Reinforce-Ada: An Adaptive Sampling Framework for Reinforce-Style LLM Training

Wei Xiong[*◦] **Chenlu Ye**[*◦] **Baohao Liao**[*■] **Hanze Dong**[*♢]
Xinxing Xu[♢] **Christof Monz**[■] **Jiang Bian**[♢] **Nan Jiang**[◦] **Tong Zhang**[◦]

[◦] *University of Illinois Urbana-Champaign* [♢] *Microsoft Research* [■] *University of Amsterdam*

Abstract

Reinforcement learning applied to large language models (LLMs) for reasoning tasks is often bottlenecked by unstable gradient estimates due to fixed and uniform sampling of responses across prompts. Prior work such as GVM-RAFT addresses this by dynamically allocating inference budget per prompt to minimize stochastic gradient variance under a budget constraint. Inspired by this insight, we propose Reinforce-Ada, an adaptive sampling framework for online RL post-training of LLMs that continuously reallocates sampling effort to the prompts with the greatest uncertainty or learning potential. Unlike conventional two-stage allocation methods, Reinforce-Ada interleaves estimation and sampling in an online successive elimination process, and automatically stops sampling for a prompt once sufficient signal is collected. To stabilize updates, we form fixed-size groups with enforced reward diversity and compute advantage baselines using global statistics aggregated over the adaptive sampling phase. Empirical results across multiple model architectures and reasoning benchmarks show that Reinforce-Ada accelerates convergence and improves final performance compared to GRPO, especially when using the balanced sampling variant. Our work highlights the central role of variance-aware, adaptive data curation in enabling efficient and reliable reinforcement learning for reasoning-capable LLMs. Code is available at `https://github.com/RLHFlow/Reinforce-Ada`.

```
  # 1. prepare batch of prompts gen_batch

  # 2. Generation API
-  gen_batch_output = self.generate_sequences (gen_batch)

+  gen_batch_output, rounds_info =
+    self.generate_multi_round_adaptive_downsampling(
+    orig_prompt_batch=gen_batch,
+    max_round=max_round, # maximum downsampling rounds
+    round_repeat=round_repeat, # rollout numbers per prompt ineach round
+    final_keep_per_prompt=final_keep_per_prompt, # finall kept number per prompt
+    context_batch=batch, # original prompts
+  )

  # 3. Union outputs to batch
  batch = batch.union(gen_batch_output)

  # 4. Compute rewards, advantage and update model ...
```

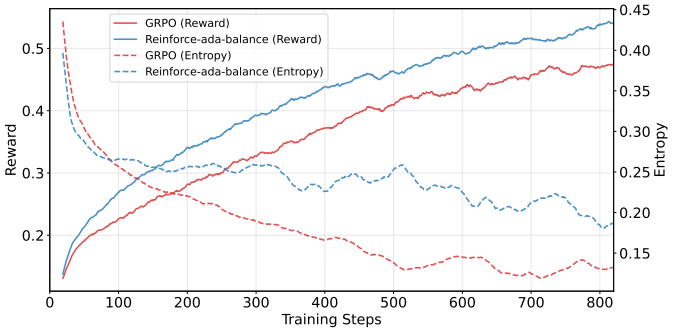

Figure 1: Plug-and-play usage. Left: a one-line swap of the generation API in `verl` (`generate_sequences` → `generate_multi_round_adaptive_downsampling`). Right: with no other changes, training attains faster reward growth and a higher asymptote than GRPO.

*Equal contribution. A detailed attribution of authorship credits is provided in Appendix A. Correspondence to Hanze Dong (`hanzedong@microsoft.com`) and Wei Xiong (`wx13@illinois.edu`).

1 Introduction

Reinforcement learning (RL) has become a central paradigm for aligning large language models (LLMs). At its core, training reduces to maximizing the expected reward of model responses under a prompt distribution. The challenge is not the objective itself, but the variance of gradient estimates, which makes learning unstable.

Formally, for a prompt $x \sim d_0$, the policy π_θ produces a response $a \sim \pi_\theta(\cdot|x)$, which a verifier scores as $r^\star(x, a) \in \{0, 1\}$. The learning objective is

$$J(\theta) = \mathbb{E}_{x\sim d_0, a\sim\pi_\theta(\cdot|x)}[r^\star(x, a)]. \tag{1}$$

Estimating its gradient requires sampling multiple responses. With only a few samples per prompt, inference is affordable but the gradient is noisy; with many samples, the signal is clear but inference becomes prohibitively expensive. The trade-off between signal quality and cost is a central challenge in RL for LLMs.

Vanilla policy gradient with small n has notoriously high variance. A standard remedy introduces a baseline $b(x)$, yielding

$$g_\theta(x, a) = (r^\star(x, a) - b(x)) \cdot \nabla_\theta \log \pi_\theta(a|x),$$

which stabilizes training while preserving unbiasedness.

Group Relative Policy Optimization (GRPO) (Shao et al., 2024) extends this principle by assigning n responses per prompt and normalizing each sample's advantage:

$$A_{\text{GRPO}}(x, a_i) = \frac{r_i - \bar{r}}{\sigma_r + \varepsilon}, \tag{2}$$

where $\bar{r}$ and σ_r are the mean and standard deviation of group rewards. This group-wise normalization highlights informative variations while suppressing noise, making GRPO widely adopted in practice.

Despite these benefits, GRPO fundamentally relies on a small and fixed n per prompt, creating a vulnerability to signal collapse. When all n samples for a prompt yield identical rewards, either all correct or all incorrect, the group mean $\bar{r}$ equals each reward r_i, producing zero advantages ($A_{\text{GRPO}} = 0$) and thus zero gradient. Such uniform-reward cases are frequent: early in training when the model fails on all attempts for hard prompts, and later when it consistently solves easy ones. Empirically, even with $n = 32$, more than half of prompts fall into this "zero-gradient" regime as models improve (Yu et al., 2025).

Crucially, this collapse is not due to the prompts being inherently trivial or impossible, but a **statistical artifact** of undersampling. Training prompts are typically filtered to ensure moderate difficulty (e.g., Yang et al. (2024) retain prompts where 2–5 out of 8 responses are correct), meaning most prompts have non-trivial success probabilities strictly between 0 and 1. With small n, however, random fluctuations make it likely to observe all-correct or all-incorrect groups, thereby masking the true learning signal. Larger n reliably recovers these signals, but at unsustainable inference cost (Figure 2[1]).

