPO, Step 1 only, Step 2 only, and the full **MIPU** framework.

Table 2: **Ablation results on Qwen3-4B under FP8-quantized rollout.** We report pass@1 accuracy (%); bold denotes the best result in each column.

Method	MATH 500	AIME 24	Olympiad	Minerva	AMC23	Avg.
Baseline	89.34	42.00	64.89	43.39	82.50	64.42
+ Step 1	90.34	41.11	**68.45**	44.85	82.03	65.36
+ Step 2	90.34	40.44	64.88	43.38	75.00	62.81
Ours	**91.15**	**43.56**	67.86	**45.96**	**85.00**	**66.71**

Table 2 and Figure 3 show that Step 1 and Step 2 address different failure modes. The baseline update is centered at the training policy π_k even though rollouts are sampled from μ_k, which makes the synchronized inference policy vulnerable to FP8-induced mismatch and can eventually lead to collapse. Step 2 alone can prevent collapse by rejecting candidates with large negative $\widehat{T}_{\text{post}}$, but it cannot improve the quality of the underlying baseline update. Thus, when the baseline rarely proposes useful candidates, Step 2 mostly keeps the previous inference policy rather than creating further improvement. Its acceptance rule improves safety, but it cannot compensate for poor candidate quality when the proposal update itself remains biased by the training-inference mismatch.

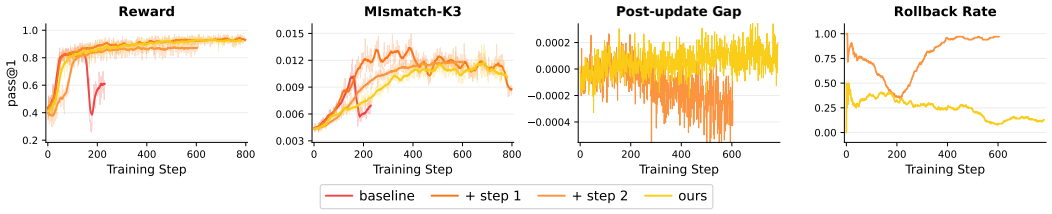
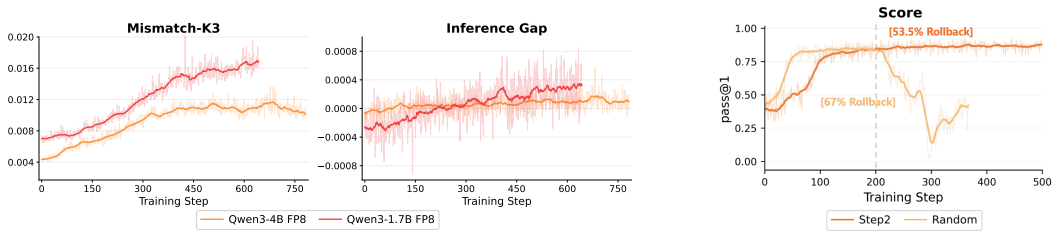


Figure 3: **Ablation training dynamics on Qwen3-4B under FP8-quantized rollout.** From left to right: training score, inference-training K3-KL, post-update gap proxy $\widehat{T}_{\text{post}}$, and rollback rate over a 100-step moving window.

Step 1 improves learning by constructing sampler-referenced candidate updates before synchronization. However, Step 1 alone accepts every synchronized candidate as the next inference policy, so

mismatch fluctuations can still accumulate. The full method combines the two roles: Step 1 produces better candidate updates, and Step 2 filters candidates whose post-update inference gap suggests that the gain may not be realized by μ_{k+1}.

5.4 Analysis of Step 2: Inference-Gap-Aware Acceptance



(a) Inference-training K3-KL and inference gap.

(b) Step 2 vs. random rollback.

Figure 4: **Step 2 analysis under FP8-quantized rollout.** (a) Inference-training K3-KL and post-update gap proxy $\widehat{T}_{\text{post}}$ for Qwen3-1.7B and Qwen3-4B. (b) Training performance of Step 2-only and random rollback on Qwen3-4B.

Post-update gap as an inference-side signal. As shown in Figure 4a, Qwen3-1.7B exhibits larger training-inference mismatch and stronger fluctuations in the post-update gap proxy $\widehat{T}_{\text{post}}$ than Qwen3-4B under FP8-quantized rollout. For both models, mismatch increases during training, while $\widehat{T}_{\text{post}}$ shifts upward from low values. These observations motivate evaluating the proxy as a criterion for candidate acceptance, which we examine through the random rollback comparison below.

