# RLVR WITHOUT INEFFECTIVE SAMPLES: GROUP PRIORITIZED OFF-POLICY OPTIMIZATION FOR LLM REASONING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Reinforcement learning with verifiable rewards (RLVR) has emerged as a powerful paradigm for enhancing the reasoning capabilities of large language models (LLMs). However, its effectiveness is substantially hindered by the prevalence of ineffective training data: many sampled prompts yield response groups that are either entirely correct or entirely incorrect, resulting in zero-variance rewards and limited learning signals. Recent state-of-the-art methods address this issue through extensive LLM rollouts to filter ineffective samples, but at the cost of considerable computational overhead. Alternative approaches, including predictive sampling and trajectory replay, aim to improve data efficiency but often remain insufficient and may introduce additional issues such as systematic bias or suboptimal constraints. To address these limitations, we propose Group Prioritized Off-Policy Optimization (POPO), a simple yet effective framework that fully exploits effective training batches without additional rollout overhead. POPO comprises two key components: prioritized group replay and decoupled off-policy optimization. The former replaces ineffective on-policy groups with effective off-policy groups via a recency-based replay mechanism that jointly considers sample quality and the degree of off-policiness. To further mitigate the off-policy gap, POPO employs decoupled importance sampling to correct off-policy bias while maintaining stable policy updates under consistent trust-region constraints. Empirical evaluations across diverse reasoning tasks, including mathematics, planning, and visual geometry, demonstrate that POPO substantially accelerates RL finetuning and achieves strong reasoning performance with significantly fewer rollouts.

## 1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) has emerged as a core technique for enhancing the reasoning capabilities of large language models (LLMs) [26; 20; 14; 52]. By encouraging LLMs to generate extended chain-of-thought (CoT) reasoning [54] and optimizing policies using verifiable reward signals [20], RLVR has enabled substantial progress in large reasoning models (LRMs) across a broad range of knowledge-intensive tasks, including scientific question answering [15], symbolic mathematics [33], logical deduction [55], and program synthesis [32].

However, the effectiveness of RLVR is significantly affected by the prevalence of ineffective training samples [60]. In practice, a substantial portion of sampled prompts produce response groups that are either entirely correct or entirely incorrect [67; 40]. This leads to zero-variance rewards and limits the availability of informative training signals [2; 4]. To alleviate this issue, recent state-of-the-art (SOTA) methods perform extensive LLM rollouts over multiple candidate batches to identify and retain effective samples [60; 7; 28]. Although effective at improving sample quality, this strategy incurs substantial computational overhead that often exceeds the cost of RL finetuning itself.

Considering this efficacy-efficiency dilemma, several alternative approaches have been explored. Predictive sampling methods estimate prompt success probabilities prior to rollout and prioritize prompts with intermediate success rates [40; 47; 57]. However, accurate prediction remains challenging, especially for rapidly evolving models or large-scale training datasets. Consequently, these approaches often achieve only limited improvements in effective sample ratios. Another line of research leverages



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

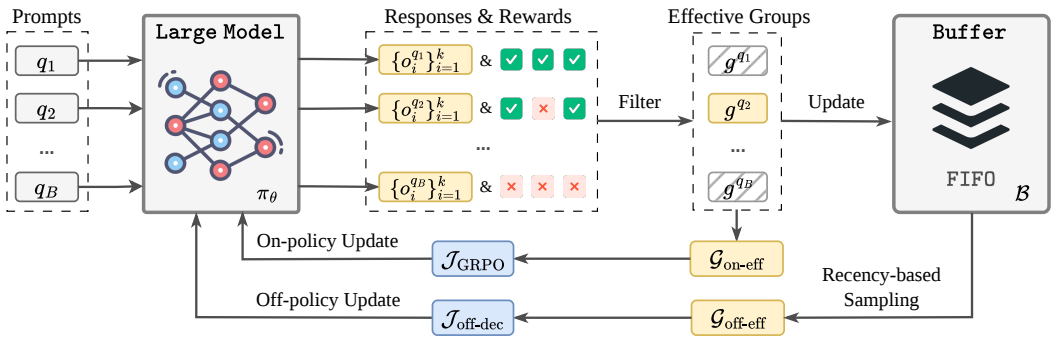



Figure 1: Framework overview of POPO. During RLVR training, ineffective online groups are replaced with high-quality off-policy groups via recency-based replay, forming batches composed entirely of effective samples without additional overhead. For replayed off-policy groups, decoupled importance sampling is introduced to correct off-policy bias and maintain a consistent trust-region constraint across on- and off-policy updates, enabling stable and efficient policy optimization.

trajectory replay to strengthen training signals from difficult prompts, typically inserting previously successful trajectories into online response groups under specific conditions [30; 65; 64]. However, trajectory replay does not fundamentally resolve prompt inefficiency, leaving a substantial proportion of training samples ineffective. Moreover, the resulting policy optimization objectives may suffer from issues including intractable bias, suboptimal constraints, or invalid advantage estimates.

To overcome these limitations, we propose Group Prioritized Off-Policy Optimization (POPO), a simple yet effective approach that stably optimizes entire batches of effective data without incurring additional rollout overhead. POPO consists of two components: prioritized group replay and decoupled off-policy optimization. The prioritized group replay mechanism jointly accounts for sample quality and off-policiness. Specifically, it replaces ineffective on-policy groups, each comprising a prompt and its associated responses, with effective off-policy groups generated by recent policies. This design not only eliminates ineffective training data but also avoids issues inherent in trajectory replay, such as intractable bias and invalid advantage estimates. Furthermore, we present a decoupled importance-sampling strategy for stable off-policy optimization over replayed groups. While prior off-policy objectives either introduce bias by treating replayed samples as on-policy or impose suboptimal constraints toward the behavior policy, POPO corrects off-policy bias and ensures stable policy updates through consistent trust-region constraints across both on-policy and off-policy samples.

Empirically, we evaluate POPO on a diverse set of downstream reasoning tasks, including competition-level mathematics, numerical planning, and visual geometry. The results show that POPO substantially accelerates RL finetuning, achieving performance comparable to the SOTA method while requiring substantially fewer rollouts. Moreover, POPO consistently improves reasoning performance across varying response group sizes and base RLVR algorithms, and remains complementary to existing sampling strategies, highlighting its broad applicability and practical utility for RLVR training.

## 2 Preliminary

**RL with Verifiable Rewards (RLVR).** Let $q \sim \mathcal{D}$ denote a prompt sampled from a dataset and $o \sim \pi_\theta(\cdot|q)$ a response generated by policy $\pi_\theta$. RLVR aims to maximize the expected reward:

$$\max_{\theta \in \Theta} \mathbb{E}_{q \sim \mathcal{D}, o \sim \pi_\theta(\cdot|q)} \left[ r(q, o) \right], \tag{1}$$

where the reward function $r(q, o)$ typically verifies response correctness, assigning a value of 1 to correct outputs and 0 otherwise. Such rewards are simple, reliable, and less susceptible to reward hacking in domains such as mathematical reasoning and code generation.

**Group Relative Policy Optimization (GRPO).** To optimize the above objective, GRPO [46] is a widely adopted policy gradient method that avoids value function modeling. At each training step, a batch of $B$ prompts is sampled, and for each prompt $q$, the policy generates $k$ responses $\{o_i\}_{i=1}^{k}$,

forming a group $g^q = \left(q, \{o_i\}_{i=1}^k\right)$. GRPO then maximizes the following objective:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{q,o_i\sim\pi_{\theta_{\text{old}}}(\cdot|q)}\left[\frac{1}{k}\sum_{i=1}^{k}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\min\left(\rho_{i,t}(\theta,\theta_{\text{old}})\hat{A}_i, \text{clip}(\rho_{i,t}(\theta,\theta_{\text{old}}), 1\pm\epsilon)\hat{A}_i\right)\right], \quad (2)$$

where $\rho_{i,t}(\theta,\theta_{\text{old}}) = \frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q,o_{i,<t})}$. The clipping term enforces proximity between $\pi_\theta$ and $\pi_{\theta_{\text{old}}}$ for stable updates. The advantages are computed via group-wise normalization:

$$\hat{A}_i = \frac{r(q,o_i) - \text{mean}\left(\{r(q,o_j)\}_{j=1}^k\right)}{\text{std}\left(\{r(q,o_j)\}_{j=1}^k\right)}. \quad (3)$$

## 3 Ineffective Samples and Prior Solutions

This section characterizes ineffective RLVR samples and reviews prior solutions and their limitations.

**Ineffective Samples.** In RLVR, not all samples contribute equally to learning. When a prompt consistently yields either all-correct or all-incorrect responses, the resulting rewards provide little informative signal for model optimization [4; 60; 2]. For algorithms such as GRPO, this leads to vanishing normalized advantages and consequently stalled optimization.

**Definition 3.1** (Ineffective Sample)**.** Given a prompt $q$ and its response group $g^q = \left(q, \{o_i\}_{i=1}^k\right)$, we define the sample as ineffective if the reward set exhibits zero variance: $\text{std}\left(\{r(q,o_i)\}_{i=1}^k\right) = 0$.

Both prior studies [67; 60; 40] and our empirical findings (Fig. 3) indicate that, under standard RLVR training, a considerable portion of samples tend to be ineffective and contribute little to learning. This observation has motivated a growing body of research aimed at mitigating such inefficiencies.

**Active Sampling.** A common approach is to prioritize informative prompts via active sampling [67; 4; 60; 2]. A representative SOTA method is DAPO [60], which performs rollouts over an expanded candidate prompt batch $\hat{\mathcal{Q}}_t$ and filters out prompts whose responses yield identical rewards:

$$\mathcal{Q}_t = \left\{q \in \hat{\mathcal{Q}}_t \;\middle|\; \text{std}\left(\{r(q,o_i)\}_{i=1}^k\right) > 0\right\}. \quad (4)$$

While effective at improving sample quality, this strategy incurs substantial computational overhead due to repeated LLM rollouts over a larger candidate pool. The additional rollout cost can dominate overall training, particularly for reasoning tasks involving long CoT outputs.

To reduce rollout overhead, predictive sampling methods estimate prompt success probabilities and prioritize prompts with intermediate success rates before executing rollouts [40; 47; 57]. However, accurate prediction remains challenging, particularly when (i) prompt success rates shift rapidly as the policy evolves, or (ii) historical information is sparse in low-epoch or large-scale dataset settings. As a result, these methods often yield only limited improvements in effective sample ratios. Moreover, prioritizing intermediate-difficulty prompts may systematically exclude highly challenging samples, which are important for improving the upper bound of reasoning capability [58].