To resolve this trade-off, we propose REINFORCE-ADA, an adaptive sampling framework that dynamically allocates inference budget across prompts. Instead of uniformly fixing n, responses of each prompt are sampled in multiple rounds and prompts are deactivated once sufficient signals are collected. This approach ensures that difficult prompts receive more rollouts while easy or saturated prompts terminate early, avoiding wasted computation. Our framework consists of the following key components:

1. **Adaptive Sampling**: Prompts are sampled in multiple rounds, and a successive elimination mechanism deactivates them once enough training signals are observed.

[1]It shows results on the Open-R1 subset for Qwen2.5-Math-1.5B and an RL-trained checkpoint (at step 400). The base model has a modest pass@1 of 26.5%, but pass@256 reaches 81.3%, revealing its latent ability to solve most prompts. Similarly, the trained model exhibits 35.3% all-correct groups at $n = 4$, but only 10.2% at $n = 256$. These results demonstrate that the missing signal is often recoverable with larger n, confirming that uniform-reward collapse is a sampling artifact rather than a model limitation.

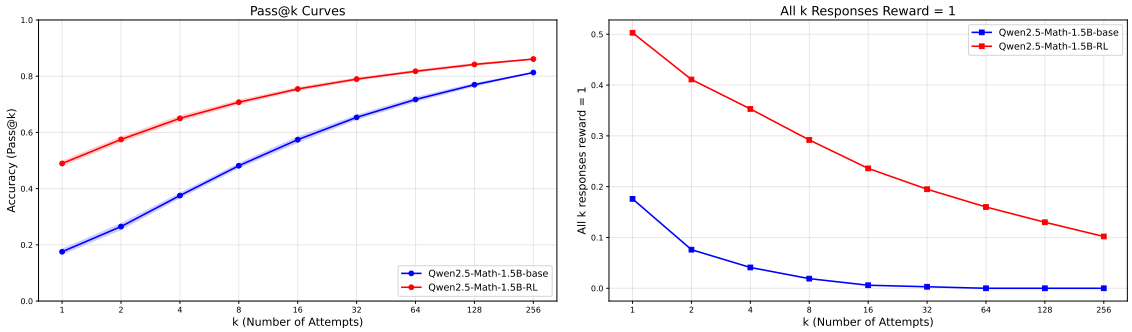

Figure 2: Pass@k curves (left) and the ratio of prompts with all-correct responses (right) for two models on a subset of the Open-R1 prompt set. The models tested are the Qwen2.5-Math-1.5B base model and an intermediate checkpoint from its RL training. The percentage of prompts yielding all-correct/all-incorrect responses is high for small k but drops significantly as k increases. This suggests that signal loss is often a statistical artifact of small sample groups.

2. **Exit Conditions**: We design principled rules (REINFORCE-ADA-POS, REINFORCE-ADA-BALANCE) that determine when prompts deactivate, trade-off between sampling efficiency and signal diversity.

3. **Global Normalization**: Within each prompt, the advantages are normalized using statistics computed from all sampled responses, instead of only the retained subset. This allows for a more robust and unbiased estimation of contribution for each response based on the prompt's pass rate.

REINFORCE-ADA is a plug-and-play replacement for the generation step in standard RL pipelines, requiring no architectural modifications. Across multiple LLMs and benchmarks, it consistently improves signal quality and sample efficiency, achieving the benefits of large-n training at a fraction of its cost.

2 Reinforce-Ada: Reinforce with Adaptive Sampling

2.1 Prior Approach: Passive Filtering and Large Group Size

Prior work has observed this issue and proposed *passively filtering-out* groups with uniform rewards (Yu et al., 2025; Xiong et al., 2025). While this prevents wasted gradient computations, it still incurs the significant upfront cost of generating responses that are ultimately discarded. Moreover, if a number of problems, especially the difficult ones, are discarded, the response signals from those prompts will remain unseen by the model, thus limiting the training improvement.

With these observations, one natural idea is to use a large n for data collection to get more learning signals in both early and later training stage. Indeed, concurrent work by Hu et al. (2025) validates this intuition, showing that increasing n up to 512 increases the ratio of prompts with valid signals and improves final model performance. However, the practicality of this approach is limited by its prohibitive computational cost, as generating hundreds of samples for every prompt is usually infeasible at scale. Moreover, as demonstrated by the seminal DeepSeek-R1 project (DeepSeek-AI et al., 2025), we can use only $n = 16$ responses per prompt to get an effective gradient for model updates. There appears to be a significant gap between the **inference budget** needed to reliably find a useful learning signal and the **update budget** (i.e., the number of samples) required for an effective gradient step.

Our work is designed to bridge this gap. We propose an adaptive sampling framework that smartly allocates a larger inference budget only to the prompts that need it, thereby discovering robust learning signals efficiently without the waste of the uniform, large-n approach.

Algorithm 1 Reinforce-Ada: Reinforce with Adaptive Sampling (One Training Iteration)

Input: Current policy π_θ, batch of prompts $\mathcal{D}$, effective group size for update n, number of sampling rounds N, samples per round $M \geq n$, and exit condition function ExitCondition$(\cdot)$
PHASE 1: ADAPTIVE SAMPLING DATA COLLECTION
Set all prompts x as active and initialize response set $\mathcal{S}_x \leftarrow \emptyset$
for $t = 1, \ldots, N$ **do** ▷ Iterate through sampling rounds
  for each prompt $x \in \mathcal{D}$ where active(x) is true **do**
    Sample M responses $\{a_j, r_j\}_{j=1}^{M} \sim \pi_\theta(\cdot|x), r^\star(\cdot,\cdot)$
    Add to collection: $\mathcal{S}_x \leftarrow \mathcal{S}_x \cup \{a_j, r_j\}_{j=1}^{M}$
    if ExitCondition$(\mathcal{S}_x)$ is met **then**
      Mark prompt as inactive: active$(x) \leftarrow$ false
    end if
  end for
end for
PHASE 2: TRAINING BATCH AND OBJECTIVE CONSTRUCTION
Initialize an empty set for the final training data: $\mathcal{B} \leftarrow \emptyset$
for each prompt $x \in \mathcal{D}$ **do** ▷ Use all collected samples ("global statistics") for normalization
  Let $\mathcal{S}_x = \{(a_j, r_j)\}_{j=1}^{|\mathcal{S}_x|}$ be the full set of collected samples for prompt x
  Compute global mean: $\bar{r}_x \leftarrow \frac{1}{|\mathcal{S}_x|} \sum_{j=1}^{|\mathcal{S}_x|} r_j$
  Form update group by downsampling $\mathcal{S}_x$ to size n, trying to ensure $\geq n/2$ size for each correct or incorrect subset (fill from the other if needed). ▷ Downsample to create the effective group
  Compute advantage for i-th response of prompt x as $A_i \leftarrow r_i - \bar{r}_x$.
end for
The policy gradient objective is then computed using the batch $\mathcal{B}$ as in Equation (3).

