Act Only When It Pays: Efficient Reinforcement Learning for LLM Reasoning via Selective Rollouts

Haizhong Zheng [1] **Yang Zhou** [1] **Brian R. Bartoldson** [2] **Bhavya Kailkhura** [2] **Fan Lai** [3] **Jiawei Zhao** [4] **Beidi Chen** [1]

Abstract

Reinforcement learning, such as PPO and GRPO, has powered recent breakthroughs in LLM reasoning. Scaling rollout to sample more prompts enables models to selectively use higher-quality data for training, which can stabilize RL training and improve model performance, but at the cost of significant computational overhead. In this paper, we first show that a substantial portion of this overhead can be avoided by skipping uninformative prompts *before rollout*. Our analysis of reward dynamics reveals a strong temporal consistency in prompt value: prompts that are uninformative in one epoch are likely to remain uninformative in near future epochs. Based on these insights, we propose GRESO (GRPO with Efficient Selective Rollout), an online, lightweight pre-rollout filtering algorithm that predicts and skips uninformative prompts. By evaluating GRESO on a broad range of math benchmarks and models, like Qwen2.5-Math-1.5B/7B and DeepSeek-R1-Distill-Qwen-1.5B, we show that GRESO achieves up to **2.4× wall-clock time speedup** in rollout and up to **2.0× speedup** in total training time without accuracy degradation.

1. Introduction

Recent reasoning models (Jaech et al., 2024; Guo et al., 2025; Team et al., 2025), such as OpenAI's o1 and DeepSeek's R1, leverage Chain-of-Thought as a form of test-time scaling to significantly enhance the reasoning capabilities of large language models (LLMs). Reinforcement Learning (RL) techniques, including PPO (Ouyang et al., 2022) and GRPO (Guo et al., 2025), have emerged as key drivers of this progress. By generating data online during each training iteration (i.e., rollout), RL enables models to iteratively refine their reasoning strategies through self-exploration, often achieving or even surpassing human-level performance (Jaech et al., 2024; Sun et al., 2024; 2023). Notably, *scaling computational resources to sample responses for more prompts* at this rollout stage can further enhance training, which allows models to selectively utilize higher-quality data and thus train models with better converged performance (Yu et al., 2025). However, scaling up rollouts introduces significant computational overhead, as rollout remains a major bottleneck in RL training (Zhong et al., 2025; Noukhovitch et al., 2024; Sheng et al., 2024; von Werra et al., 2020). For instance, as shown in Figure 1, filtering out uninformative examples[1] and resampling to fill the batch with effective data (also known as Dynamic Sampling in (Yu et al., 2025)) can improve model performance, but it comes at the cost of significantly increased rollout overhead. Motivated by this challenge, we aim to investigate the following research question in this paper:

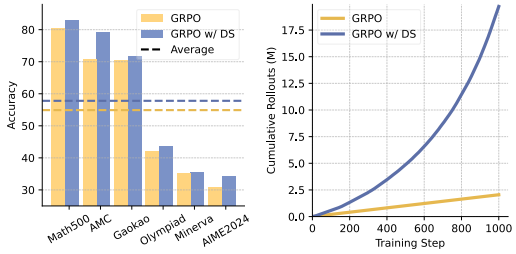


Figure 1: **Left:** GRPO training with more effective data through Dynamic Sampling (DS) leads to improved final model performance. **Right:** However, DS requires additional rollouts to maintain the same training batch size.

> *How can we perform more selective rollouts—focusing on sampling more valuable prompts—to make this scaling more efficient?*

Existing methods face several limitations in addressing this question. First, some approaches (Wang et al., 2025; Li et al., 2025) attempt to improve data efficiency by pruning datasets before training. These methods typically rely on training a model to identify valuable data points; however, there is no conclusive evidence that such strategies improve the overall efficiency of RL training as well. Sec-

[1]Carnegie Mellon University [2]Lawrence Livermore National Laboratory [3]University of Illinois Urbana-Champaign [4]Meta AI.