**Trajectory Replay.** Another line of work improves data efficiency through replay mechanisms. These methods typically store historical successful responses in a prompt-specific buffer and insert them into online response groups under specific conditions [30; 65; 64; 68], such as when all online rollouts fail [30; 68]. Although effective at amplifying sparse success signals in difficult tasks, trajectory replay does not fully resolve the problem of ineffective samples. In particular, samples remain ineffective when (i) all online responses are correct, or (ii) no successful trajectory has ever been obtained for extremely hard prompts or in large-scale dataset settings.

Moreover, trajectory replay introduces additional challenges for policy optimization. Many methods directly apply on-policy objectives to replayed samples [30; 65], leading to biased optimization when the behavior policy differs from the current policy. Recent work [64; 68] has attempted to correct this bias via importance sampling with respect to the behavior policy that generated the replayed sample. However, this remains imperfect as selectively replayed trajectories are drawn from a filtered distribution rather than the true behavior policy, introducing potentially intractable bias. Furthermore, constraining updates toward the behavior policy can lead to inconsistent trust-region constraints and may overly restrict policy improvement. Finally, mixing off-policy trajectories within groups invalidates normalized advantage estimates, as they no longer correspond to any well-defined policy.

# 4 GROUP PRIORITIZED OFF-POLICY OPTIMIZATION

To address the limitations of prior methods, we propose Group Prioritized Off-Policy Optimization (POPO), a simple yet effective framework that constructs fully effective training batches via replay, enabling stable off-policy optimization without additional rollout cost. POPO consists of two components: prioritized group replay (Section 4.1) and decoupled off-policy optimization (Section 4.2).

## 4.1 PRIORITIZED GROUP REPLAY

Existing trajectory replay methods face the aforementioned challenges primarily because they retain all sampled prompts and operate at the response level. Since extremely easy or hard prompts provide limited learning signals [4; 40], manipulating responses alone is often insufficient and may further introduce intra-group policy inconsistency. We therefore propose prioritized group replay, a group-level mechanism that replaces ineffective prompt-response groups with historical groups according to both quality and off-policiness criteria.

**Quality of Groups.** Partially correct groups are generally of higher quality as they provide stronger learning signals [2; 4]. At each training step, we sample a batch of $B$ prompts $\mathcal{Q}_{\text{on}} \subset \mathcal{D}$ and perform online rollouts, yielding online groups $\mathcal{G}_{\text{on}} = \{g^q \mid q \in \mathcal{Q}_{\text{on}}\}$. From these online groups, we retain only effective groups with non-zero reward variance:

$$\mathcal{G}_{\text{on-eff}} = \left\{ g^q \in \mathcal{G}_{\text{on}} \;\middle|\; \text{std}\big(\{r(q, o_i)\}_{i=1}^{k}\big) > 0 \right\}. \tag{5}$$

Similar to DAPO, this filtering step reduces the training batch size. Instead of continually rolling out additional prompts, we replenish the batch using a replay buffer $\mathcal{B}$ that stores previously observed effective groups. As $|\mathcal{G}_{\text{on-eff}}| \leq B$, we sample $B - |\mathcal{G}_{\text{on-eff}}|$ effective groups from $\mathcal{B}$ according to a specified sampling strategy to form the off-policy effective group set $\mathcal{G}_{\text{off-eff}}$:

$$\mathcal{G}_{\text{off-eff}} = \text{Sample}(\mathcal{B},\, B - |\mathcal{G}_{\text{on-eff}}|). \tag{6}$$

The training batch is constructed as $\mathcal{G}_{\text{train}} = \mathcal{G}_{\text{on-eff}} \cup \mathcal{G}_{\text{off-eff}}$. This design ensures that every training batch consists entirely of effective groups, without requiring rollout-intensive rejection schemes such as DAPO. Compared with predictive sampling methods, it also preserves exploration by sampling prompts uniformly rather than concentrating on specific types, such as medium-difficulty prompts.

**Off-policiness of Groups.** While the quality criterion determines which groups enter the buffer, the off-policiness criterion determines which groups are replayed. To mitigate distribution shift, replayed groups should be generated by policies close to the current policy. A direct approach would be to compute KL divergence for all candidate groups and then perform filtering, which is computationally expensive for LLMs. Fortunately, under proximal policy updates, policies trained at more recent steps are expected to remain closer to the current policy. Lemma 4.1 formalizes this intuition by showing that the total variation bound between policies grows linearly with the update-step difference.

**Lemma 4.1.** *Assume the per-step parameter update is bounded, $\|\theta_{i+1} - \theta_i\|_2 \leq \Delta$ for all steps $i$, and that $\theta \mapsto \log \pi_\theta(a|s)$ is $L$-Lipschitz over the parameter region visited during training. Then, for any step difference $n \geq 1$, $\text{TV}\big(\pi_i(\cdot|s), \pi_{i+n}(\cdot|s)\big) \leq \frac{n}{2}\big(e^{L\Delta} - 1\big) = \frac{nL\Delta}{2} + O\big(n(L\Delta)^2\big)$.*

Motivated by this observation, we adopt a recency-based replay strategy. Specifically, we maintain the replay buffer $\mathcal{B}$ as a first-in-first-out (FIFO) queue and replay the most recently stored groups. Let $\mathcal{B} = (g_1, g_2, \ldots, g_{|\mathcal{B}|})$ denote the buffer ordered by insertion time. To control off-policiness, we deterministically select groups from the tail of the buffer:

$$\mathcal{G}_{\text{off-eff}} = \left\{ g_{|\mathcal{B}|-i+1} \in \mathcal{B} \;\middle|\; i = 1, \ldots, B - |\mathcal{G}_{\text{on-eff}}| \right\}. \tag{7}$$

This strategy provides a lightweight approximation to explicit KL-based filtering without additional LLM likelihood computation. Moreover, since at most $B$ groups are replayed per step, the buffer capacity can be set to $B$, incurring minimal memory overhead. After constructing the training batch, the replay buffer is updated by including all effective online groups at the current step: $\mathcal{B} \leftarrow \mathcal{B} \cup \mathcal{G}_{\text{on-eff}}$.

## 4.2 DECOUPLED OFF-POLICY OPTIMIZATION

Although prioritized replay limits the off-policy gap, a residual discrepancy remains. We introduce decoupled off-policy optimization to correct this bias while preserving stable trust-region updates.

Let $\pi_\beta$ denote the behavior policy that generated a replayed sample, i.e., the LLM policy at the time the sample was collected. A straightforward approach, adopted by several existing replay-based methods [30; 65], is to treat off-policy data as if they were on-policy. This yields an objective that shares the same form as $\mathcal{J}_{\text{GRPO}}$, but with the expectation taken under $\pi_\beta$ rather than $\pi_{\theta_{\text{old}}}$:

$$\mathcal{J}_{\text{off-}\pi_{\text{old}}}(\theta) = \mathbb{E}_{q,o_i\sim\pi_\beta(\cdot|q)}\left[\frac{1}{k}\sum_{i=1}^{k}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\min\left(\rho_{i,t}(\theta,\theta_{\text{old}})\hat{A}_i, \text{clip}(\rho_{i,t}(\theta,\theta_{\text{old}}),1\pm\epsilon)\hat{A}_i\right)\right]. \quad (8)$$

However, the objective becomes biased whenever $\pi_\beta \neq \pi_{\theta_{\text{old}}}$. In typical replay scenarios, $\pi_\beta$ may deviate substantially from $\pi_{\theta_{\text{old}}}$, introducing systematic bias that degrades policy updates.

To correct this off-policy mismatch, a second line of work [64; 49; 25] replaces the denominator in the importance ratio with the behavior policy $\pi_\beta$, yielding the following objective:

$$\mathcal{J}_{\text{off-}\pi_\beta}(\theta) = \mathbb{E}_{q,o_i\sim\pi_\beta(\cdot|q)}\left[\frac{1}{k}\sum_{i=1}^{k}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\min\left(\rho_{i,t}(\theta,\beta)\hat{A}_i, \text{clip}(\rho_{i,t}(\theta,\beta),1\pm\epsilon)\hat{A}_i\right)\right], \quad (9)$$

where we generalize the notation $\rho$ and define, for any $\pi_{\theta_1}$ and $\pi_{\theta_2}$, $\rho_{i,t}(\theta_1,\theta_2) = \frac{\pi_{\theta_1}(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_2}(o_{i,t}|q,o_{i,<t})}$. However, by constraining the optimized policy $\pi_\theta$ toward the behavior policy $\pi_\beta$ through its clipping operation, $\mathcal{J}_{\text{off-}\pi_\beta}$ introduces three issues. (i) Suboptimal constraint: $\pi_\beta$ is usually inferior to the current policy, so constraining toward it may impair performance. (ii) Unstable updates: Unlike on-policy updates that consistently constrain $\pi_\theta$ toward $\pi_{\theta_{\text{old}}}$, this objective enforces proximity to different behavior policies across replayed samples, disrupting the standard trust-region dynamics and undermining its stability advantage. (iii) Less informative signals: Since $\pi_\theta$ generally deviates much more substantially from $\pi_\beta$ than from $\pi_{\theta_{\text{old}}}$, the objective causes clipping to occur more frequently, thereby reducing the informativeness of the gradient signal.

These issues stem from using $\pi_\beta$ as the proximal policy that constrains the optimized policy. To address them, we aim to retain $\pi_{\theta_{\text{old}}}$ as the proximal policy in the clipping ratio, even when optimizing over off-policy samples drawn from different $\pi_\beta$. To this end, we adopt a decoupled importance sampling scheme that factorizes the correction term $\rho_{i,t}(\theta,\beta)$ into the product of two importance weights, $\rho_{i,t}(\theta,\theta_{\text{old}})$ and $\rho_{i,t}(\theta_{\text{old}},\beta)$, and optimize the following decoupled off-policy objective:

$$\mathcal{J}_{\text{off-dec}}(\theta) = \mathop{\mathbb{E}}_{q,o_i\sim\pi_\beta}\left[\frac{1}{k}\sum_{i=1}^{k}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\rho_{i,t}(\theta_{\text{old}},\beta)\min\left(\rho_{i,t}(\theta,\theta_{\text{old}})\hat{A}_i, \text{clip}(\rho_{i,t}(\theta,\theta_{\text{old}}),1\pm\epsilon)\hat{A}_i\right)\right]. \quad (10)$$

The objective $\mathcal{J}_{\text{off-dec}}$ preserves the same off-policy correction as $\mathcal{J}_{\text{off-}\pi_\beta}$ while modifying only the trust-region constraint. The following lemma formalizes this connection.