2.2 Adaptive Sampling for Group Construction

To resolve the trade-off between sampling cost and signal stability, we propose a simple adaptive sampling strategy that dynamically allocates the inference budget to prompts where it is most needed. This is related to prior work on budget allocation for rejection sampling fine-tuning (Yao et al., 2025; Dong et al., 2023), which typically follows an "explore-then-exploit" paradigm. In those methods, pass rates ($\hat{p}_i$) are first estimated periodically using a small portion of compute budget (in an offline manner); then, an online sampling budget is allocated proportionally to $1/\sqrt{\hat{p}_i}$. This two-stage process, however, is inefficient as it discards the samples used for estimation, is difficult to adapt to a fully online setting, and still suffers from high estimation error due to the limited number of samples used for the initial estimation.

To overcome these limitations, we propose *Adaptive Sampling*, an online algorithm inspired by *successive elimination methods* in the multi-armed bandit literature (Slivkins et al., 2019). Instead of separating estimation from decision-making phases (sampling), we integrate them into a unified online process. The algorithm operates over a batch of prompts in N rounds:

1. **Initialization:** All prompts in the current batch begin in an *active set* (active arms).
2. **Iterative Sampling:** In each round, we generate M new responses for every prompt currently in the *active set*.
3. **Elimination:** At the end of each round, we evaluate each active prompt against a predefined exit condition. If a prompt meets the condition, it is considered "resolved" and removed from the active set for all subsequent rounds.[2]

[2]We also experimented with a more complex variant that estimates pass rates and allocates budgets proportionally within each round. However, this did not yield clear performance gains, so we uses this simpler, easy-to-implement version.

If a prompt remains active after all rounds are completed, we consider it too difficult for the model to resolve within the given budget and revert to a passive filtering strategy.

Training batch construction. After the adaptive collection process, each prompt has an associated pool of responses of varying size. A naive approach would be to use all collected responses. However, in this case, prompts at both ends of the difficulty spectrum can dominate the training batch. On one hand, easy prompts (high pass rate) quickly accumulate a large number of successful responses. Meanwhile, difficult prompts might require extensive sampling to find correct responses, resulting in a large pool of negative samples. Including all these failures could lead to training instability, as the gradient updates would be overwhelmed by "unlearning" signals (Xiong et al., 2025).

To normalize each prompt's influence and ensure signal quality, we downsample each prompt to a fixed-size group of n responses from its collected pool. Moreover, to ensure a balanced signal, we try to draw $n/2$ samples from the set of correct responses and $n/2$ from the incorrect ones. This balanced sampling strategy is an effective technique used in several recent works (Xu et al., 2025; Ye et al., 2025). If one set is smaller than $n/2$, the remaining slots are filled from the other set. This balanced construction ensures that every group has a mix of outcomes, thereby guaranteeing a non-zero reward variance ($\sigma_r > 0$) and a meaningful gradient for every prompt in the training batch.

Exit condition. The elimination rule plays a central role in shaping the algorithm's behavior. We consider two primary exit conditions:

- **Positive-focused** (REINFORCE-ADA-POS): A prompt is deactivated until we collect at least one correct response.
- **Balanced** (REINFORCE-ADA-BALANCE): A prompt is deactivated until a minimum of $n/2$ correct **and** $n/2$ incorrect responses have been collected.

The REINFORCE-ADA-POS condition is motivated by variance reduction principles in rejection sampling fine-tuning (Yao et al., 2025), emphasizing the efficient accumulation of positive examples. In contrast, REINFORCE-ADA-BALANCE enforces the collection of both successes and failures, ensuring a more diverse set of training signals before a prompt is deactivated.

2.3 Advantage Calculation with a Simplified Global Baseline

With adaptive sampling, we naturally accumulate a larger and more varied set of responses for each prompt compared to uniform sampling. Instead of calculating the normalization statistics on the final selected small set, we compute the baseline mean ($\bar{r}$) and standard deviation (σ_r) using the **entire pool of responses** generated during the adaptive collection phase. This allows for a more robust estimation of a prompt's statistics. We term this approach as "using global statistics".

However, the role of the standard deviation (σ_r) as a normalization term becomes more complex with large sample sizes. For instance, for a single prompt with up to 512 responses, the scaling factor $1/(\sigma_r + \varepsilon) \approx 22.6$. The amplified advantage can potentially destabilizing the training process. Given this risk, we remove the standard deviation term and return to the simpler Reinforce with baseline paradigm: $A(x, a_i) = r_i - \bar{r}$. In our experiments, even with moderate inference budget (e.g., for up to 32 responses per prompt), we found that while including the σ_r term had a slight effect on the training dynamics, the final performance of the converged models was nearly identical. This observation is consistent with recent findings suggesting that a simple mean-reward baseline is often sufficient and (Liu et al., 2025; Xiong et al., 2025; Chu et al., 2025). We also aggregate the loss by averaging over all valid tokens across the batch so that all tokens share the same weight regardless of their sequence length (Yu et al., 2025; Liu et al., 2025). In particular, removing the standard deviation and length normalization in GRPO reduces the gradient to the standard Reinforce with baseline.

Finally, we employ the importance sampling correction and clipping technique from the seminal PPO work (Schulman et al., 2017). To improve data efficiency, we typically perform multiple gradient updates using a batch of samples collected by a fixed, older policy, denoted as $\pi_{\theta_{old}}$. Since the current policy π_θ differs from the sampling policy, we need to correct the distribution by importance sampling. Meanwhile, to prevent excessively large updates, we clip the gradient when the importance sampling ratio $\rho_{i,t} = \frac{\pi_\theta(a_{i,t}|x)}{\pi_{\theta_{\text{old}}}(a_{i,t}|x)}$ exceeds thresholds, where $a_{i,t}$ is the t-th token of responses a_i. We notice that the trajectory-level importance sampling suggested by recent work (Zheng et al., 2025a) can also be combined with the proposed adaptive sampling, which we leave for future work. The final objective becomes:

$$\mathcal{L}(\theta) = \frac{1}{|\mathcal{B}|} \sum_{(x,a_i)\in\mathcal{B}} \sum_{t=1}^{|a_i|} \min\left(\rho_{i,t} \cdot A(x,a_i), \text{clip}\left(\rho_{i,t}, 1-\epsilon_{low}, 1+\epsilon_{high}\right) \cdot A(x,a_i)\right). \tag{3}$$

3 Experiment

Models, Data, and Verifier. To evaluate the generality of our approach, we experiment with several foundation models: Qwen2.5-Math-7B, Qwen2.5-Math-1.5B, Qwen3-4B-it, and Llama-3.2-3B-it. For training, we adopt the default subset of the OpenR1-Math-220k dataset[3], which has been shown to be particularly effective in prior work. Solution correctness is assessed automatically using the Math-Verify tool[4] (Kydlíček), which serves as our verifier throughout all experiments.