Signal-conditioned rollback. We test whether the stability of Step 2 can be explained solely by accepting fewer updates. We compare the Step 2-only variant with random rollback, setting the rejection probability to 70% based on its approximate rollback rate over the first 500 steps. In the window shown in Figure 4b, random rollback rejects more candidates than Step 2 (67.0% versus 53.5%), yet collapses after an initial peak, whereas Step 2 maintains a stable score trajectory. This comparison indicates that reducing the number of accepted updates alone is insufficient to explain the observed stability, supporting the use of the post-update gap proxy $\widehat{T}_{\text{post}}$ to guide acceptance.

We further analyze the sensitivity of the acceptance tolerance in Appendix D, showing that Step 2 requires calibrated acceptance rather than simply stricter rejection.

6 Conclusion

We revisit training-inference mismatch in LLM reinforcement learning from an objective-level perspective. Rather than treating mismatch only as a system discrepancy to be reduced, we show that it changes the object of policy improvement: standard RL updates optimize the training policy, while deployment depends on the inference policy. Based on this view, we formulate Monotonic Inference Policy Improvement (**MIPI**) and propose Monotonic Inference Policy Update (**MIPU**), which realizes **MIPI** through sampler-referenced candidate construction and inference-gap-aware candidate acceptance. Experiments under FP8-quantized rollout show that **MIPU** improves average reasoning performance and training stability, and further analysis confirms the rationality of Step 2. These results suggest that training-inference mismatch should not be treated only as a low-level system discrepancy, but also as a change in the objective of policy improvement.

Limitations. Due to significant computational constraints, our experiments are currently limited to moderate-scale models. Future work should examine whether the same mismatch patterns also appear in larger models and more diverse RL training systems. In addition, Step 2 should be viewed as a flexible design space rather than a fixed implementation. Our validation-based post-update gap proxy provides one effective instantiation for evaluating synchronized candidates from the deployment side. Future work may further improve this component with more efficient estimators, stronger acceptance criteria, or optimization schemes that directly leverage the post-update inference gap.

Acknowledgments

This work is supported by the National Key Research and Development Program of China (Grant No. 2026ZD0126104), National Natural Science Foundation of China (Grant Nos. 62533021, 62422605, 92370132), Tianjin Natural Science Foundation Project (No. 25JCQNJC01720), MoE Key Laboratory of Brain-inspired Intelligent Perception and Cognition, University of Science and Technology of China (No. 2521006), and Fundamental Research Program of Shanxi Province (Serial No. 202503021212091).

We thank Ju Huang for her valuable support and assistance during this work.

References

GLM-Team. GLM-4.5: Agentic, reasoning, and coding (ARC) foundation models. *arXiv preprint arXiv:2508.06471*, 2025.

Guo, D., Yang, D., Zhang, H., Song, J., Wang, P., Zhu, Q., Xu, R., Zhang, R., Ma, S., Bi, X., et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081): 633–638, 2025.

He, C., Luo, R., Bai, Y., Hu, S., Thai, Z., Shen, J., Hu, J., Han, X., Huang, Y., Zhang, Y., Liu, J., Qi, L., Liu, Z., and Sun, M. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *ACL*, pp. 3828–3850, 2024.

He, J., Liu, J., Liu, C. Y., Yan, R., Wang, C., Cheng, P., Zhang, X., Zhang, F., Xu, J., Shen, W., Li, S., Zeng, L., Wei, T., Cheng, C., An, B., Liu, Y., and Zhou, Y. Skywork open reasoner 1 technical report. *arXiv preprint arXiv:2505.22312*, 2025a.

He, Z., Liang, T., Xu, J., Liu, Q., Chen, X., Wang, Y., Song, L., Yu, D., Liang, Z., Wang, W., Zhang, Z., Wang, R., Tu, Z., Mi, H., and Yu, D. DeepMath-103K: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *arXiv preprint arXiv:2504.11456*, 2025b.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset. In *NeurIPS*, 2021.

Hu, J., Zhang, Y., Han, Q., Jiang, D., Zhang, X., and Shum, H.-Y. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. In *NeurIPS*, 2025.

Jaech, A., Kalai, A., Lerer, A., Richardson, A., El-Kishky, A., Low, A., Helyar, A., Madry, A., Beutel, A., Carney, A., et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Jain, N., Han, K., Gu, A., Li, W., Yan, F., Zhang, T., Wang, S., Solar-Lezama, A., Sen, K., and Stoica, I. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *ICLR*, 2025.