Accepted at ES-FoMo Workshop@ICML2025.

[1]In GRPO, many examples yield identical rewards across all responses, resulting in zero advantage and thus contributing no learning signal during training.

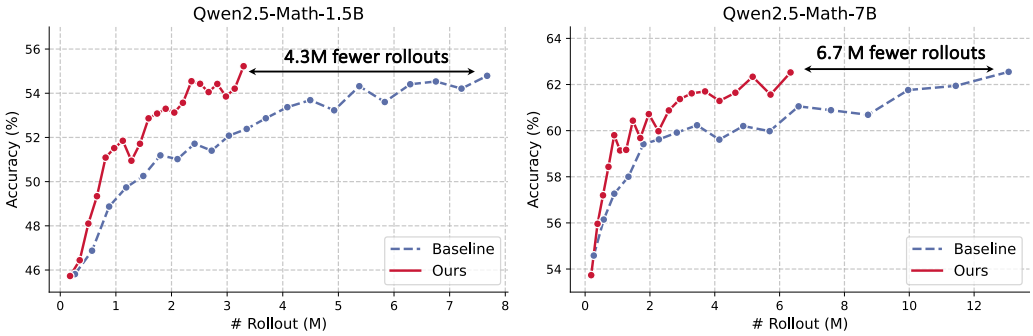


Figure 2: We train Qwen2.5-Math-1.5B/7B on the DAPO + MATH and evaluate them on five math benchmarks: MATH500, AMC, Gaokao, Minerva, and Olympiad. Compared to the baseline method (Dynamic Sampling), our approach (GRESO) reduces rollout overhead by up to 2× while achieving comparable accuracy, improving the efficiency of rollout scaling.

ond, these static pruning methods overlook the fact that the value of a data point can vary across models and training stages, limiting their ability to support adaptive data selection. Finally, online selection approaches such as Dynamic Sampling (Yu et al., 2025) perform oversampling and filter out uninformative data only after rollout, leading to substantial additional rollout cost. Estimating data quality accurately and efficiently *before rollout* remains a challenging and underexplored problem.

Consequently, an ideal selective rollout algorithm for efficient LLM RL should have the following properties: **1) Online data selection.** Instead of relying on an auxiliary model trained offline to pre-prune the dataset, an ideal method should perform data selection online during training. This avoids the additional overhead of training a separate model and enables decisions to be made based on the current training states. **2) Model-based data value estimation.** Data values evolve throughout training and vary across different models, requiring a selective rollout strategy to adapt to different models and training stages. **3) Low computational overhead.** To ensure scalability, the selective rollout strategy should introduce minimal cost during training.

In this paper, we aim to design an efficient selective rollout strategy for LLM RL to make rollout scaling more efficient. We begin by analyzing the training dynamics of prompts across epochs and observe a strong temporal consistency across different training epochs (Section 2). In particular, prompts that yield zero advantage in one epoch are more likely to do so in future epochs as well. This temporal correlation suggests that historical reward dynamics can be leveraged to predict and preemptively skip zero-variance examples before rollout. Building on these observations, we propose **GRESO** (GRPO with Efficient Selective Rollout) in Section 3, an online efficient pre-rollout filtering algorithm that reduces rollout cost by selectively skipping prompts predicted to be zero-variance. Instead of performing filtering after rollout, GRESO estimates a skipping probability for each prompt based on its reward dynamics during training prior to the rollout stage, significantly reducing prompt selection overhead and making rollout scaling more efficient.