**Lemma 4.2.** *$\mathcal{J}_{\text{off-dec}}$ and $\mathcal{J}_{\text{off-}\pi_\beta}$ differ only in their trust-region constraints: the former clips $\rho_{i,t}(\theta,\theta_{old})$, whereas the latter clips $\rho_{i,t}(\theta,\beta)$. For any token at which neither ratio is clipped, the two objectives yield identical values and gradients.*

This formulation decouples the behavior policy that generates the data from the proximal policy that constrains the update, enabling POPO to maintain a consistent, high-quality trust-region constraint across both on-policy and off-policy data, and thereby enhance training stability. While the decoupled formulation originates from classical RL for batch-size-invariant optimization [16], and has recently been adopted in LLM training to address precision mismatch between rollout and training engines [69; 12], this work repurposes it for replay-based off-policy optimization with stable trust-region updates.

### 4.3 Overall Algorithm

This section summarizes our final algorithm, POPO. As described in Section 4.1, the final training batch $\mathcal{G}_{\text{train}}$ consists of the online effective group set $\mathcal{G}_{\text{on-eff}}$ and the off-policy effective group set $\mathcal{G}_{\text{off-eff}}$. For samples in $\mathcal{G}_{\text{on-eff}}$, we perform standard GRPO training by Eq. (2), while for samples in $\mathcal{G}_{\text{off-eff}}$, we apply decoupled off-policy optimization using Eq. (10). In practice, this only requires multiplying the standard policy loss of each replayed sample by $\rho_{i,t}(\theta_{\text{old}},\beta)$ before loss aggregation. Therefore, POPO is easy to implement and introduces neither additional computational overhead nor modeling complexity compared to standard RLVR methods. The complete algorithm is presented in Algorithm 1, and an overview of the framework is illustrated in Fig. 1.

**Algorithm 1:** Group Prioritized Off-Policy Optimization

**Input:** Prompt dataset $\mathcal{D}$; Batch size $B$; Group size $k$; LLM $\pi_\theta$; Total training steps $T$.
Initilize a FIFO replay buffer $\mathcal{B} = \varnothing$ with capacity $B$;
**for** $t = 1$ **to** $T$ **do**
  Sample a batch of prompts $\mathcal{Q}_{\text{on}}$ from $\mathcal{D}$;
  Generate $k$ responses per $q \in \mathcal{Q}_{\text{on}}$ using $\pi_\theta$ to obtain the online group set $\mathcal{G}_{\text{on}} = \{g^q | q \in \mathcal{Q}_{\text{on}}\}$;
  Remove group $g^q$ from $\mathcal{G}_{\text{on}}$ if std $\left(\{r(q, o_i)\}_{i=1}^{k}\right) = 0$ to obtain the online effective set $\mathcal{G}_{\text{on-eff}}$;
  Construct the off-policy effective set $\mathcal{G}_{\text{off-eff}} = \left\{ g_{|\mathcal{B}|-i+1} \in \mathcal{B} \;\middle|\; i = 1, \ldots, B - |\mathcal{G}_{\text{on-eff}}| \right\}$;
  Update the LLM $\pi_\theta$ using groups from $\mathcal{G}_{\text{on-eff}}$ and $\mathcal{G}_{\text{off-eff}}$ by Eq. (2) and (10), respectively;
  Add groups $g^q \in \mathcal{G}_{\text{on-eff}}$ to the FIFO replay buffer $\mathcal{B}$;
**end for**

# 5 Experiments

In this section, we conduct a series of experiments to examine the effectiveness of POPO. Appendices D, E, and G provide experimental details, extended results, and data examples, respectively.

## 5.1 Experimental Setup

We evaluate POPO across three challenging reasoning domains, using diverse model sizes and architectures. Performance is measured by average pass@1 over multiple rollouts per problem.

**Mathematics.** Following prior work [33; 40], we train DeepSeek-R1-Distill-Qwen-1.5B (DSR-1.5B) and 7B (DSR-7B) [14] on the DeepScaleR dataset [33], which comprises 40k competition-level math problems. Evaluation is conducted on a suite of standard mathematical benchmarks: AIME25, AMC23, MATH500 [26], Minerva Math (Minerva.) [23], and OlympiadBench (Olymp.) [15]. To examine out-of-distribution (OOD) generalization, we further use general reasoning benchmarks including MMLU-Pro [53], ARC-c [6], and GPQA-diamond (GPQA) [43].

**Numerical Planning.** We adopt the Countdown Numbers Game to evaluate arithmetic planning. Following prior work [4; 40], we train Qwen2.5-3B [59] on a 20k subset of the Countdown-34 (CD-34) dataset [39], and evaluate on its held-out split and a harder variant, Countdown-4 (CD-4).

**Visual Geometry.** Following prior work [40], we train the vision-language model Qwen2.5-VL-3B-Instruct [3] on the Geometry3k training set [31; 17], and evaluate on its test split. The dataset contains 3k visual-geometric problems that require both image understanding and symbolic reasoning.

**Baselines.** We compare against the following representative baselines. (i) GRPO: the GRPO algorithm with the clip higher technique [60]. (ii) DAPO: a resource-intensive approach that oversamples and filters prompts using rollout feedback to form fully effective online batches [60]. (iii) MoPPS: a predictive sampling method that predicts effective prompts using Bayesian inference [40]. (iv) ARPO: a trajectory replay method that reuses historical correct responses when online groups contain only incorrect ones [30]. On DSR-1.5B, we additionally include the off-policy baselines ReMix [25], REPO [24], RR [49], and the predictive sampling method GRESO [71]. Our primary goal is reducing computational overhead relative to DAPO, rather than surpassing its accuracy.

## 5.2 RLVR Performance and Efficiency

**Training Progress.** Fig. 2 presents training curves across tasks and models, plotting test accuracy against training steps. POPO consistently yields faster improvement and higher final accuracy than GRPO, ARPO, and MoPPS, while performing on par with the computationally expensive baseline DAPO. POPO also reaches GRPO's final performance with only 40%–70% of training steps. To understand how training data quality contributes to this gap, Fig. 3 reports the effective sample ratio (i.e., the proportion of effective data in training batches) of each method throughout training. While POPO and DAPO optimize full batches of effective samples, GRPO typically maintains effective ratios below 50% across tasks. Consistent with our analysis in Section 3, MoPPS and ARPO improve the effective ratio mainly on small datasets: on Geometry (3k), their effective ratios are markedly higher; this advantage diminishes on larger datasets such as DeepScaleR (40k) and Countdown

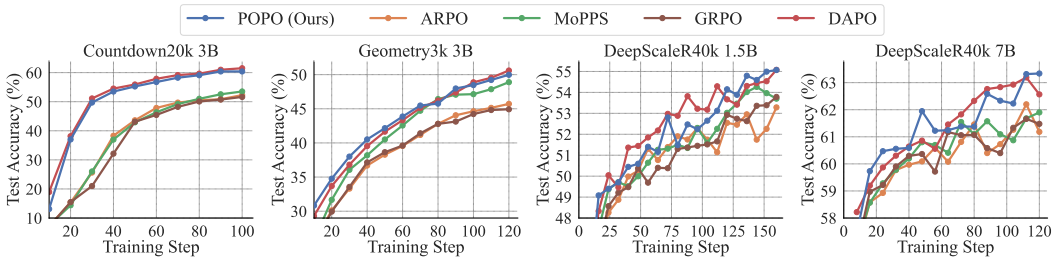



Figure 2: Training curves of different methods across three reasoning tasks with varying model sizes. Notably, DAPO is included as a high-resource baseline as it requires substantially more LLM rollouts per training step than competing methods.

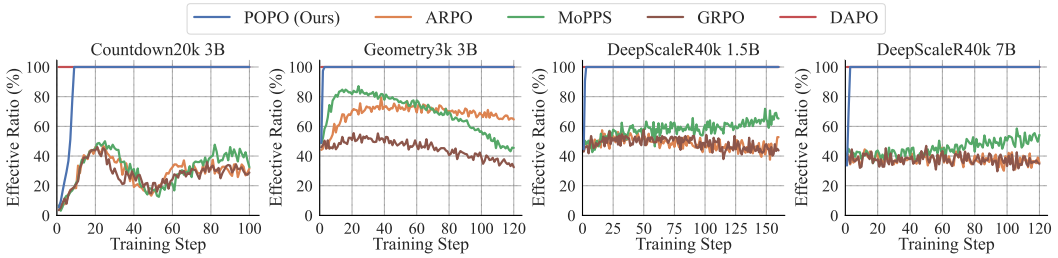



Figure 3: Proportion of effective samples within training batches across different RLVR methods.

(20k), where limited per-prompt history weakens their predictive or replay mechanisms, causing performance similar to GRPO. Notably, although ARPO achieves higher effective ratios on Geometry, its performance gain is limited, possibly due to inaccurate advantage estimation.

**Generalization Performance.** For models trained on DeepScaleR, Table 1 reports results on both standard in-distribution mathematical benchmarks and OOD general reasoning benchmarks. Table 2 presents evaluations on Countdown and Geometry. Across all benchmarks, POPO consistently matches the performance of DAPO while outperforming the other baselines.

**Rollout and Runtime Efficiency.** We further compare methods in terms of rollout usage and runtime. Tables 1 and 2 list the total number of generated rollouts and runtime for each task, and Fig. 4 plots model performance against rollout count during training (see Fig. 7 for full curves). POPO achieves competitive performance with DAPO using only 30% of its rollout budget and roughly half the runtime. The slight runtime increase of POPO over GRPO stems from longer response lengths associated with higher performance, as shown in Fig. 9.

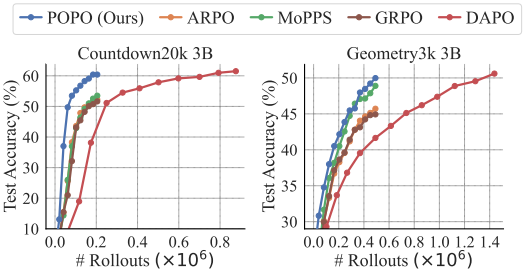



Figure 4: Training curves of different methods as a function of the number of LLM rollouts.