Kakade, S. M. and Langford, J. Approximately optimal approximate reinforcement learning. In *ICML*, pp. 267–274, 2002.

Kimi-Team. Kimi k2.5: Visual agentic intelligence. *arXiv preprint arXiv:2602.02276*, 2026.

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J. E., Zhang, H., and Stoica, I. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Lewkowycz, A., Andreassen, A., Dohan, D., Dyer, E., Michalewski, H., Ramasesh, V., Slone, A., Anil, C., Schlag, I., Gutman-Solo, T., et al. Solving quantitative reasoning problems with language models. In *NeurIPS*, 2022.

Li, Y., Elangovan, R., Dong, X., Panda, P., and Khailany, B. QuRL: Low-precision reinforcement learning for efficient reasoning. In *ICLR*, 2026a.

Li, Y., Liu, J., Xu, J., Tong, Y., Li, Z., and Wang, B. Trust region masking for long-horizon LLM reinforcement learning. In *ICML*, 2026b.

Liang, J., Liu, J., Ma, Y., Tang, H., ZHENG, Y., Hu, S., BAI, L., and HAO, J. Squeeze the soaked sponge: Efficient off-policy RFT for large language model. In *ICLR*, 2026.

Liu, J., Li, Y., Fu, Y., Wang, J., Liu, Q., and Shen, Y. When speed kills stability: Demystifying RL collapse from the training-inference mismatch, September 2025. URL https://richardli.xyz/rl-collapse.

MAA. American mathematics contest 12 (amc 12), November 2023. URL https://artofproblemsolving.com/wiki/index.php/AMC_12_Problems_and_Solutions. Accessed: 2026-04-30.

MAA. American invitational mathematics examination (aime), February 2024. URL https://artofproblemsolving.com/wiki/index.php/2024_AIME_I. Accessed: 2026-04-30.

Qi, P., Liu, Z., Zhou, X., Pang, T., Du, C., Lee, W. S., and Lin, M. Defeating the training-inference mismatch via fp16. *arXiv preprint arXiv:2510.26788*, 2025.

Qwen-Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Schulman, J., Levine, S., Abbeel, P., Jordan, M. I., and Moritz, P. Trust region policy optimization. In *ICML*, volume 37, pp. 1889–1897, 2015.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Zhang, M., Li, Y. K., Wu, Y., and Guo, D. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., and Catanzaro, B. Megatron-LM: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Sutton, R. S. and Barto, A. G. *Reinforcement learning - an introduction*. Adaptive computation and machine learning. MIT Press, 1998. ISBN 978-0-262-19398-6.

Tang, H. and Berseth, G. Improving deep reinforcement learning by reducing the chain effect of value and policy churn. In *NeurIPS*, 2024.

Trung, L., Zhang, X., Jie, Z., Sun, P., Jin, X., and Li, H. Reft: Reasoning with reinforced fine-tuning. In *ACL*, pp. 7601–7614, 2024.

van Hasselt, H., Doron, Y., Strub, F., Hessel, M., Sonnerat, N., and Modayil, J. Deep reinforcement learning and the deadly triad. *arXiv preprint arXiv:1812.02648*, 2018.

Wang, W., Xiong, S., Chen, G., Gao, W., Guo, S., He, Y., Huang, J., Liu, J., Li, Z., Li, X., et al. Reinforcement learning optimization for large-scale learning: An efficient and user-friendly scaling library. *arXiv preprint arXiv:2506.06122*, 2025.

Wang, W., Xu, X., An, W., Dai, F., Gao, W., He, Y., Huang, J., Ji, Q., Jin, H., Li, X., Li, Y., Li, Z., Lin, S., Liu, J., Liu, Z., Luo, T., Muhtar, D., Qu, Y., Shi, J., Sun, Q., Tan, Y., Tang, H., Wang, R., Wang, Y., Wang, Z., Wu, Y., Xiong, S., Xu, B., Xu, X., Xu, Y., Zhang, Q., Zhang, X., Zhao, H., Zhao, J., Zhao, S., Zheng, B., Zheng, J., Zheng, S., Zhu, Y., Cai, M., Cao, K., Chen, X., Dai, Y., Du, L., Feng, T., He, T., Hu, J., Hu, Y., Jiang, Z., Li, C., Li, X., Liang, J., Lin, X., Liu, C., Liu, Z., Lv, Z., Mi, H., Mo, Y., Ni, J., Pei, S., Shen, J., Song, X., Wang, C., Wang, C., Wang, K., Wang, P., Wang, T., Wang, W., Xiao, K., Xu, M., Xu, T., Ya, N., Yang, S., Ye, J., Zang, Y., Zhang, D., Zhang, J., Zheng, B., Deng, W., Pan, L., Qu, L., Su, W., Wang, J., Wang, W., Wei, H., Wu, M., Yu, C., Zhao, B., Zheng, Z., and Zheng, B. Let it flow: Agentic crafting on rock and roll, building the ROME model within an open agentic learning ecosystem. *arXiv preprint arXiv:2512.24873*, 2026.