In Section 4, we empirically verify the efficiency of GRESO on six math reasoning benchmarks and three models (Qwen2.5-Math-1.5B (Yang et al., 2024), DeepSeek-R1-Distill-Qwen-1.5B (Guo et al., 2025), and Qwen2.5-Math-7B (Yang et al., 2024)). Our evaluation results show that GRESO achieves up to **2.4× speedup** in rollout and **2.0× speedup** in total training time while maintaining comparable accuracy (Section 4.1). We also conduct a more detailed study on how GRESO reduces training overhead by performing selective rollout and ablation study on different components of GRESO in Section D.3.

2. Observation

In this section, we empirically show that a high zero-variance[2] ratio can hurt the training performance. Besides, our analysis reveals a strong temporal consistency in prompt value: prompts that are uninformative in one training epoch tend to remain uninformative in subsequent epochs.

2.1. Reduction of Effective Prompts in GRPO Training

The existence of zero-advantage prompts can largely reduce the ratio of effective prompts in a training batch. During GRPO training on Qwen2.5-Math-7B (Yang et al., 2024), the ratio of effective prompts keeps decreasing as the training proceeds: at the late stage of training, this ratio can be around only 20% (See an example in Figure 4a). A varying ratio of effective prompts can potentially hurt training stability and final model performance (Yu et al., 2025).

A potential way to address this instability issue is to oversample and select a batch only containing effective prompts, which is also known as Dynamic Sampling (DS) (Yu et al., 2025). As shown in Figure 1, GRPO with DS consistently outperforms the vanilla GRPO, particularly on datasets such

[2]We refer to prompts for which all sampled responses receive identical rewards—and thus yield zero reward variance and no learning signal—as *zero-variance prompts*, while those producing non-identical rewards are termed *effective prompts*.

as AMC and AIME24. This performance gain stems from DS's ability to filter out zero-variance prompts, thereby stabilizing training. While DS leads to better performance, it incurs significantly higher computational cost due to its need to oversample more data to maintain the training batch size of effective prompts (as shown in Figure 1). However, a substantial amount of rollout computation is wasted on prompts that ultimately result in zero-variance prompts. Identifying such prompts prior to rollout can significantly reduce computational overhead.

2.2. Temporal Correlation of Examples across Epochs

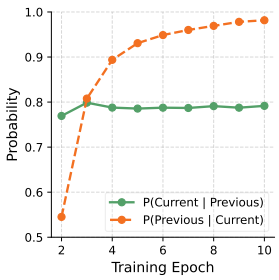


Figure 3: Temporal correlation of prompts across epochs.

Training data typically exhibits strong temporal correlations across epochs (Zheng et al., 2023; Li et al., 2025; Toneva et al., 2018). We hypothesize that zero-variance prompts in GRPO training also have such strong correlations in their training dynamics. To test this hypothesis, we conduct a study on the temporal correlation of zero-variance prompts in GRPO training. Specifically, we train Qwen2.5-Math-7B with GRPO and measure two probabilities: **1)** `P(Previous|Current)`: The probability that a prompt identified as zero-variance in the current epoch was also zero-variance in any previous epoch. **2)** `P(Current|Previous)`: The probability that a prompt identified as zero-variance in any previous epoch remains zero-variance in the current epoch.

The results shown in Figure 3 indicate that zero-variance prompts exhibit strong temporal correlations throughout training. We have two key observations: *1) Prompts previously identified as zero-variance are likely to remain zero-variance.* `P(Previous|Current)` curve shows that the majority of zero-variance prompts in a given epoch (e.g., over 90%) were also identified as zero-variance in earlier epochs. *2) Some zero-variance prompts can become effective again in future epochs.* `P(Current|Previous)` curve shows that approximately 20% of prompts previously labeled as zero-variance become effective prompts that contribute to training again. This suggests that, rather than statically pruning zero-variance prompts, it is beneficial to retain some degree of exploration.

3. Methodology: GRESO

In this section, we present GRESO (GRPO with Efficient Selective Rollout), a novel, online, efficient selective rollout algorithm that predicts and skips zero-variance prompts using reward training dynamics before the rollout stage. The full algorithm is provided in Algorithm 1 in the appendix.