### 5.3 Effects of Components in POPO

**Replay Mechanisms.** We examine alternative replay strategies. POPO-KL selects replay groups by ranking KL divergence from a buffer 5× larger than the batch. As shown in Fig. 5 (left), POPO achieves similar performance but with significantly better efficiency (3.2h vs 4.8h), suggesting recency as an effective low-cost proxy for distance-based replay. GRPO-filter removes ineffective groups and updates only on on-policy effective groups. This filtering reduces the training batch size, which increases gradient magnitude due to batch averaging but also amplifies noise. As shown in Fig. 5 (left), this variant is less effective than POPO and only marginally outperforms GRPO, highlighting the importance of replay-based batch completion.

POPO's prioritized replay taking into account both sample quality and off-policiness. To disentangle their effects, we test the following variants: POPO-ineff substitutes ineffective online groups by

Table 1: Evaluation on mathematics benchmarks. '+' denotes finetuning with the method.

| Method | In-Distribution | | | | | | Out-of-Distribution | | | | Runtime |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | AIME25 | AMC23 | Minerva. | MATH500 | Olymp. | Avg. | MMLU-P | ARC-c | GPQA | Avg. | |
| DSR-1.5B | 19.2 | 57.0 | 23.5 | 78.0 | 38.2 | 43.2 | 21.1 | 42.2 | 23.0 | 28.8 | - |
| +GRPO | 26.9 | 76.1 | 30.8 | 86.2 | 49.0 | 53.8 | 22.3 | 45.4 | 27.5 | 31.7 | 16h |
| +ARPO | 27.7 | 76.7 | 28.8 | 86.2 | 48.1 | 53.3 | 22.3 | 45.3 | 26.3 | 31.3 | 16h |
| +GRESO | 25.6 | 77.1 | 30.7 | 86.2 | 50.6 | 54.1 | 24.7 | 46.7 | 26.4 | 32.6 | 27h |
| +ReMix | 27.4 | 79.4 | 30.0 | 85.4 | 51.0 | 54.5 | 22.5 | 45.3 | 27.7 | 31.8 | 16h |
| +RR | 26.5 | 76.7 | 31.0 | 85.0 | 49.6 | 53.8 | 21.2 | 45.4 | 27.1 | 31.2 | 14h |
| +REPO | 28.3 | 77.5 | 30.3 | 86.4 | 50.9 | 54.6 | 21.9 | 44.9 | 27.4 | 31.4 | 19h |
| +MoPPS | 28.4 | 77.7 | 29.9 | 85.2 | 50.1 | 54.3 | 21.4 | 44.6 | 27.5 | 31.2 | 17h |
| +DAPO | 27.5 | 79.2 | 30.4 | 87.2 | 51.0 | **55.1** | 24.5 | 46.7 | 26.8 | 32.7 | 30h |
| +POPO | 29.9 | 77.9 | 29.9 | 86.2 | 51.5 | **55.1** | 22.6 | 45.9 | 30.3 | **32.9** | 16h |
| DSR-7B | 30.2 | 76.5 | 35.8 | 87.2 | 47.5 | 55.4 | 50.5 | 76.1 | 19.2 | 48.6 | - |
| +GRPO | 35.2 | 87.7 | 38.1 | 92.4 | 54.9 | 61.7 | 50.9 | 75.6 | 23.8 | 50.1 | 31h |
| +ARPO | 36.4 | 87.5 | 38.6 | 91.4 | 57.1 | 62.2 | 51.6 | 75.7 | 24.4 | 50.6 | 31h |
| +MoPPS | 33.3 | 88.0 | 39.2 | 92.2 | 56.8 | 61.9 | 51.6 | 76.7 | 25.4 | **51.2** | 32h |
| +DAPO | 39.7 | 89.1 | 37.9 | 92.2 | 57.1 | 63.2 | 52.0 | 76.9 | 23.4 | 50.8 | 55h |
| +POPO | 39.4 | 88.9 | 38.0 | 92.6 | 57.7 | **63.3** | 52.1 | 76.3 | 25.2 | **51.2** | 34h |

Table 2: Evaluation on Countdown and Geometry. '+' denotes finetuning with the method.

| Method | Countdown | | | | | Method | Geometry | | |
|---|---|---|---|---|---|---|---|---|---|
| | CD4 | CD34 | Avg. | Runtime | Rollouts | | Avg. | Runtime | Rollouts |
| Qwen-3B | 0.5 | 2.6 | 1.5 | - | - | Qwen-VL-3B | 24.6 | - | - |
| +GRPO | 39.0 | 64.3 | 51.6 | 2.8h | **205k** | +GRPO | 44.9 | 6.8h | **492k** |
| +ARPO | 39.7 | 64.8 | 52.3 | 2.9h | **205k** | +ARPO | 45.7 | 6.5h | **492k** |
| +MoPPS | 40.7 | 66.3 | 53.5 | 3.4h | **205k** | +MoPPS | 48.9 | 7.4h | **492k** |
| +DAPO | 49.8 | 73.2 | **61.5** | 5.6h | 877k | +DAPO | **50.6** | 11.2h | 1438k |
| +POPO | 48.7 | 72.1 | 60.4 | 3.2h | **205k** | +POPO | 50.0 | 6.8h | **492k** |

replaying all recent groups regardless of their effectiveness, whereas POPO-stale(n) randomly replays historical effective groups from a larger buffer of size $n \times B$, regardless of their staleness. As shown in Fig. 5 (middle), POPO-ineff causes substantial performance degradation, validating the quality criterion. Furthermore, while POPO-stale(1) performs slightly worse, POPO-stale(10) collapses due to a large off-policy gap, underscoring the necessity of the off-policiness criterion.

**Off-policy Optimization Objectives.** To examine different off-policy optimization objectives, we fix the replay mechanism in POPO and test $\mathcal{J}_{\text{off-}\pi_{\text{old}}}$ and $\mathcal{J}_{\text{off-}\pi_\beta}$ as alternative off-policy objectives, denoting the resulting variants as POPO-$\pi_{\text{old}}$ and POPO-$\pi_\beta$. As shown in Fig. 5 (right), POPO-$\pi_\beta$ leads to a slowdown in performance improvements due to suboptimal trust-region constraints, while POPO-$\pi_{\text{old}}$ experiences performance collapse, possibly due to the bias in the optimization process. These results support the effectiveness of the decoupled off-policy optimization objective $\mathcal{J}_{\text{off-dec}}$.

### 5.4 Additional Analysis

**Sensitivity to Batch Replenishment Ratio.** POPO and DAPO address ineffective samples via different strategies: After filtering ineffective groups, POPO replenishes the batch with off-policy groups, whereas DAPO generates additional on-policy rollouts. To systematically compare these strategies, we vary the replenishment ratio and report results on Countdown. As shown in Fig. 6(a), performance increases monotonically as the proportion of effective data grows. Notably, POPO consistently matches DAPO in accuracy while maintaining a constant rollout cost.

**Sensitivity to Group Size.** We evaluate POPO, GRPO, and DAPO under varying response group sizes, specifically $k \in \{4, 8, 16, 32\}$. Fig. 6(b) reports the final average test accuracies on Countdown. The results show that POPO consistently outperforms GRPO and performs comparably to DAPO across all group sizes, demonstrating the generality of POPO.

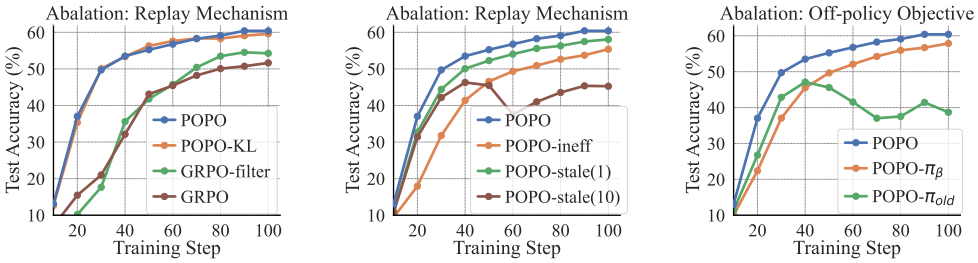



Figure 5: Ablation study on Countdown, including variants that use KL-based replay (POPO-KL), optimize only over on-policy effective groups (GRPO-filter), replay all recent groups regardless of effectiveness (POPO-ineff), replay stale effective groups (POPO-stale(n)), and employ alternative off-policy optimization objectives, $\mathcal{J}_{\text{off-}\pi_{\text{old}}}$ (POPO-$\pi_{\text{old}}$) and $\mathcal{J}_{\text{off-}\pi_\beta}$ (POPO-$\pi_\beta$).

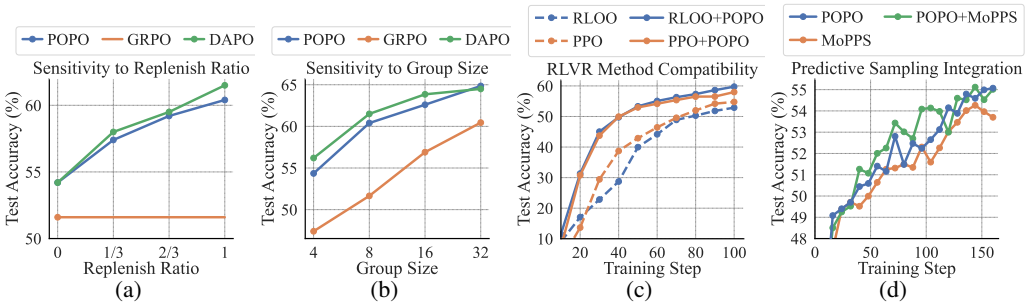



Figure 6: (a) Final performance of POPO and DAPO on Countdown under varying batch replenishment ratios. (b) Final performance of POPO, GRPO, and DAPO on Countdown under varying response group sizes. (c) Training curves on Countdown when applying POPO to RLOO and PPO. (d) Training curves on DeepScaleR 1.5B when integrating POPO with predictive sampling (MoPPS).

**Compatibility with Base RLVR Methods.** While our primary experiments focus on GRPO, POPO can be readily integrated into other base RLVR algorithms. We combine POPO with RLOO [1] and PPO [45], and test on the Countdown task. As shown in Fig. 6(c), incorporating POPO consistently improves both training efficiency and final accuracy across these methods. These findings highlight its broad applicability for improving sample efficiency in RLVR pipelines.