Wu, Z., Tang, H., Ma, Y., Liu, J., Zheng, Y., and Hao, J. The rank and gradient lost in non-stationarity: Sample weight decay for mitigating plasticity loss in reinforcement learning. In *ICLR*, 2026.

Yao, F., Liu, L., Zhang, D., Dong, C., Shang, J., and Gao, J. Your efficient rl framework secretly brings you off-policy rl training, August 2025. URL https://fengyao.notion.site/off-policy-rl.

Yu, Q., Zhang, Z., Zhu, R., Yuan, Y., Zuo, X., Yue, Y., Fan, T., Liu, G., Liu, L., Liu, X., Lin, H., Lin, Z., Ma, B., Sheng, G., Tong, Y., Zhang, C., Zhang, M., Zhang, W., Zhu, H., Zhu, J., Chen, J., Chen, J., Wang, C., Yu, H., Dai, W., Song, Y., Wei, X., Zhou, H., Liu, J., Ma, W., Zhang, Y., Yan, L., Qiao, M., Wu, Y., and Wang, M. DAPO: an open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Zhan, R., Li, Y., Wang, Z., Qu, X., Liu, D., Shao, J., Wong, D. F., and Cheng, Y. ExGRPO: Learning to reason from experience. In *ICLR*, 2026.

Zhang, Y., Li, Y., Liu, J., Xu, J., Li, Z., Liu, Q., and Li, H. Beyond precision: Training-inference mismatch is an optimization problem and simple lr scheduling fixes it. *arXiv preprint arXiv:2602.01826*, 2026.

Zhao, Y., Gu, A., Varma, R., Luo, L., Huang, C.-C., Xu, M., Wright, L., Shojanazeri, H., Ott, M., Shleifer, S., Desmaison, A., Balioglu, C., Damania, P., Nguyen, B., Chauhan, G., Hao, Y., Mathews, A., and Li, S. Pytorch fsdp: Experiences on scaling fully sharded data parallel. *Proc. VLDB Endow.*, 16(12):3848–3860, August 2023.

Zheng, C., Liu, S., Li, M., Chen, X.-H., Yu, B., Gao, C., Dang, K., Liu, Y., Men, R., Yang, A., Zhou, J., and Lin, J. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

Zheng, L., Yin, L., Xie, Z., Sun, C., Huang, J., Yu, C. H., Cao, S., Kozyrakis, C., Stoica, I., Gonzalez, J. E., Barrett, C., and Sheng, Y. SGLang: Efficient execution of structured language model programs. In *NeurIPS*, 2024.

A Implementation Details

A.1 Hyperparameters

The major hyperparameter choices are shown in Table 3.

Table 3: **Hyperparameter setups.**

Parameter	Value
Decoding Configuration	
training temperature	1.0
evaluation temperature	0.7
top-p	1.0
top-k	-1
response_length	8192
prompt_length	512
Training Configuration	
learning_rate	1e-6
num_return_sequences_in_group	8
train_batch_size	64
dual_clip_loss	true
kl_loss_coef	0.001
gradient_accumulation_steps	32
Infrastructure Configuration	
GPUs	H100 * 8
RL framework	ROLL (Wang et al., 2025)
Training engine	Megatron
Inference engine	vLLM

A.2 Training Implementation

Baseline. We use vanilla GRPO with the same rollout, reward, and optimization settings as other methods. In implementation, we additionally use the dual-clipped policy loss. For a token-level ratio $r_{i,t}(\theta) = \pi_\theta(y_{i,t} \mid x, y_{i,<t}) / \pi_k(y_{i,t} \mid x, y_{i,<t})$ and group-relative advantage $\hat{A}_i$, the standard clipped objective is $\ell_{i,t}^{\text{clip}} = \min\Big(r_{i,t}(\theta)\hat{A}_i, \text{clip}(r_{i,t}(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_i\Big)$. The dual-clipped objective is then

$$\ell_{i,t}^{\text{dc}} = \begin{cases} \ell_{i,t}^{\text{clip}}, & \hat{A}_i \geq 0, \\ \max\Big(\ell_{i,t}^{\text{clip}}, (1+2\epsilon)\hat{A}_i\Big), & \hat{A}_i < 0, \end{cases}$$

where we set $\epsilon = 0.2$. The baseline maximizes the average dual-clipped GRPO objective over sampled responses.