We have a standard preprocessing pipeline for all experiments. We first deduplicate the prompts, then discard any prompt for which both the reference solutions and the annotated standard solution fail verification by Math-Verify. Finally, following Yang et al. (2024), we focus on problems of moderate difficulty by sampling 16 responses for each remaining prompt and removing those that yield no correct solutions.

RL Training Details. All experiments are conducted based on the `verl` framework (Sheng et al., 2024), where a Tinker-based[5] implementation is also provided. Training is configured with a prompt batch size of 512. The (effective) group size is set to be $n = 4$ for both GRPO and REINFORCE-ADA, where adaptive sampling can sample up to 32 responses for each prompt. We use the AdamW optimizer with a fixed learning rate of 1×10^{-6}. To encourage exploration (with 10-step warm-up), an entropy regularization term with coefficient 1×10^{-4} is applied, while no KL penalty is introduced. Following Yu et al. (2025), we adopt the clip-higher trick by setting the clipping range to $[0.2, 0.28]$. Each RL run is carried out for 600 training steps to obtain the final model.

Evaluation. We evaluate the mathematical reasoning capabilities of the trained models on standard benchmarks including MATH500 (Hendrycks et al., 2021), Minerva Math (Lewkowycz et al., 2022), and OlympiadBench (He et al., 2024). Additionally, we compile a challenging new AIME-like test set of 230 problems by aggregating several smaller, popular competition datasets: AIME24, AIME25, HMMT24, HMMT25, BRUMO25, AMC23, and CMIMC25, sourced from MathArena (Balunović et al., 2025). For all evaluations, we report the Ave@32 metric, where we generate 32 responses per problem using a temperature of 1.0 and a maximum token limit of 4096 and compute the average accuracy.

3.1 Main Results

Adaptive sampling improves training efficiency and final accuracy. Figure 3 compares GRPO (with a uniform sampling strategy) with our REINFORCE-ADA variants across different starting foundation models. In all cases, REINFORCE-ADA climbs faster in the early phases and maintains a higher reward

[3] `https://huggingface.co/datasets/open-r1/OpenR1-Math-220k`
[4] `https://github.com/huggingface/Math-Verify`
[5] `https://github.com/thinking-machines-lab/tinker-cookbook`

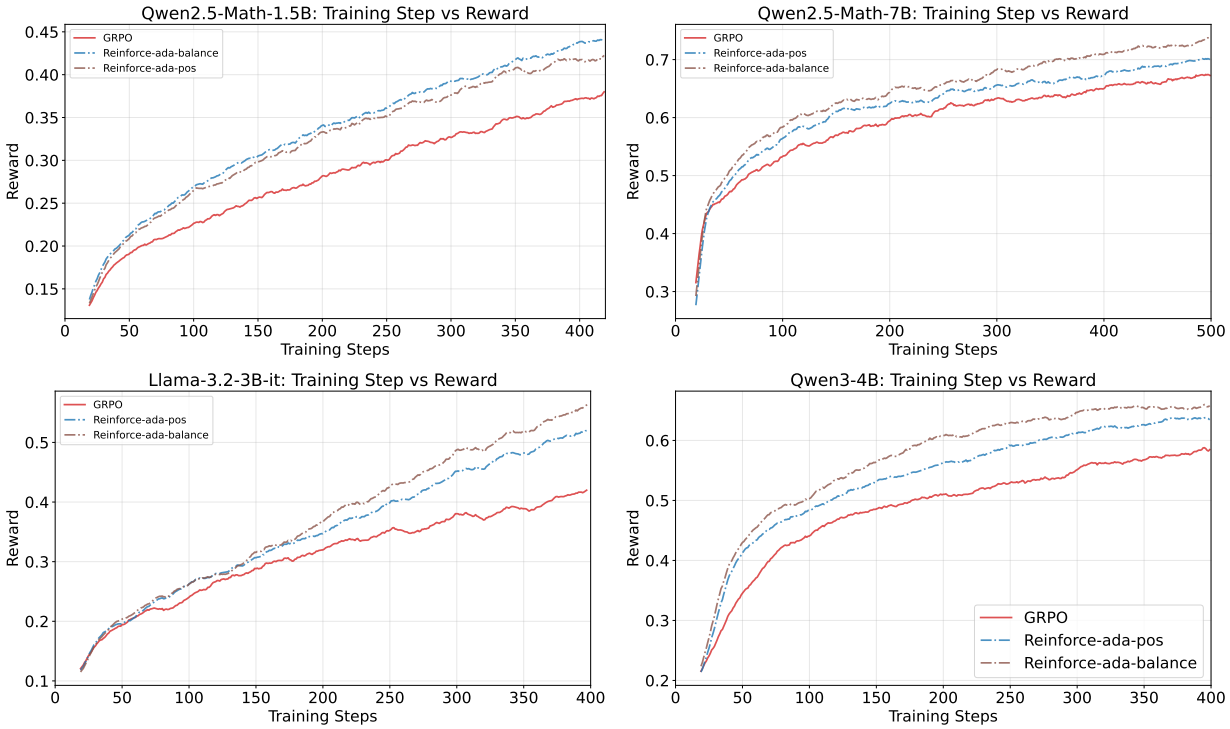

Figure 3: Training reward vs. steps for GRPO and REINFORCE-ADA across backbones: Qwen2.5-Math-1.5B, Qwen2.5-Math-7B, and Llama-3.2-3B-it, Qwen3-4B. Curves are smoothed with a 20-step moving average. In all cases, REINFORCE-ADA learns faster and reaches a higher reward than GRPO, with the BALANCE variant typically achieving the highest asymptote.

throughout training, indicating better sample efficiency per update step. The gap is visible from the first 50-150 steps and persists until convergence, with the balanced version typically achieving the highest reward.