**Integration with Sampling Methods.** POPO is orthogonal to active prompt sampling methods and can be naturally integrated with them (e.g., MoPPS), potentially improving online data utilization. We compare POPO+MoPPS with standalone methods in Fig. 6(d) (full results, including batch composition ratios, in Fig. 8). Compared to POPO alone, the combination yields a higher ratio of online effective groups and faster policy improvement, indicating complementary benefits.

## 6 Conclusion and Limitations

This work presents POPO, a simple yet effective framework for improving sample efficiency in RL finetuning of LLMs. By replacing ineffective on-policy groups with high-quality off-policy groups through recency-based replay, POPO constructs training batches composed entirely of effective samples without incurring additional rollout cost. POPO further employs decoupled importance sampling to correct off-policy bias while maintaining a consistent trust-region constraint for stable optimization. Empirical results across diverse reasoning tasks show that POPO achieves performance comparable to the SOTA method while requiring substantially fewer rollouts.

While POPO achieves promising results, several limitations remain. For example, the replay mechanism does not necessarily select the most informative samples. Exploring hybrid strategies that combine short-term buffers with a long-term reservoir is promising. Moreover, future work may further reduce rollout costs by generating fewer online samples and leveraging more off-policy data.

### AI USE STATEMENT

In this work, we used generative AI tools solely to polish the writing of the paper and improve its readability. We did not use them for any task for which the ICLR 2027 AI Policy requires disclosure: the ideas, theory, methodology, implementation, experiments, and analysis of this work are entirely our own. We have reviewed all AI-assisted work. All AI-assisted edits were checked to confirm that they did not alter the technical content or claims of the paper. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

### ETHICS STATEMENT

This work studies the sample efficiency of RL finetuning for LLMs and, to the best of our knowledge, raises no ethical concerns. It involves no human subjects and no private or personally identifiable data; all experiments use publicly available datasets and pretrained models in accordance with their licenses. Broader societal impacts are discussed in Section F.

### REPRODUCIBILITY STATEMENT

We have made the following efforts to ensure that our results can be reproduced. The complete POPO algorithm is summarized as pseudocode in Algorithm 1. The assumptions and complete proofs of our theoretical claims are given in Section B. Section D reports the full experimental setup, covering the datasets, base models, evaluation protocol, and training hyperparameters, and Section G lists the prompt templates used in each task. Finally, we provide an anonymized implementation of POPO, together with the configuration files needed to launch the reported runs, as supplementary material.

## REFERENCES

[1] Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms. *arXiv preprint arXiv:2402.14740*, 2024.

[2] Sanghwan Bae, Jiwoo Hong, Min Young Lee, Hanbyul Kim, JeongYeon Nam, and Donghyun Kwak. Online difficulty filtering for reasoning oriented reinforcement learning. *arXiv preprint arXiv:2504.03380*, 2025.

[3] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[4] Xiaoyin Chen, Jiarui Lu, Minsu Kim, Dinghuai Zhang, Jian Tang, Alexandre Piché, Nicolas Gontier, Yoshua Bengio, and Ehsan Kamalloo. Self-evolving curriculum for llm reasoning. *arXiv preprint arXiv:2505.14970*, 2025.

[5] Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*, 2025.

[6] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

[7] Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025.

[8] Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. Safe rlhf: Safe reinforcement learning from human feedback. *arXiv preprint arXiv:2310.12773*, 2023.

[9] Quy-Anh Dang and Chris Ngo. Reinforcement learning for reasoning in small llms: What works and what doesn't. *arXiv preprint arXiv:2503.16219*, 2025.

[10] Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. Rlhf workflow: From reward modeling to online rlhf. *arXiv preprint arXiv:2405.07863*, 2024.

[11] Shihan Dou, Muling Wu, Jingwen Xu, Rui Zheng, Tao Gui, and Qi Zhang. Improving rl exploration for llm reasoning through retrospective replay. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pp. 594–606. Springer, 2025.

[12] Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, Jiashu Wang, et al. Areal: A large-scale asynchronous reinforcement learning system for language reasoning. *arXiv preprint arXiv:2505.24298*, 2025.

[13] Zhaolin Gao, Joongwon Kim, Wen Sun, Thorsten Joachims, Sid Wang, Richard Yuanzhe Pang, and Liang Tan. Prompt curriculum learning for efficient llm post-training. *arXiv preprint arXiv:2510.01135*, 2025.

[14] 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.

[15] Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng 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. *arXiv preprint arXiv:2402.14008*, 2024.

[16] Jacob Hilton, Karl Cobbe, and John Schulman. Batch size-invariance for policy optimization. *Advances in Neural Information Processing Systems*, 35:17086–17098, 2022.

[17] Hiyouga. Geometry3K: A large-scale multi-modal geometry reasoning dataset. https://huggingface.co/datasets/hiyouga/geometry3k, 2025.

[18] Jian Hu. Reinforce++: A simple and efficient approach for aligning large language models. *arXiv preprint arXiv:2501.03262*, 2025.

[19] Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*, 2025.

[20] 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.

[21] Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. Vineppo: Unlocking rl potential for llm reasoning through refined credit assignment. *arXiv preprint arXiv:2410.01679*, 2024.

[22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[23] 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*, 35:3843–3857, 2022.

[24] Siheng Li, Zhanhui Zhou, Wai Lam, Chao Yang, and Chaochao Lu. Repo: Replay-enhanced policy optimization. *arXiv preprint arXiv:2506.09340*, 2025.

[25] Jing Liang, Hongyao Tang, Yi Ma, Jinyi Liu, Yan Zheng, Shuyue Hu, Lei Bai, and Jianye Hao. Squeeze the soaked sponge: Efficient off-policy reinforcement finetuning for large language model. *arXiv preprint arXiv:2507.06892*, 2025.

[26] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

[27] Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*, 8(3):293–321, 1992.

[28] Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. Prorl: Prolonged reinforcement learning expands reasoning boundaries in large language models. *arXiv preprint arXiv:2505.24864*, 2025.

[29] 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.

[30] Fanbin Lu, Zhisheng Zhong, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Arpo: End-to-end policy optimization for gui agents with experience replay. *arXiv preprint arXiv:2505.16282*, 2025.

[31] Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. Inter-gps: Interpretable geometry problem solving with formal language and symbolic reasoning. In *The Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP 2021)*, 2021.

[32] Michael Luo, Sijun Tan, Roy Huang, Ameen Patel, Alpay Ariyak, Qingyang Wu, Xiaoxiang Shi, Rachel Xin, Colin Cai, Maurice Weber, et al. Deepcoder: A fully open-source 14b coder at o3-mini level. *Notion Blog*, 2025.

[33] Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Li Erran Li, et al. Deepscaler: Surpassing o1-preview with a 1.5 b model by scaling rl. *Notion Blog*, 2025.

[34] Fanqing Meng, Lingxiao Du, Zongkai Liu, Zhixiang Zhou, Quanfeng Lu, Daocheng Fu, Botian Shi, Wenhai Wang, Junjun He, Kaipeng Zhang, et al. Mm-eureka: Exploring visual aha moment with rule-based large-scale reinforcement learning. *CoRR*, 2025.

[35] Volodymyr Mnih. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[36] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[37] Youssef Mroueh, Nicolas Dupuis, Brian Belgodere, Apoorva Nitsure, Mattia Rigotti, Kristjan Greenewald, Jiri Navratil, Jarret Ross, and Jesus Rios. Revisiting group relative policy optimization: Insights into on-policy and off-policy training. In *International Conference on Learning Representations*, volume 2026, pp. 110991–111007, 2026.

[38] 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*, 35:27730–27744, 2022.

[39] Jiayi Pan, Junjie Zhang, Xingyao Wang, Lifan Yuan, Hao Peng, and Alane Suhr. Tinyzero. https://github.com/Jiayi-Pan/TinyZero, 2025. Accessed: 2025-01-24.

[40] Yun Qu, Qi Wang, Yixiu Mao, Vincent Tao Hu, Björn Ommer, and Xiangyang Ji. Can prompt difficulty be online predicted for accelerating rl finetuning of reasoning models? *arXiv preprint arXiv:2507.04632*, 2025.

[41] James Queeney, Yannis Paschalidis, and Christos G Cassandras. Generalized proximal policy optimization with sample reuse. *Advances in Neural Information Processing Systems*, 34: 11909–11919, 2021.

[42] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

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

[44] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

[45] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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

[47] Qianli Shen, Daoyuan Chen, Yilun Huang, Zhenqing Ling, Yaliang Li, Bolin Ding, and Jingren Zhou. Bots: A unified framework for bayesian online task selection in llm reinforcement finetuning. *arXiv preprint arXiv:2510.26374*, 2025.

[48] 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. *arXiv preprint arXiv: 2409.19256*, 2024.

[49] Yifan Sun, Jingyan Shen, Yibin Wang, Tianyu Chen, Zhendong Wang, Mingyuan Zhou, and Huan Zhang. Improving data efficiency for llm reinforcement fine-tuning through difficulty-targeted online data selection and rollout replay. *arXiv preprint arXiv:2506.05316*, 2025.

[50] Zhiqing Sun, Sheng Shen, Shengcao Cao, Haotian Liu, Chunyuan Li, Yikang Shen, Chuang Gan, Liang-Yan Gui, Yu-Xiong Wang, Yiming Yang, et al. Aligning large multimodal models with factually augmented rlhf. *arXiv preprint arXiv:2309.14525*, 2023.

[51] Xinyu Tang, Zhenduo Zhang, Yurou Liu, Wayne Xin Zhao, Zujie Wen, Zhiqiang Zhang, and Jun Zhou. Towards high data efficiency in reinforcement learning with verifiable reward. *arXiv preprint arXiv:2509.01321*, 2025.

[52] Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025.

[53] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290, 2024.

[54] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[55] Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, and Chong Luo. Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning. *arXiv preprint arXiv:2502.14768*, 2025.

[56] Guowei Xu, Peng Jin, Li Hao, Yibing Song, Lichao Sun, and Li Yuan. Llava-o1: Let vision language models reason step-by-step. *arXiv preprint arXiv:2411.10440*, 2024.

[57] Zhongwen Xu and Zihan Ding. Single-stream policy optimization. *arXiv preprint arXiv:2509.13232*, 2025.

[58] Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. Learning to reason under off-policy guidance. *arXiv preprint arXiv:2504.14945*, 2025.

[59] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv e-prints*, pp. arXiv–2412, 2024.

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

[61] 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.

[62] Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, et al. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*, 2025.

[63] Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *arXiv preprint arXiv:2503.18892*, 2025.