Ours. For Step 1, we instantiate the sampler-referenced update with token-level truncated importance weights. Specifically, for each generated token, we compute the training-to-inference mismatch weight $w_{i,t}^k$ and set $w_{\max} = 2$. The weight is applied after the dual-clipped GRPO objective is computed, yielding the Step 1 token-level objective

$$\mathcal{J}_{\text{S1}} = \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{T_i}\sum_{t=1}^{T_i}\bar{w}_{i,t}^k\,\ell_{i,t}^{\text{dc}}\right].$$

Vanilla-IS follows the same implementation except that it uses the untruncated weight $w_{i,t}^k$ instead of the truncated $\bar{w}_{i,t}^k$.

For Step 2, after the candidate training policy π_{k+1} is synchronized to the inference engine as μ_{k+1}, we estimate the post-update inference gap proxy $\widehat{T}_{\text{post}}$ on a validation batch. The candidate is

accepted if $\widehat{T}_{\text{post}} \geq -c_t$, where $c_t \geq 0$ is a dynamic acceptance tolerance parameter. Motivated by the observation in Figure 4a that $\widehat{T}_{\text{post}}$ is initially low and then gradually increases under FP8 rollout, we use a larger tolerance in the early stage to avoid over-rejecting useful candidates. The threshold is linearly annealed during the first 100 steps:

$$c_t = c_{\text{start}} + \frac{\min(t, 100)}{100} \left(c_{\text{end}} - c_{\text{start}}\right).$$

For Qwen3-1.7B, we set $c_{\text{start}} = 4 \times 10^{-4}$ and $c_{\text{end}} = 1 \times 10^{-4}$. For Qwen3-4B, we set $c_{\text{start}} = 1 \times 10^{-4}$ and $c_{\text{end}} = 0$. After the first 100 steps, the tolerance is fixed at c_{end}. We further analyze the effect of different tolerance choices in Appendix D.

MIS. We implement MIS as a token-level mismatch-filtering baseline. For each token, we compute the same training-to-inference ratio $w_{i,t}^k$. Tokens whose mismatch weight exceeds the upper bound are masked out:

$$m_{i,t}^k = \mathbb{1}\left[w_{i,t}^k \leq \bar{w}\right], \qquad \bar{w} = 2.$$

The MIS objective applies this mask to the dual-clipped GRPO objective:

$$\mathcal{J}_{\text{MIS}} = \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{T_i}\sum_{t=1}^{T_i} m_{i,t}^k \, \ell_{i,t}^{\text{dc}}\right].$$

LR-decay. We implement LR-decay following the original decay rule. Let η_t denote the learning rate at training step t. At every decay interval T_{decay}, the learning rate is halved until it reaches a lower bound η_∞:

$$\eta_t = \begin{cases} \max(\eta_{t-1}/2, \eta_\infty), & t \equiv 0 \pmod{T_{\text{decay}}}, \\ \eta_{t-1}, & \text{otherwise.} \end{cases}$$

The model parameters are then updated using η_t. Following the authors' guidance, we set $T_{\text{decay}} = 249$ for Qwen3-1.7B and $T_{\text{decay}} = 234$ for Qwen3-4B under FP8 rollout.

A.3 Computational Overhead

We profile the wall-clock overhead of **MIPU** over 800 attempted updates.

MIPU reuses the validation rollouts of each accepted candidate as training data for the next iteration. In our timing logs, rollout generation accounts for 80.7% of the total validation time. This reuse therefore avoids most of the additional generation cost for accepted candidates. The reported wall-clock time includes validation, checkpointing, synchronization, and rollback.

Under the same budget of 800 attempted updates, **MIPU** takes 63.04 hours, compared with 53.20 hours for the reference setting without the additional Step 1 and Step 2 operations. This corresponds to an additional 9.84 hours, or an 18.5% wall-clock overhead. During this run, **MIPU** accepts 617 candidate updates and rolls back 183 (22.9%).