These training gains also translate to stronger held-out results (Table 1). REINFORCE-ADA delivers consistent gains over uniform GRPO, typically increasing 1-3 Avg@32 points. Improvements appear across all suites (Math500, Minerva, Olympiad, AIME-like), indicating robustness rather than benchmark-specific tuning. In comparison, we notice that the adaptive sampling is more beneficial on the challenging training prompt set (denoted by "hard"). We defer a more detailed analysis of the dynamic to Section 3.2. Between variants, REINFORCE-ADA-BALANCE is the most reliable, offering steadier uplifts by preserving exploration and assigning more budgets for signal-poor prompts. Notably, these gains come without extra hyperparameter tuning, but with only one-line replacement of the generation process as illustrated in Figure 1.

The Benefit of Balanced Sampling. REINFORCE-ADA-BALANCE consistently outperforms the positive-only variant, and the gap widens later in training. As the policy improves, uniform-reward groups (especially all-correct) become common (Fig. 2), which starves REINFORCE-ADA-POS of gradient signal. By requiring both successes *and* failures before deactivating a prompt, REINFORCE-ADA-BALANCE keeps these prompts active, mines informative hard negatives, and sustains exploration—yielding more stable advantages, slower entropy collapse, and higher final accuracy.

Reward-Entropy Curve and Pass@K evaluation. A known challenge in evaluating the reasoning ability of LLMs is on the trade-off between reward (accuracy) and entropy (generation diversity). Previous work has argued that post-training can trade this uncertainty for higher rewards in a predictable manner

Model	Algorithm	Math500	Minerva Math	Olympiad Bench	AIME-like	Weighted Average
Qwen2.5-Math-1.5B	GRPO	74.2	34.4	38.4	16.2	45.3
	REINFORCE-ADA-POS	75.8	35.7	38.6	16.5	46.1
	REINFORCE-ADA-BALANCE	77.4	36.5	40.5	17.5	**47.6 (+2.3)**
Qwen2.5-Math-1.5B (hard)	GRPO	71.0	31.8	34.3	13.8	41.9
	REINFORCE-ADA-POS	73.9	33.1	36.4	16.4	44.6
	REINFORCE-ADA-BALANCE	74.7	33.7	38.7	17.6	**45.5 (+3.6)**
Qwen2.5-Math-7B	GRPO	82.2	44.7	45.6	23.2	53.3
	REINFORCE-ADA-POS	82.7	45.1	46.7	23.7	54.2
	REINFORCE-ADA-BALANCE	84.0	45.2	47.1	23.7	**54.6 (+1.3)**
Qwen2.5-Math-7B (hard)	GRPO	80.7	42.8	42.9	21.8	51.3
	REINFORCE-ADA-POS	82.4	43.1	45.0	22.2	52.8
	REINFORCE-ADA-BALANCE	83.1	43.4	46.4	24.9	**53.9 (+2.6)**
Llama-3.2-3B-instruct	GRPO	51.7	20.5	20.4	7.2	27.9
	REINFORCE-ADA-POS	52.6	22.2	21.0	7.5	28.8
	REINFORCE-ADA-BALANCE	53.2	22.4	21.2	8.0	**29.1 (+1.2)**
Qwen3-4B-instruct	GRPO	90.4	51.2	64.9	38.5	66.5
	REINFORCE-ADA-POS	91.6	50.4	66.3	38.8	67.4
	REINFORCE-ADA-BALANCE	91.7	53.0	65.7	38.8	**67.6 (+1.1)**

Table 1: Performance comparison of GRPO and REINFORCE-ADA. We report average@32 accuracy with a sampling temperature of 1.0 and a maximum generation length of 4096 tokens. The weighted average score is computed according to the number of prompts in each benchmark. "Hard" indicates training on a more challenging prompt set, with details provided in Section 3.2.

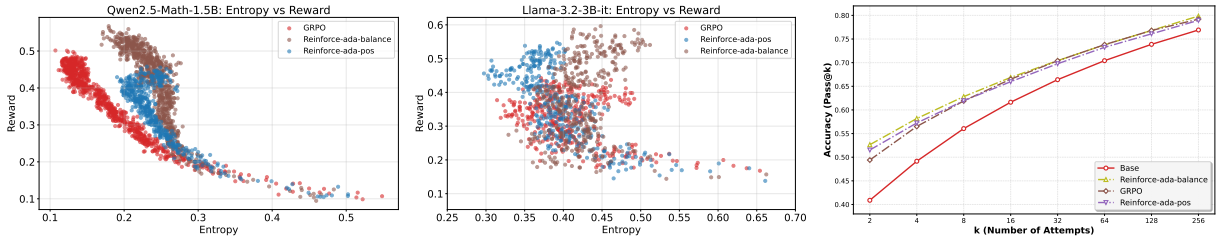

Figure 4: Reward–entropy trade-off (left and mid) and Pass@k (right) on the test benchmarks. REINFORCE-ADA shifts the frontier outward: higher reward at fixed entropy and higher entropy at fixed reward and converts this into stronger Pass@k at small, practical budgets ($k \leq 8$), with REINFORCE-ADA-BALANCE typically best.

(Cui et al., 2025). An ideal algorithm should improve rewards without causing the policy to become overly deterministic (entropy collapse). To provide a more comprehensive evaluation, we analyze this trade-off in this part and present the results in Figure 4.

(i) **Reward-entropy frontier.** While the precise entropy dynamics can vary depending on the foundation model, REINFORCE-ADA consistently achieves a comparable (Llama-3.2-3B-it, Qwen2.5-Math-7B) or superior (Qwen2.5-Math-1.5B, Qwen3-4B-it) reward-entropy curve. Specifically, on Qwen2.5-Math-1.5B (left), GRPO concentrates mass early (low entropy, narrow cloud) and achieves lower reward for a given entropy. Both REINFORCE-ADA variants shift the frontier outward and REINFORCE-ADA-BALANCE lies furthest, at equal reward it sustains higher entropy, and at equal entropy it achieves higher reward. On Llama-3.2-3B-it (mid), the base policy starts with a higher entropy floor, so the separation among different methods is smaller. But the REINFORCE-ADA variants still achieve a competitive frontier than GRPO. We also observe that REINFORCE-ADA-BALANCE > REINFORCE-ADA-POS. We attribute this to the fact that exposure to negative signals discourages the model from becoming overconfident in a single solution path, thus preserving valuable policy diversity.

(ii) **Pass@k behavior.** We also observe that the moderate policy entropy typically converts to an improved pass@k compared to the base model for a wide range of k. Specifically, bottom panel shows that all RL methods dominate the base across k, but the largest, most practical gains appear at small budgets ($k \leq 8$). In this regime, REINFORCE-ADA yields the highest Pass@k; the advantage narrows as k grows (diminishing-returns regime), where REINFORCE-ADA-BALANCE typically remains marginally best. The pattern implies two complementary effects: (a) improved top-1 quality (higher reward at given entropy), and (b) retained diversity among high-scoring modes (shallower saturation with k). Together, adaptive sampling moves the reward–entropy curve outward and converts that shift into higher Pass@k at realistic attempt budgets, with REINFORCE-ADA-BALANCE offering the most stable trade-off.