[64] Runzhe Zhan, Yafu Li, Zhi Wang, Xiaoye Qu, Dongrui Liu, Jing Shao, Derek F Wong, and Yu Cheng. Exgrpo: Learning to reason from experience. *arXiv preprint arXiv:2510.02245*, 2025.

[65] Hongzhi Zhang, Jia Fu, Jingyuan Zhang, Kai Fu, Qi Wang, Fuzheng Zhang, and Guorui Zhou. Rlep: Reinforcement learning with experience replay for llm reasoning. *arXiv preprint arXiv:2507.07451*, 2025.

[66] Kaiyan Zhang, Yuxin Zuo, Bingxiang He, Youbang Sun, Runze Liu, Che Jiang, Yuchen Fan, Kai Tian, Guoli Jia, Pengfei Li, et al. A survey of reinforcement learning for large reasoning models. *arXiv preprint arXiv:2509.08827*, 2025.

[67] Xiaojiang Zhang, Jinghui Wang, Zifei Cheng, Wenhao Zhuang, Zheng Lin, Minglei Zhang, Shaojie Wang, Yinghan Cui, Chao Wang, Junyi Peng, et al. Srpo: A cross-domain implementation of large-scale reinforcement learning on llm. *arXiv preprint arXiv:2504.14286*, 2025.

[68] 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.

[69] Chujie Zheng, Kai Dang, Bowen Yu, Mingze Li, Huiqiang Jiang, Junrong Lin, Yuqiong Liu, Hao Lin, Chencan Wu, Feng Hu, et al. Stabilizing reinforcement learning with llms: Formulation and practices. *arXiv preprint arXiv:2512.01374*, 2025.

[70] Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

[71] 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. *arXiv preprint arXiv:2506.02177*, 2025.

# A Related Work

**RL Finetuning for LLMs.** RL has emerged as a central paradigm for adapting LLMs to complex tasks and target behaviors. Among its variants, Reinforcement Learning with Human Feedback (RLHF) has demonstrated strong effectiveness in aligning LLM outputs with human preferences and safety requirements [38; 10; 42; 8; 50; 48]. In settings where reward signals are verifiable, such as mathematical problem solving or program synthesis, Reinforcement Learning with Verifiable Rewards (RLVR) has led to substantial improvements in LLM reasoning performance [20; 46; 52; 5; 14]. From an algorithmic standpoint, Proximal Policy Optimization (PPO) [45], a widely used policy-gradient method, can be directly applied to LLM finetuning. Building on this foundation, Group Relative Policy Optimization (GRPO) [46] removes the need for a separate value network by employing a group-normalized advantage estimator, significantly reducing computational overhead. As a result, GRPO has quickly become a prevalent choice for RLVR. Subsequent research has further refined RLVR by addressing gradient bias, improving training stability, and enhancing computational efficiency [61; 62; 29; 60; 21; 18; 70]. On the application front, recent efforts have extended RL finetuning to a wider range of tasks and increasingly large-scale models [33; 9; 32; 63; 34; 56]. In parallel, infrastructure-level advances have produced scalable and resource-efficient frameworks that support distributed RL training tailored to the demands of LLMs [48; 19].

**Active Sampling for RL Finetuning.** A growing body of work emphasizes that the effectiveness of RL finetuning is severely constrained by ineffective training samples [60], which are either too easy or too difficult, yielding degenerate rewards and limited optimization signals [2; 4]. To address this, recent studies explore active prompt sampling strategies that dynamically select informative prompts on a per-step or per-epoch basis [2; 67; 7; 28]. A representative SOTA method, DAPO [60], oversamples and filters prompts using rollout feedback, improving sample quality but incurring substantial overhead from enlarged batch generation. To avoid this cost, predictive sampling methods aim to identify prompts with intermediate success probability prior to rollout [40; 47; 57; 71]. These methods typically estimate the current success rate of each prompt based on historical reward statistics, employing Bayesian inference or heuristic modeling. For example, GRESO [71] proposes selective rollouts to reduce unnecessary generations, while PCL [13] estimates prompt difficulty for prioritized sampling. Nevertheless, accurate prediction is difficult under continually evolving policies and sparse observations, often limiting the number of effective samples. In contrast, POPO avoids additional rollout cost by reusing previously collected effective groups, enabling full batches of effective data for optimization and offering an orthogonal perspective focused on post-hoc utilization rather than pre-rollout filtering.

**Experience Replay for RL Finetuning.** Experience replay is an important technique in classical RL to improve sample efficiency [27; 35; 36; 44] and has recently gained traction in RL finetuning due to the high online generation cost of LRMs [66]. Recent studies show that replaying certain trajectories can accelerate training. For example, ARPO [30] and AR3PO [68] store historical successful trajectories and substitute one response when all online rollouts in a group fail. RLEP [65] reuses successful trajectories from a well-trained policy and re-trains a policy by appending successful trajectories to online groups. ExGRPO [64] identifies rollout correctness and entropy as indicators of experience value, and mixes past correct responses with the lowest entropy into online groups. RRL [11] selectively replays promising earlier reasoning traces to continue exploring them. DEPO [51] introduces a two-stage filtering and replay strategy with offline data selection and online explorability-based filtering to improve data efficiency. RR [49] generates fewer online rollouts and reuses past ones to fill a fixed fraction of batches, thereby reducing rollout cost. ReMix [25] revisits the generalized proximal gradient theory [41] and introduces mix-policy proximal policy gradient in RLVR. Beyond replay, Mroueh et al. [37] analyze off-policy GRPO and show that its policy-improvement guarantee weakens as the behavior policy drifts and as group reward variance vanishes. However, these methods do not fully resolve the issue of ineffective samples. Moreover, their off-policy optimization objectives may be subject to issues such as intractable bias, suboptimal constraints, or invalid advantage estimates. In contrast, POPO addresses these limitations and leverages high-quality off-policy optimization signals to accelerate RL finetuning.

# B PROOFS

## B.1 PROOF OF LEMMA 4.1

**Lemma 4.1.** *Assume the per-step parameter update is bounded, $\|\theta_{i+1} - \theta_i\|_2 \le \Delta$ for all steps $i$, and that $\theta \mapsto \log \pi_\theta(a|s)$ is $L$-Lipschitz over the parameter region visited during training. Then, for any step difference $n \ge 1$, $\mathrm{TV}\big(\pi_i(\cdot|s), \pi_{i+n}(\cdot|s)\big) \le \frac{n}{2}\big(e^{L\Delta} - 1\big) = \frac{nL\Delta}{2} + O\big(n(L\Delta)^2\big)$.*

*Proof.* Fix a state $s$ and a step index $j$. For every action $a$, the $L$-Lipschitzness of $\theta \mapsto \log \pi_\theta(a|s)$ together with the bounded-update assumption gives

$$\left|\log \frac{\pi_{j+1}(a|s)}{\pi_j(a|s)}\right| \;\le\; L\,\|\theta_{j+1} - \theta_j\|_2 \;\le\; L\Delta. \tag{11}$$

Write $\delta_a$ for the log-ratio in Eq. (11), so that $\pi_{j+1}(a|s) = \pi_j(a|s)\, e^{\delta_a}$ with $|\delta_a| \le L\Delta$. Since $|e^\delta - 1| \le e^{|\delta|} - 1$ for every real $\delta$,

$$\big|\pi_{j+1}(a|s) - \pi_j(a|s)\big| \;=\; \pi_j(a|s)\,\big|e^{\delta_a} - 1\big| \;\le\; \pi_j(a|s)\big(e^{L\Delta} - 1\big). \tag{12}$$

Summing over the action set and using $\sum_a \pi_j(a|s) = 1$ yields the one-step bound

$$\mathrm{TV}\big(\pi_j(\cdot|s), \pi_{j+1}(\cdot|s)\big) \;=\; \frac{1}{2}\sum_a \big|\pi_{j+1}(a|s) - \pi_j(a|s)\big| \;\le\; \frac{1}{2}\big(e^{L\Delta} - 1\big). \tag{13}$$

Total variation is a metric, so for any $n \ge 1$ the triangle inequality applies along the sequence of intermediate policies, and Eq. (13) bounds each term:

$$\mathrm{TV}\big(\pi_i(\cdot|s), \pi_{i+n}(\cdot|s)\big) \;\le\; \sum_{j=0}^{n-1} \mathrm{TV}\big(\pi_{i+j}(\cdot|s), \pi_{i+j+1}(\cdot|s)\big) \;\le\; \frac{n}{2}\big(e^{L\Delta} - 1\big). \tag{14}$$

Expanding the exponential gives $\frac{nL\Delta}{2} + O\big(n(L\Delta)^2\big)$. Since $s$ was arbitrary, the bound holds at every state. □

## B.2 PROOF OF LEMMA 4.2

**Lemma 4.2.** *$\mathcal{J}_{\text{off-dec}}$ and $\mathcal{J}_{\text{off-}\pi_\beta}$ differ only in their trust-region constraints: the former clips $\rho_{i,t}(\theta, \theta_{old})$, whereas the latter clips $\rho_{i,t}(\theta, \beta)$. For any token at which neither ratio is clipped, the two objectives yield identical values and gradients.*

*Proof.* Fix a token $(i, t)$ and define

$$r = \rho_{i,t}(\theta, \theta_{\text{old}}), \qquad w = \rho_{i,t}(\theta_{\text{old}}, \beta), \qquad \hat{A} = \hat{A}_i, \tag{15}$$

Note that $\rho_{i,t}(\theta, \beta) = rw$. The per-token contributions of the two objectives $\mathcal{J}_{\text{off-}\pi_\beta}$ and $\mathcal{J}_{\text{off-dec}}$ are

$$L_\beta = \min\big(rw\hat{A},\ \mathrm{clip}(rw,\, 1 \pm \epsilon)\,\hat{A}\big), \qquad L_{\text{dec}} = w \cdot \min\big(r\hat{A},\ \mathrm{clip}(r,\, 1 \pm \epsilon)\,\hat{A}\big). \tag{16}$$

Thus $\mathcal{J}_{\text{off-}\pi_\beta}$ clips $rw$ while $\mathcal{J}_{\text{off-dec}}$ clips $r$, differing in the argument of the trust-region clipping.