B Supplementary Performance Evaluation

We extend the evaluation beyond the main setting to coding, a Llama-family model, alternative rollout precisions, and a different inference backend. Each experiment compares **MIPU** with the GRPO baseline.

B.1 Evaluation on Coding

We train Qwen3-1.7B on Skywork-OR1-RL-Data (He et al., 2025a) and evaluate it on LiveCodeBench(8/1/24–2/1/25) (Jain et al., 2025). To accommodate longer coding prompts, we increase the maximum prompt length to 1,600 tokens while retaining the other hyperparameters.

As shown in Table 4, **MIPU** improves the score from 21.22 to 25.54, a gain of 4.32 percentage points. This result extends the evidence for inference-aware policy optimization beyond mathematical reasoning to code generation.

Table 4: **Coding performance (%) on LiveCodeBench with Qwen3-1.7B. Bold** denotes the best result.

Method	LiveCodeBench
Baseline	21.22
Ours	**25.54**

B.2 Evaluation on a Llama-Family Model

To examine whether the gains extend beyond Qwen3, we evaluate DeepSeek-R1-Distill-Llama-8B on the same five mathematical reasoning benchmarks used in the main experiments.

Table 5: **Pass@1 accuracy (%) on five mathematical benchmarks with DeepSeek-R1-Distill-Llama-8B. Bold** denotes the best result in each column.

Method	MATH 500	AIME 24	Olympiad	Minerva	AMC23	Avg.
Baseline	87.12	39.78	64.29	37.87	85.94	63.00
Ours	**89.34**	**41.11**	**67.26**	**39.34**	**87.50**	**64.91**

Table 5 shows that **MIPU** improves performance on all five benchmarks, increasing the average score from 63.00 to 64.91. The gains range from 1.33 percentage points on AIME24 to 2.97 on OlympiadBench, providing evidence that the improvement is not confined to the Qwen3 models used in the main experiments.

B.3 Evaluation Across Rollout Precisions

We further evaluate Qwen3-1.7B under FP16 and INT8 rollout to assess whether **MIPU** remains beneficial beyond the FP8 setting. We vary the rollout precision while retaining the other hyperparameters.

Table 6: **Pass@1 accuracy (%) on five mathematical benchmarks under FP16 and INT8 rollout with Qwen3-1.7B. Bold** denotes the best result in each column within each rollout setting.

Rollout	Method	MATH 500	AIME 24	Olympiad	Minerva	AMC23	Avg.
FP16	Baseline	82.29	24.67	58.93	32.72	63.13	52.35
FP16	Ours	**85.11**	**25.11**	**61.31**	**36.77**	**63.28**	**54.32**
INT8	Baseline	82.70	23.78	57.14	32.35	57.31	50.66
INT8	Ours	**83.50**	**25.78**	**59.53**	**34.20**	**62.19**	**53.04**

As reported in Table 6, **MIPU** improves all five benchmark scores under both rollout settings. The average score increases from 52.35 to 54.32 under FP16 and from 50.66 to 53.04 under INT8, corresponding to gains of 1.97 and 2.38 percentage points, respectively. In particular, the improvement under FP16 shows that the benefit of **MIPU** is not restricted to FP8-induced mismatch.

B.4 Evaluation with an Alternative Inference Backend

We additionally evaluate **MIPU** using SGLang in place of the vLLM inference backend used in the main experiments.

Table 7 reports improvements on all five benchmarks, with the average score increasing from 51.13 to 54.24. The largest gain is on OlympiadBench, where **MIPU** improves the score by 5.96 percentage points. These results provide evidence that its benefit extends beyond the inference backend used in the main experiments.

Table 7: **Pass@1 accuracy (%) on five mathematical benchmarks using the SGLang inference backend. Bold** denotes the best result in each column.

Method	MATH 500	AIME 24	Olympiad	Minerva	AMC23	Avg.
Baseline	82.09	22.89	57.14	33.82	59.69	51.13
Ours	**83.50**	**25.11**	**63.10**	**37.13**	**62.34**	**54.24**

Taken together, these supplementary experiments show consistent performance gains across different task domains, model families, rollout precisions, and inference backends. These results provide empirical evidence for the robustness of MIPU across the evaluated settings and support the broader applicability of inference-aware policy optimization.