3.2 Training Dynamic and Ablation Studies

Computational Overhead and dynamic of sampling. Notably, adaptive sampling improves signal quality at a higher wall-clock cost per update. Using `verl` and 8×NVIDIA H100, the average step times are summarized in Table 2. REINFORCE-ADA-POS typically accelerates slightly in later training steps, whereas REINFORCE-ADA-BALANCE remains roughly flat, reflecting its stricter stop rule. These numbers complement the accuracy results in Table 1, showing the trade-off between computational overhead and performance gain.

Model	Algorithm	Avg. Step Time (s)	Relative Cost
Qwen2.5-Math-1.5B	GRPO	102	1.0×
	REINFORCE-ADA-POS	228	2.2×
	REINFORCE-ADA-BALANCE	290	2.8×
Qwen2.5-Math-7B	GRPO	236	1.0×
	REINFORCE-ADA-POS	333	1.41×
	REINFORCE-ADA-BALANCE	375	1.59×

Table 2: Average step time (wall-clock seconds per update) of GRPO vs. REINFORCE-ADA on 8×H100. Relative cost is normalized against GRPO for the same model.

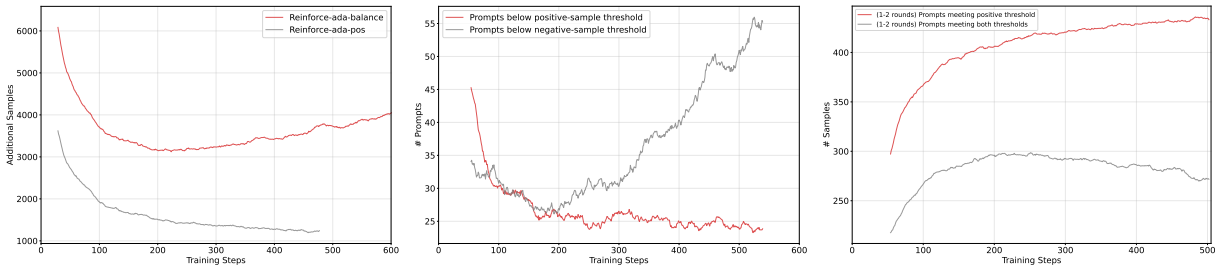

Figure 5: Sampling dynamics with the Qwen2.5-Math-1.5B model. Left: additional samples generated in later rounds compared to standard GRPO. Middle: number of prompts that remain active after multi-round adaptive sampling with the REINFORCE-ADA-BALANCE variant. Right: number of prompts that satisfy the stopping criteria within the first two rounds with the REINFORCE-ADA-BALANCE variant. All curves are smoothed using a moving average with a window size of 20.

Figure 5 provides more detailed information about the behavior

- Left: total extra samples/iteration are always higher for REINFORCE-ADA-BALANCE than REINFORCE-ADA-POS. For REINFORCE-ADA-POS, overhead monotonically declines as prompts quickly satisfy the positive quota and deactivate. For REINFORCE-ADA-BALANCE, overhead drops

early but rises after around 300 steps: as the model improves, correct responses become common while negatives grow scarce, so meeting the negative quota requires additional rounds.

- Middle: for REINFORCE-ADA-BALANCE variant, the count of prompts still active after all rounds shifts from "missing positives" early to "missing negatives" late, confirming the bottleneck change.
- Right: within the first two rounds, many prompts fail the positive quota early (<50 steps), making extra sampling crucial; after ∼200 steps most prompts clear the positive quota quickly, yet fewer meet both quotas without extra budget because negatives are harder to obtain.

In short, REINFORCE-ADA-BALANCE spends more inference to preserve exploration and avoid signal loss in late training; REINFORCE-ADA-POS is cheaper and speeds up as positives dominate, but is more prone to prematurely deactivating all-correct prompts.

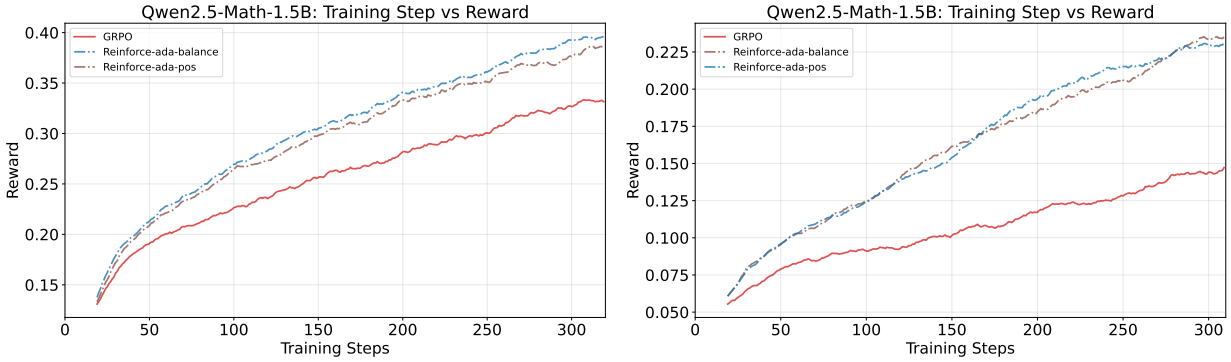

Figure 6: Ablation studies on the prompt set difficulty. Left: prompt set with moderate difficulty. Right: challenging prompt set. The benefit of adaptive sampling is more obvious with challenging prompt set.

The impact of prompt set difficulty. After ∼200 steps (Fig. 5, right), most prompts become easy—two rounds already satisfy the positive quota—so additional sampling brings diminishing returns. To stress-test our method, we construct a hard subset that keeps only prompts with 1-2 correct out of 16 initial samples. As shown in Fig. 6, adaptive sampling yields a much larger margin over GRPO on this challenging set, and the gap widens late in training.

Intuitively, harder prompts reduce all-correct saturation and keep uncertainty high; REINFORCE-ADA continues to mine informative negatives and avoids premature deactivation, sustaining exploration and gradient signal. This suggests that adaptive sampling is most beneficial when (i) the prompt pool is large enough to avoid many-epoch reuse (≈1 epoch per run), and (ii) we couple training with curriculum/difficulty-aware selection to feed consistently challenging items (e.g., online difficulty prediction (Qu et al., 2025; Liao et al., 2025; Shi et al., 2025)).