Consider any token where neither ratio is clipped by its respective clipping operator. This means

$$r \in [1 - \epsilon,\, 1 + \epsilon] \quad \text{and} \quad rw \in [1 - \epsilon,\, 1 + \epsilon]. \tag{17}$$

Under these conditions,

$$L_\beta = \min\big(rw\hat{A},\ rw\hat{A}\big) = w \cdot \min\big(r\hat{A},\ r\hat{A}\big) = L_{\text{dec}}. \tag{18}$$

Thus $L_\beta = L_{\text{dec}}$ pointwise. Because the outer expectation over $q$ and $o_i \sim \pi_\beta$ is identical for both objectives, their total values coincide on the set of unclipped tokens. For gradients, the advantage $\hat{A}$ is treated as constant and $w = \rho_{i,t}(\theta_{\text{old}}, \beta)$ is independent of $\theta$, so

$$\nabla_\theta L_\beta = \hat{A}\,\nabla_\theta(rw) = \hat{A}\,w\,\nabla_\theta r = \nabla_\theta L_{\text{dec}}. \tag{19}$$

Hence on any unclipped token the two objectives coincide in value and gradient. □

## C DISCUSSIONS

**Behavior Policy under Selective Replay.** POPO performs prioritized replay at the group level, which preserves a well-defined behavior policy for each prompt. Specifically, all trajectories within a group are generated by the same rollout policy (i.e., the LLM policy at that training step), ensuring that the replayed data follows a tractable behavior distribution. This property enables principled importance sampling for off-policy correction. In contrast, prior trajectory-level replay methods perform selective replay at the level of individual trajectories within a prompt. Such filtering effectively samples from a truncated and implicit subset of the original behavior policy, resulting in an intractable distribution. Consequently, the corresponding importance correction becomes ill-defined, leading to unresolved bias. This distinction highlights a key advantage of group-level replay in maintaining both statistical correctness and optimization stability.

**Sensitivity to Off-policy Gap and Robustness.** Our ablations in Sec. 5.3 show that excessive off-policy drift, induced by intentionally replaying stale groups (POPO-stale($n$)), leads to performance degradation. POPO is explicitly designed to mitigate this issue via (i) recency-based replay, which limits distribution shift, and (ii) decoupled importance sampling, which corrects residual bias while maintaining stable trust-region updates. As shown in Sec. 5.3, alternative replay strategies and objectives result in inferior performance, supporting the importance of these design choices. At the same time, POPO is not overly sensitive to moderate variations in recency: for example, POPO-stale(1) incurs only minor degradation and still significantly outperforms GRPO.

**Robustness to Replay-induced Overfitting.** POPO is designed to mitigate potential overfitting under replay and maintain strong generalization. Specifically, several mechanisms contribute to this robustness. First, replay is bounded: the FIFO buffer has a small, fixed capacity and is continuously refreshed with newly collected effective groups, preventing the training data from collapsing to a static subset. Second, replay is used only to fill ineffective slots rather than replacing the entire batch, which preserves continuous exposure to fresh, randomly sampled prompts and responses. Third, as the policy model evolves, the set of effective samples also changes dynamically; it is not tied to a fixed subset of prompts. Combined with random online prompt sampling, this ensures continual exploration and discovery of new effective samples rather than repeatedly focusing on the same ones. Empirically, we do not observe signs of overfitting (e.g., performance degradation or instability); instead, POPO consistently improves both sample efficiency and generalization across tasks.

**Sample Reuse and Off-policy Control.** POPO adopts a small FIFO buffer with recency-based replay, which may discard earlier useful groups. This design is intentional to bound the off-policy gap: under proximal updates, recent policies are theoretically closer to the current policy (Lemma 4.1), making replayed samples more reliable for off-policy correction. Empirical results further support this choice. Our ablations show that replaying increasingly stale groups degrades performance, with POPO-stale(10) collapsing, while small recency windows remain stable. This design reflects a deliberate trade-off between sample reuse and off-policy control, prioritizing stability and reliable optimization over the reuse of older data. Nevertheless, retaining certain informative historical groups may still be beneficial. Exploring hybrid strategies, such as combining short-term buffers with a small long-term reservoir to balance recency and diversity without significantly increasing off-policy gaps, is a promising direction for future work.

**Variance Control in Importance Sampling.** POPO effectively controls variance in importance sampling, even under policy drift, through two mechanisms. First, recency-based replay keeps behavior policies close to the current policy, directly reducing the magnitude of importance weights. Second, we apply clipping (with an upper bound of 2.0) to $\rho_{i,t}(\theta_{\text{old}}, \beta)$ to suppress outliers in practice. Empirically, we do not observe instability arising from variance, suggesting that these mechanisms are effective.

**Generalization to Diverse Reward Structures.** While Definition 3.1 is motivated by common binary-reward RLVR settings, the underlying principle extends naturally to more general reward structures, including continuous or multi-level rewards. In such cases, the strict zero-variance condition can be relaxed to a small-variance threshold or other low-information criteria that similarly yield weak learning signals. Thus, the definition is task-specific, while the POPO framework itself

does not fundamentally rely on strict binary rewards or exact zero variance; instead, it operates on the presence of informative relative differences across groups.

# D Experimental Details

## D.1 Details of Tasks and Models

We evaluate POPO across three distinct and challenging reasoning domains: competition-level mathematics, numerical planning, and visual geometric reasoning. To assess its general applicability, conduct experiments using a diverse set of large language and multimodal models spanning different capacities and architectures. We fine-tune all models using the widely adopted GRPO algorithm implemented in the verl framework [48]. Model performance is measured by test accuracy, reported as the average pass@1 over multiple independent completions per example, with the number of generations varying across benchmarks. The training datasets, evaluation benchmarks, and base models for each domain are described below, with illustrative data examples provided in Section G.

### D.1.1 Competition-level Mathematics

**Training Dataset.** For competition-level mathematics, we train large reasoning models on the DeepScaleR dataset [33], which consists of 40,315 problems designed to reflect competition-level difficulty. Specifically, we use the Hugging Face release from https://huggingface.co/datasets/agentica-org/DeepScaleR-Preview-Dataset.

**Test Benchmarks.** We evaluate mathematical reasoning ability on a diverse collection of benchmarks, including AIME24, AMC23, MATH500 [26], Minerva Math [23], and OlympiadBench [15], using the datasets hosted at https://huggingface.co/datasets/math-ai. Training curves summarize results as the average accuracy across these mathematics benchmarks. To examine generalization beyond the training distribution, we further conduct evaluations on three broader reasoning benchmarks: MMLU-Pro [53], ARC-c [6], and GPQA-diamond (GPQA) [43], using the datasets provided by Yan et al. [58]. These benchmarks contain general reasoning problems in domains such as science and business. Following prior work [13], we adopt mixed Avg@k settings for evaluation, using larger $k$ for smaller datasets to reduce variance: Avg@32 for AIME25 and AMC23, Avg@4 for Minerva Math, Avg@1 for MATH500 and OlympiadBench, and Avg@8 for OOD benchmarks (MMLU-Pro, ARC-c, and GPQA-diamond).

**Base Models.** Following prior work [33], we adopt two base models from DeepSeek [14]: DeepSeek-R1-Distill-Qwen-1.5B, available at https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-1.5B and DeepSeek-R1-Distill-Qwen-7B, available at https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-7B.

### D.1.2 Numerical Planning

**Training Dataset.** For arithmetic planning, we use the Countdown Number Game, in which agents must form a target value using basic operations over a given set of numbers [39]. Training is conducted on a 20,000-instance subset of the full Countdown-34 dataset, available from the Hugging Face repository at https://huggingface.co/datasets/Jiayi-Pan/Countdown-Tasks-3to4.

**Test Benchmarks.** Models are evaluated on two benchmarks: (i) CD-34, which contains 512 held-out instances from Countdown-34; and (ii) CD-4, comprising 512 problems from Countdown-4, a more challenging variant that uses four input numbers, available at https://huggingface.co/datasets/Jiayi-Pan/Countdown-Tasks-4. Compared with CD-34, CD-4 substantially expands the search space and increases task difficulty. We report Avg@16 for both benchmarks, and training curves show the average test performance across CD-34 and CD-4.

**Base Models.** Following prior work [4; 40], we finetune the base model Qwen2.5-3B from Qwen [59], availiable at https://huggingface.co/Qwen/Qwen2.5-3B.

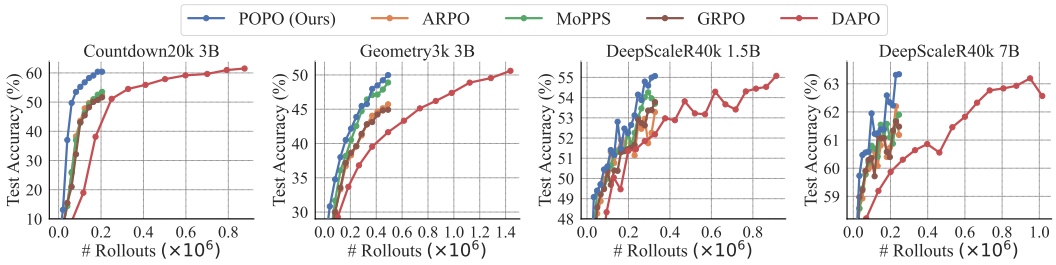



Figure 7: Training curves of different methods across tasks as a function of LLM rollout count.

#### D.1.3 Visual Geometry

**Training Dataset.** For visual geometry, we use the training split of the Geometry3k dataset [31; 17], available at https://huggingface.co/datasets/hiyouga/geometry3k. This dataset contains 2,101 diagram-based geometry problems that require both visual interpretation and symbolic reasoning.

**Test Benchmarks.** We assess the trained models on the corresponding benchmark test split, which comprises 601 visual reasoning problems, and report Avg@16.

**Base Models.** For visual geometric reasoning, we follow Qu et al. [40] and employ the vision-language model Qwen2.5-VL-3B-Instruct from Qwen [3], available at https://huggingface.co/Qwen/Qwen2.5-VL-3B-Instruct.

### D.2 Training Details

In the main experiments, both our approach and all baselines are built upon GRPO [46] as the underlying RLVR algorithm. To ensure a fair comparison, we adopt the same GRPO implementation provided in the verl framework [48], as detailed below. At each training step, $k = 8$ responses are sampled per prompt to estimate advantages and finetune models, using a temperature of 1.0 and $\texttt{top_p} = 1.0$. Performance is measured with pass@1, computed across multiple independent generations for each prompt, with the number of samples varying by benchmark. For evaluation, we follow Luo et al. [33]; Qu et al. [40] and generate responses with a temperature of 0.6 and $\texttt{top_p} = 0.95$. We remove the KL-divergence penalty from the policy optimization loss, consistent with Yu et al. [60]. In addition, all methods employ the clip-higher technique from DAPO [60], which uses asymmetric clipping bounds with $\epsilon_{\text{low}} = 0.2$ and $\epsilon_{\text{high}} = 0.28$ to enhance exploration.