C Analysis of Step 1: Sampler-Referenced Update

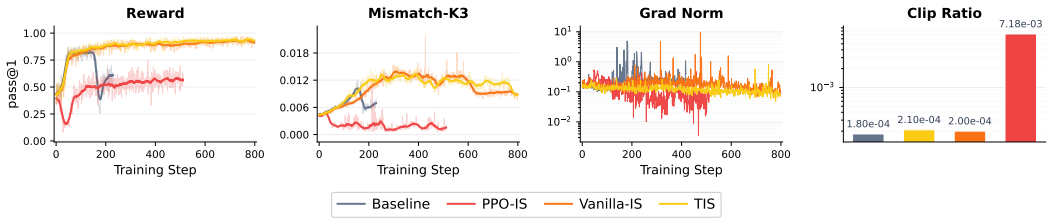


Figure 5: **Step 1 implementation analysis under the Qwen3-4B FP8-quantized rollout.** Comparison of PPO-IS, Vanilla-IS, and TIS in terms of performance, gradient norm, inference-training K3-KL, and clip ratio.

Step 1 implements the sampler-referenced update derived in Equation 6. For readability, we omit token or trajectory arguments in this section, and all ratios are evaluated on sampled rollouts. A direct implementation is PPO-IS, which clips the total trainer-to-sampler ratio:

$$\textbf{PPO-IS:} \qquad \omega_{\text{PPO-IS}} = \text{clip}\left(\frac{\pi_\theta}{\mu_k}, 1-\epsilon, 1+\epsilon\right).$$

Although this ratio directly references the sampler, it mixes the pre-update mismatch between π_k and μ_k with the current update from π_k to π_θ. As a result, the ratio can already deviate substantially from 1 even before the current update is applied. This makes the PPO clipping overly active and therefore less informative, because the trust-region constraint is no longer centered on the current training-side update itself. This phenomenon is also consistent with the analysis in Yao et al. (2025). Empirically, Figure 5 shows that PPO-IS has a much larger clip ratio than the other variants and, despite reducing the measured inference-training KL, fails to produce a sustained upward learning trajectory.

We therefore use the factorization

$$\frac{\pi_\theta}{\mu_k} = \underbrace{\frac{\pi_k}{\mu_k}}_{\text{pre-update mismatch correction}} \cdot \underbrace{\frac{\pi_\theta}{\pi_k}}_{\text{current update ratio}},$$

which separates the pre-update mismatch correction from the current training-side update. This leads to two decomposed variants:

$$\textbf{Vanilla-IS:} \qquad \omega_{\text{Vanilla-IS}} = \frac{\pi_k}{\mu_k} \cdot \text{clip}\left(\frac{\pi_\theta}{\pi_k}, 1-\epsilon, 1+\epsilon\right),$$

$$\textbf{TIS:} \qquad \omega_{\text{TIS}} = \min\left(\frac{\pi_k}{\mu_k}, C\right) \cdot \text{clip}\left(\frac{\pi_\theta}{\pi_k}, 1-\epsilon, 1+\epsilon\right).$$

Vanilla-IS keeps the trust-region constraint on the current update ratio π_θ/π_k, which is more suitable than directly clipping π_θ/μ_k. However, its mismatch-correction weight π_k/μ_k is unbounded. When a rollout token is sampled with low probability under μ_k, this weight can become large and significantly amplify gradient variance, as also discussed in Yao et al. (2025). This explains why Vanilla-IS recovers a clearer upward learning trend than PPO-IS, but still exhibits larger gradient-norm spikes in Figure 5(a).

TIS further truncates the mismatch-correction weight while preserving the decomposed form. As shown in Figure 5, this yields the best stability-performance trade-off among the Step 1 variants: compared with PPO-IS, it avoids overly aggressive clipping, and compared with Vanilla-IS, it controls variance amplification from large mismatch weights. We therefore use TIS to construct the Step 1 candidate update in the full method.

In summary, directly clipping π_θ/μ_k makes the trust-region constraint depend on the pre-existing mismatch, which can render the update overly conservative. The decomposed form separates mismatch correction from the current update, and the truncated version used by TIS further controls variance, making it the most effective Step 1 implementation in our setting.

D Analysis of the Tolerance Parameter c

We further study how the acceptance tolerance affects Step 2. In the main method, the synchronized candidate is accepted if $\widehat{T}_{\text{post}} \geq -c$, where $c \geq 0$ is a tolerance parameter. For sensitivity analysis, we keep the same boundary form $\widehat{T}_{\text{post}} \geq -c$ and also test $c < 0$, which corresponds to a stricter-than-zero acceptance rule rather than a tolerance. Thus, $c > 0$ allows mild negative proxy values, $c = 0$ gives strict non-regression under the proxy, and $c < 0$ requires a positive margin.