4 Related Work

Data filtering and selection in online RL training for LLMs. Our work is related to the growing body of literature on data selection and filtering for online reinforcement learning with LLMs, which furthe dates back to the RLHF studies (Zhang et al.; Xiong et al., 2023; Dong et al., 2024; Shi et al., 2024; Feng et al., 2025). In the context of RLVR, some methods employ an oversample-then-downsample strategy: they first generate a large, uniform set of responses for each prompt and then select a subset based on specific criteria. Xu et al. (2025) propose downsampling to maximize reward variance within the group, while Ye et al. (2025) use process rewards to select positive samples, mitigating issues with falsely correct responses.

Xue et al. (2025) study the data filtering in the context of tool-integrated reasoning, where they find that the trajectories with invalid tool calling will significantly hurt the training stability. Our work is mostly related to Yao et al. (2025). Li et al. (2025) propose to use a process search to branch out the trajectories and collect data.

Our work is most closely related to Yao et al. (2025). Yao et al. (2025) frame rejection sampling fine-tuning (RAFT) (Dong et al., 2023) within an Expectation-Maximization (EM) framework (Singh et al., 2023; Zhong et al., 2025). They provide a key theoretical insight that the stochastic gradient estimator in RAFT is from the statistical rejection sampling stage and the optimal budget allocation to minimize stochastic gradient variance is proportional to the prompt's difficulty (pass rate). Their enhanced RAFT and GRPO algorithms, which use this principle, outperform the original versions with fixed group size (Dong et al., 2023; Shao et al., 2024). However, their approach follows a two-stage "explore-then-exploit" paradigm: a small portion of the budget is first used to estimate pass rates (and often in an offline manner), and the remainder is allocated based on these estimates. This process is inefficient as it discards the initial estimation samples, is challenging to implement in a truly online fashion, and the initial estimates can still suffer from high variance. In contrast, our adaptive sampling framework unifies estimation and exploitation into a single, online process. Inspired by successive elimination algorithms from the multi-armed bandit literature (Slivkins et al., 2019), it dynamically allocates the inference budget on the fly. Furthermore, our method supports more general exit conditions, considering both positive and negative signals.

Addressing signal loss in GRPO. A central challenge in applying GRPO is the "signal loss" problem, where groups of responses with uniform rewards yield zero gradients. Prior work has identified this issue and proposed several solutions. The most direct approach is passively filtering out these prompts (Yu et al., 2025; Xiong et al., 2025). Another line of works proposes to modify the advantage computation and avoid zero gradient. To avoid discarding samples, Nan et al. (2025) propose augmenting the reward group with a constant (the maximum possible reward), ensuring the variance is not zero by introducing a bias. Le et al. (2025) propose to assign advantages based on entropy information for prompts with uniform rewards. Finally, some works propose to mitigate this issue by selectively choose the prompt batch. For instance, Qu et al. (2025) employ a Bayesian framework to predict a prompt's pass rate and selectively sample informative prompts during online training. Our work differs by tackling the problem at the collection stage itself, ensuring a sufficient signal is gathered adaptively rather than correcting for its absence afterward. Similarly, Shi et al. (2025) propose to use an adaptive curriculum learning to select prompts with suitable difficulty during the online RL training. Zhang et al. (2025) also develop a curriculum learning methods to select training prompts of intermediate difficulty in a two-stage manner. Zheng et al. (2025b) use a dictionary-based approach to record historical reward from the last epoch and skip uninformative prompts.

GRPO Variant Designs. Our work is orthogonal to another line of research focusing on modifying the policy gradient algorithm itself, such as innovations in advantage estimation and clipping mechanisms (Hu, 2025; Zhu et al., 2025; Zheng et al., 2025a; Huang et al., 2025; Chu et al., 2025). While these methods refine the core update rule, we focus on data generation and construction pipeline. Our adaptive sampling framework is complementary to these algorithmic improvements and can be readily combined with them.

5 Discussion and End Note

In this work, we introduced Reinforce-Ada, an adaptive sampling framework to enhance the stability and efficiency of group-based reinforcement learning for LLMs. Our method dynamically allocates a larger inference budget only to prompts requiring more exploration, efficiently constructing informative training groups and preventing the common signal collapse issue. Designed as a *lightweight, drop-in replacement* for the standard RL training loop, our experiments show that Reinforce-Ada delivers *consistent and robust improvements* across a diverse set of foundation models, accelerating the training reward curve and boosting final performance on held-out test benchmarks. These benefits come with only moderate computational

overhead, offering a more scalable and effective alternative to the brute-force approach of uniformly large group sizes.

We view this work as part of the broader recent effort on **data curation for online reinforcement learning**. Recent studies have explored *macro, prompt-level* strategies, such as curriculum learning (Zhao et al., 2024; Shi et al., 2025; Zhang et al., 2025), to shape the distribution of training data during online learning. In contrast, our contribution operates at the *response-sampling level*, focusing on how to construct effective learning signals within each prompt. However, as discussed in Section 3.2, the relative difficulty of the prompt set evolves alongside model training, and this interplay critically affects both learning dynamics and final performance of our method. Moreover, while our experiments are restricted to the math domain due to resource constraints, real-world post-training systems require data curation as a *holistic challenge* spanning the entire pipeline. We hope that the adaptive sampling framework can serve as an effective building block in this broader ecosystem when combined with complementary approaches from the literature.

References

Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. Matharena: Evaluating llms on uncontaminated math competitions, February 2025. URL `https://matharena.ai/`.

Xiangxiang Chu, Hailang Huang, Xiao Zhang, Fei Wei, and Yong Wang. Gpg: A simple and strong reinforcement learning baseline for model reasoning. *arXiv preprint arXiv:2504.02546*, 2025.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, et al. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*, 2025.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL `https://arxiv.org/abs/2501.12948`.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, KaShun SHUM, and Tong Zhang. RAFT: Reward ranked finetuning for generative foundation model alignment. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=m7p5O7zblY.

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.

Yunzhen Feng, Ariel Kwiatkowski, Kunhao Zheng, Julia Kempe, and Yaqi Duan. Pilaf: Optimal human preference sampling for reward modeling. *arXiv preprint arXiv:2502.04270*, 2025.

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.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

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

Jian Hu, Mingjie Liu, Ximing Lu, Fang Wu, Zaid Harchaoui, Shizhe Diao, Yejin Choi, Pavlo Molchanov, Jun Yang, Jan Kautz, and Yi Dong. Brorl: Scaling reinforcement learning via broadened exploration, 2025. URL https://arxiv.org/abs/2510.01180.