The training batch size $B$ is set to 256 for both DeepScaleR and Countdown, with mini-batch sizes of 128 and 64, respectively. For Geometry, $B$ is increased to 512 with a mini-batch size of 256. The maximum output length is set to 8192 tokens for DeepScaleR and 1024 tokens for Countdown and Geometry. Entropy regularization is applied to Countdown with a coefficient of 0.001, while it is disabled for DeepScaleR and Geometry. Models is optimized using Adam [22] with momentum parameters $\beta = (0.9, 0.999)$ and a weight decay of 0.01. The learning rate is set to $1\text{e}{-6}$ for Countdown and Geometry. For DeepScaleR, we use $2\text{e}{-6}$ for the 7B model and $4\text{e}{-6}$ for the 1.5B model. DeepScaleR adopts a binary reward function that assigns a value of 1 to correct answers and 0 otherwise, following the default configuration in verl [48]. In contrast, Countdown and Geometry3k incorporate an additional format bonus of 0.1 when the response is incorrectly solved but adheres to the required formatting, consistent with the setup in [39]. All experiments are executed on 8 NVIDIA A100 GPUs with 80GB memory.

## E Extended Experimental Results

### E.1 Rollout Efficiency

Figure 2 shows that POPO and DAPO substantially accelerate RL fine-tuning compared to the other baselines when measured by training steps. However, training-step-based comparisons fail to account

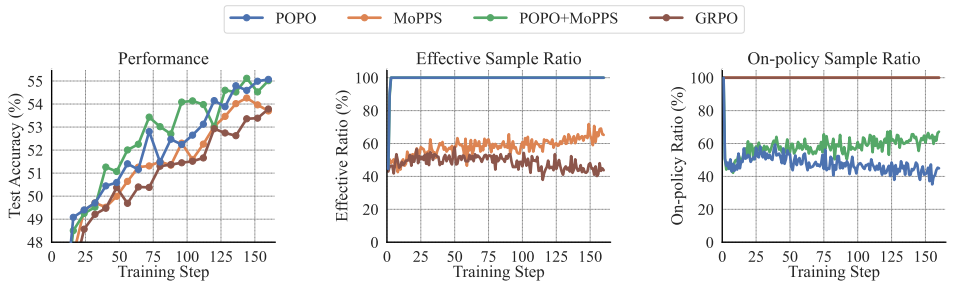



Figure 8: Test performance, effective sample ratio, and on-policy sample ratio on the DeepScaleR 1.5B task when integrating POPO with the predictive sampling method MoPPS.

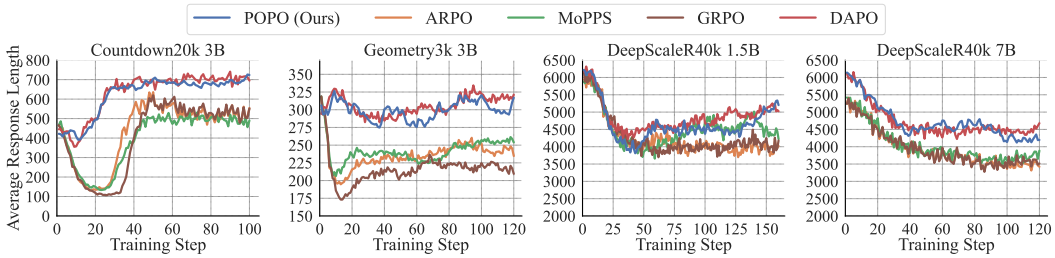



Figure 9: Comparison of the average response length across different RLVR methods.

for the cost of LLM inference, which often dominates the overall training expense. Because DAPO relies on oversampling, it generates a larger number of rollouts per step, leading to significantly higher inference overhead. Figure 7 instead plots performance as a function of the cumulative number of rollouts. The results demonstrate that POPO achieves strong performance using markedly fewer rollouts than DAPO, typically requiring 30% of DAPO's rollout budget to reach comparable results.

### E.2 Integration with Sampling Methods

POPO can be naturally combined with sampling methods that adaptively select informative prompts. By incorporating predictive sampling (e.g., MOPPS), POPO increases the likelihood of sampling effective groups online, thereby reducing the need to fill training batches with off-policy effective groups and potentially yielding further performance gains. To investigate this effect, we integrate MOPPS with POPO, denoted as POPO+MOPPS, and compare it against standalone POPO, MOPPS, and GRPO on the DeepScaleR 1.5B task. Fig. 8 reports the full results, including test performance, effective sample ratio, and on-policy sample ratio throughout training.

As shown in Fig. 8, POPO +MOPPS maintains a 100% effective sample ratio, the same as POPO, while achieving a higher on-policy sample ratio. This shift toward more on-policy data leads to additional performance gains, as evidenced by faster improvement and higher test accuracy. These results indicate that combining predictive sampling with POPO enables a more favorable balance between on-policy and off-policy data, yielding complementary benefits for policy optimization.

### E.3 Response Length

Response length has been identified as a strong correlate of reasoning capability [60]. Figure 9 illustrates how different methods affect this metric over the course of training. Across tasks, the average response length of POPO follows a trajectory similar to that of DAPO, whereas ARPO and MoPPS exhibit trends more closely aligned with GRPO. This suggests that POPO leverages off-policy data to induce training dynamics comparable to those of DAPO, which relies solely on online data. Both POPO and DAPO consistently produce longer responses than the remaining baselines. Such extended outputs encourage deeper exploration and support more elaborate reasoning processes, partly explaining the observed performance differences [60].

## F Broader Societal Impacts

This work presents POPO, a sample-efficient framework for RL finetuning of LLMs. By replacing ineffective training data with high-quality off-policy samples, POPO reduces the number of model rollouts required during training, thereby lowering computational overhead and associated energy consumption. Improved efficiency at this stage has the potential to broaden access to the development of advanced reasoning models, particularly for researchers and organizations with limited computational resources. At the same time, as with any method that enhances the capabilities of LLMs, there remains the possibility of misuse. However, POPO does not introduce fundamentally new risks beyond those already inherent in existing RLVR paradigms; rather, it improves the efficiency of established training processes without altering their core objectives or deployment characteristics. The societal impact of such technology ultimately depends on the specific downstream applications and the safeguards employed during deployment. We therefore emphasize the importance of continued research into responsible AI practices to ensure that gains in training efficiency are directed toward positive and broadly beneficial outcomes.

## G Data Examples

We present representative examples of the data used across all tasks in our experiments. For the DeepScaleR dataset and mathematics benchmarks, prompts are constructed by appending a chain-of-thought instruction [54] along with a formatting constraint: "Let's think step by step and output the final answer within \boxed{}". For general reasoning benchmarks, we follow the evaluation protocol of Yan et al. [58] and adopt the PRIME prompt template. For the Countdown task, we use the prompt format introduced by Pan et al. [39]. For the Geometry task, we adopt the prompt templates from verl [48].

**DeepscaleR Data Example**

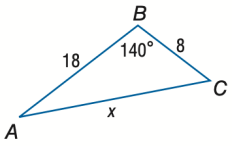
**Prompt:**
Consider all 1000-element subsets of the set $\{1, 2, 3, \ldots, 2015\}$. From each such subset choose the least element. The arithmetic mean of all of these least elements is $\frac{p}{q}$, where $p$ and $q$ are relatively prime positive integers. Find $p + q$. Let's think step by step and output the final answer within `\boxed{}`.
**Answer:**
2016

**Countdown Data Example**

**Prompt:**
A conversation between User and Assistant. The user asks a question, and the Assistant solves it. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer.
User: Using the numbers [95, 11, 56], create an equation that equals 28. You can use basic arithmetic operations (+, -, *, /) and each number can only be used once. Show your work in <think> < /think> tags. And return the final answer in <answer> < /answer> tags, for example <answer> $(1 + 2)/3$ </answer>.
Assistant: Let me solve this step by step.
<think>

**Geometry3k Data Example**

**Prompt:**




Find x. Round to the nearest tenth. You FIRST think about the reasoning process as an internal monologue and then provide the final answer. The reasoning process MUST BE enclosed within <think> < /think> tags. The final answer MUST BE put in `\boxed{}`.

**Answer:**

24.7

**MMLU-Pro Data Example**

**Prompt:**

The following are multiple choice questions (with answers) about $. Think step by step and then finish your answer with "`\boxed{X}`" where X is the correct letter choice.

Question:

Managers are entrusted to run the company in the best interest of _____. Specifically, they have a duty to act for the benefit of the company, as well as a duty of _____ and of _____.

Options:

A. Shareholders, Diligence, Self-interest
B. Shareholders, Self-interest, Care and Skill
C. Stakeholders, Care and skill, Self-interest
D. Stakeholders, Diligence, Care and Skill
E. Customers, Care and Skill, Diligence
F. Shareholders, Care and Skill, Diligence
G. Shareholders, Self-interest, Diligence
H. Employees, Care and Skill, Diligence
I. Stakeholders, Self-interest, Diligence
J. Stakeholder, Care and Skill, Diligence

**Answer:**

F

**GPQA-Diamond Data Example**

**Prompt:**

The following are multiple choice questions (with answers) about $. Think step by step and then finish your answer with "`\boxed{X}`" where X is the correct letter choice.

Question:

Astronomers are studying a system of three exoplanets (Planet1, Planet2, and Planet3) with circular orbits discovered through the TTV method. They have found that the ratio of the equilibrium temperatures between Planet1 and Planet2 is approximately 1.4, and between Planet2 and Planet3, it is about 2.3. They have also found that the ratio of the masses between Planet1 and Planet2 is approximately 1.15, and between Planet2 and Planet3, it is about 1.35. By what factor is the orbital period of Planet3 larger than that of Planet1, if the albedo for all three planets is equal to 0.3 (similar to that of Earth)?

Options:

A. $\sim$ 3.2
B. $\sim$ 4.4
C. $\sim$ 10.4
D. $\sim$ 33.4

**Answer:**

D

**ARC-C Data Example**

**Prompt:**
The following are multiple choice questions (with answers) about $. Think step by step and then finish your answer with "`\boxed{X}`" where X is the correct letter choice.
Question:
George wants to warm his hands quickly by rubbing them. Which skin surface will produce the most heat?
Options:
A. dry palms
B. wet palms
C. palms covered with oil
D. palms covered with lotion
**Answer:**
A