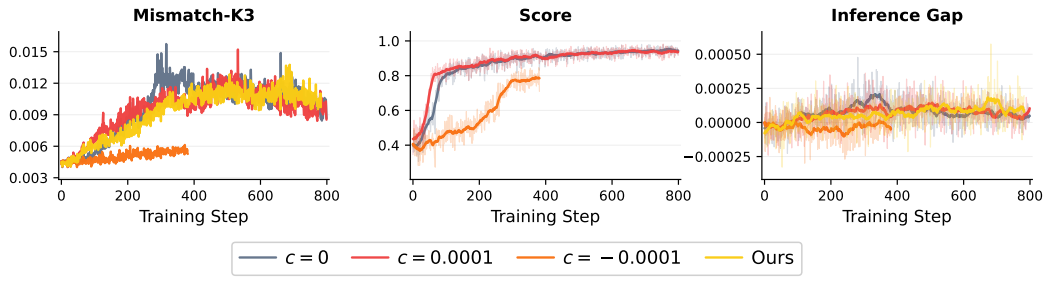


Figure 6: **Sensitivity to the acceptance tolerance c under the Qwen3-4B FP8-quantized rollout** in terms of inference-training K3-KL, training score, and $\widehat{T}_{\text{post}}$.

Figure 6 shows that stricter acceptance is not always beneficial. When the acceptance rule is too strict, Step 2 rejects many early candidates. This is especially harmful because $\widehat{T}_{\text{post}}$ is initially low under FP8-quantized rollout and gradually increases during training, as observed in Figure 4(a). As a result, overly strict acceptance slows down learning and leads to a lower performance plateau. In particular, the stricter setting enters persistent rollback after around 280 steps, suggesting that conservative rejection alone cannot provide sufficient policy improvement.

These results show that rollback rate itself is not the objective. Although rejecting more updates may appear safer, excessive rollback can prevent the inference policy from moving to a better region. The dynamic tolerance used in **MIPU** provides a calibrated acceptance rule: it starts with a larger tolerance to allow useful early candidates, and gradually tightens the criterion as the post-update gap becomes more stable. This balances early learning efficiency with later-stage protection against unreliable updates caused by mismatch.

E Licenses

The licenses of used assets in this paper are listed as follows:

- MATH500: MIT License

- AIME24: Apache-2.0 License
- AMC23: No License
- Minerva Math: No License
- OlympiadBench: Apache-2.0 License
- DAPO-Math-17k: Apache-2.0 License
- DeepMath-103k: MIT License
- Qwen3 Models: Apache-2.0 License
- ROLL: Apache-2.0 License
- vLLM: Apache-2.0 License
- Megatron: Apache-2.0 License

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: See Abstract and Introduction for our contribution.

   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: The limitation is in Section 6.

   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: Our assumptions and proof are in Section 4.

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: Our implementation details are in Appendix A.

   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 use open-sourced datasets for training and evaluation.

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: The hyperparameters are in Appendix A.1.

   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: To ensure the statistical significance of our results on small-scale datasets, we report the avg@16 score for pass@1.

   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: See Appendix A.1 for compute resources 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: Our research strictly adheres to the NeurIPS Code of Ethics.

   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: [N/A]

    Justification: This research focuses on foundational algorithmic improvements for reinforcement learning and mathematical reasoning. As a theoretical contribution using public benchmarks, it does not have a direct or immediate societal impact, making a dedicated discussion not applicable.

    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: [N/A]

    Justification: This work focuses on fundamental algorithmic advancements in reinforcement learning. Given the nature of the methodology and the use of public mathematical benchmarks for evaluation, we do not foresee any significant risk of misuse or negative social impact as defined by the current guidelines.

    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: [Yes]

    Justification: The licenses of existing assets are in Appendix E.

    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: [Yes]

    Justification: See main paper for details.

    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: This research does not involve crowdsourcing or experiments with human subjects.

    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: This work does not involve human subjects, personal data, or any experiments requiring IRB approval.

    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: LLMs were used only to improve the clarity and grammar of the text. The core components of this work, such as the original ideas, theoretical proofs, and empirical results, are the sole contributions of the authors. This research does not rely on LLMs as a part of its technical approach.

    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.