Wenke Huang, Quan Zhang, Yiyang Fang, Jian Liang, Xuankun Rong, Huanjin Yao, Guancheng Wan, Ke Liang, Wenwen He, Mingjun Li, et al. Mapo: Mixed advantage policy optimization. *arXiv preprint arXiv:2509.18849*, 2025.

Hynek Kydlíček. Math-Verify: Math Verification Library. URL https://github.com/huggingface/math-verify.

Thanh-Long V Le, Myeongho Jeon, Kim Vu, Viet Lai, and Eunho Yang. No prompt left behind: Exploiting zero-variance prompts in llm reinforcement learning via entropy-guided advantage shaping. *arXiv preprint arXiv:2509.21880*, 2025.

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.

Yizhi Li, Qingshui Gu, Zhoufutu Wen, Ziniu Li, Tianshun Xing, Shuyue Guo, Tianyu Zheng, Xin Zhou, Xingwei Qu, Wangchunshu Zhou, et al. Treepo: Bridging the gap of policy optimization and efficacy and inference efficiency with heuristic tree-based modeling. *arXiv preprint arXiv:2508.17445*, 2025.

Baohao Liao, Yuhui Xu, Hanze Dong, Junnan Li, Christof Monz, Silvio Savarese, Doyen Sahoo, and Caiming Xiong. Reward-guided speculative decoding for efficient llm reasoning. *arXiv preprint arXiv:2501.19324*, 2025.

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.

Gongrui Nan, Siye Chen, Jing Huang, Mengyu Lu, Dexun Wang, Chunmei Xie, Weiqi Xiong, Xianzhou Zeng, Qixuan Zhou, Yadong Li, et al. Ngrpo: Negative-enhanced group relative policy optimization. *arXiv preprint arXiv:2509.18851*, 2025.

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.

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

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, YK Li, Y Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

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.

Ruizhe Shi, Runlong Zhou, and Simon S Du. The crucial role of samplers in online direct preference optimization. *arXiv preprint arXiv:2409.19605*, 2024.

Taiwei Shi, Yiyang Wu, Linxin Song, Tianyi Zhou, and Jieyu Zhao. Efficient reinforcement finetuning via adaptive curriculum learning. *arXiv preprint arXiv:2504.05520*, 2025.

Avi Singh, John D Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Peter J Liu, James Harrison, Jaehoon Lee, Kelvin Xu, Aaron Parisi, et al. Beyond human data: Scaling self-training for problem-solving with language models. *arXiv preprint arXiv:2312.06585*, 2023.

Aleksandrs Slivkins et al. Introduction to multi-armed bandits. *Foundations and Trends® in Machine Learning*, 12(1-2):1–286, 2019.

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. Iterative preference learning from human feedback: Bridging theory and practice for rlhf under kl-constraint. 2023.

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

Yixuan Even Xu, Yash Savani, Fei Fang, and Zico Kolter. Not all rollouts are useful: Down-sampling rollouts in llm reinforcement learning. *arXiv preprint arXiv:2504.13818*, 2025.

Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru Li, Xiaosen Zheng, Zejun Ma, and Bo An. Simpletir: End-to-end reinforcement learning for multi-turn tool-integrated reasoning. *arXiv preprint arXiv:2509.02479*, 2025.

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

Jiarui Yao, Yifan Hao, Hanning Zhang, Hanze Dong, Wei Xiong, Nan Jiang, and Tong Zhang. Optimizing chain-of-thought reasoners via gradient variance minimization in rejection sampling and rl. *arXiv preprint arXiv:2505.02391*, 2025.

Chenlu Ye, Zhou Yu, Ziji Zhang, Hao Chen, Narayanan Sadagopan, Jing Huang, Tong Zhang, and Anurag Beniwal. Beyond correctness: Harmonizing process and outcome rewards through rl training. *arXiv preprint arXiv:2509.03403*, 2025.

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.

Chuheng Zhang, Wei Shen, Li Zhao, Xuyun Zhang, Lianyong Qi, Wanchun Dou, and Jiang Bian. Policy filtration in rlhf to fine-tune llm for code generation.

Ruiqi Zhang, Daman Arora, Song Mei, and Andrea Zanette. Speed-rl: Faster training of reasoning models via online curriculum learning. *arXiv preprint arXiv:2506.09016*, 2025.

Zirui Zhao, Hanze Dong, Amrita Saha, Caiming Xiong, and Doyen Sahoo. Automatic curriculum expert iteration for reliable llm reasoning. *arXiv preprint arXiv:2410.07627*, 2024.

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*, 2025a.

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*, 2025b.

Han Zhong, Yutong Yin, Shenao Zhang, Xiaojun Xu, Yuanxin Liu, Yifei Zuo, Zhihan Liu, Boyi Liu, Sirui Zheng, Hongyi Guo, et al. Brite: Bootstrapping reinforced thinking process to enhance language model reasoning. *arXiv preprint arXiv:2501.18858*, 2025.

Xinyu Zhu, Mengzhou Xia, Zhepei Wei, Wei-Lin Chen, Danqi Chen, and Yu Meng. The surprising effectiveness of negative reinforcement in llm reasoning. *arXiv preprint arXiv:2506.01347*, 2025.

A Authorship and Credit Attribution

All authors provided valuable contributions to this project, each bringing unique expertise and insights that were crucial for its success.

WX proposed the project idea, initiated and organized the project, and developed the core adaptive sampling algorithm; implemented the initial codebase, processed the data, and ran proof-of-concept experiments to validate its effectiveness; drafted the initial version of the paper, with subsequent contributions and revisions from co-authors.

CY proposed the idea of global normalization; jointly developed the core adaptive sampling algorithms, including the codes for Reinforce-Ada-pos, Reinforce-Ada-balance, global normalization, and verification; ran the experiments for Qwen2.5-Math-1.5B and Qwen2.5-Math-7B; contributed to the paper writing.

BL processed the data; ran the experiments for Qwen2.5-Math-7B, Llama, Qwen3-4B; provided the released version of code (including the Tinker version); and contributed to the paper proofreading.

HD initiated, coordinated, and drove the project; provided insights about the algorithm and project design; implemented the production-ready Reinforce-Ada; developed and refactored a scalable and configurable codebase; conducted key experiments for Qwen2.5-Math-1.5B; made substantial contributions to the manuscript writing.

XX, **CM**, **JB**, **NJ**, **TZ** are senior authors, supported and advised the work, provided resources, and suggested experiments and improvements to the writing.