3.1. Probabilistic Pre-rollout Example Filtering

Building on our observation in Section 2.2, we propose to leverage reward training dynamics to detect and filter these prompts *before rollout* to save rollout computation. During training, each prompt x_i is associated with a training dynamics trace: $T_i = (e_{i,1}, R_{i,1}), \ldots, (e_{i,n}, R_{i,n})$, where $e_{i,j}$ denotes the epoch number of the j-th sampling for example x_i, and $R_{i,1} = \{r_{i,1}^{(k)}\}_{k=1}^{G}$ represents the set of response rewards obtained in that epoch. The goal of our algorithm is to predict whether x_i is a zero-variance prompt—i.e., one that yields identical rewards for all responses – based on its reward dynamics T_i prior to rollout.

Probabilistic Filtering. To utilize this reward training dynamics, we propose a *probabilistic filtering strategy*: each prompt is calculated with a filtering probability based on its current training dynamics. As observed in Section 2.2, some zero-variance prompts can become effective again in later epochs. A key advantage of this probabilistic-based approach is that it naturally balances exploitation and exploration, allowing zero-variance prompts to still be occasionally sampled. More specifically, given a prompt x_i whose training dynamics trace is $T_i = (e_{i,1}, R_{i,1}), \ldots, (e_{i,n}, R_{i,n})$, we calculated the filtering probability by:

$$p_f(x_i) = 1 - p_e^{z_i}, \tag{1}$$

$$z_i = \max\left\{ k \in [0, n] \,\middle|\, \prod_{j=n-k+1}^{n} \mathbb{I}_{i,j} = 1 \right\}, \tag{2}$$

$$\mathbb{I}_{i,j} = \begin{cases} 1, & \text{if all rewards in } R_{i,j} \text{ are identical,} \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

where p_e is the base exploration probability controlling how likely a prompt is selected for rollout. z_i represents the number of most recent consecutive rollouts for prompt x_i that were zero-variance.

Self-adjustable Base Exploration Probability. One challenge of the above probabilistic filtering algorithm lies in determining the base exploration probability, which can vary across models, datasets, and even different training stages. In addition, different base probabilities may be appropriate for easy and hard zero-variance prompts. Manually selecting the probabilities for all scenarios is impractical.

To address this challenge, GRESO employs an adaptive algorithm that automatically adjusts the base exploration probability at each training iteration (Lines 16–25 in Algorithm 1). Rather than requiring users to manually select the base probability, which can vary across different settings, GRESO only requires a target zero-variance percentage. It then automatically increases or decreases the exploration rate by a step size Δ_p based on whether the observed zero-variance percentage is above or below the target. (We set

Table 1: Performance (%) comparison across six math reasoning benchmarks. We train three models on DAPO + MATH (DM). Compared to Dynamic Sampling (DS), GRESO achieves similar accuracy while significantly reducing the number of rollouts. The results trained on Open R1 subset (OR1) can be found in Table 2 in the Appendix.

Dataset	Method	Math500	AIME24	AMC	Gaokao	Miner.	Olymp.	Avg.	# Rollout
Qwen2.5-Math-1.5B (Yang et al., 2024)									
DM	DS	77.3	16.7	61.7	64.2	31.8	38.7	48.4	7.6M
	GRESO	76.6	15.0	61.4	66.2	33.3	38.5	<u>48.5</u>	**3.3M**
DeepSeek-R1-Distill-Qwen-1.5B (Guo et al., 2025)									
DM	DS	87.9	36.7	71.7	78.7	35.3	55.9	<u>61.0</u>	2.4M
	GRESO	87.7	36.7	71.1	78.4	33.9	55.1	60.5	**1.6M**
Qwen2.5-Math-7B (Yang et al., 2024)									
DM	DS	82.9	34.2	79.2	71.7	35.4	43.6	<u>57.8</u>	13.1M
	GRESO	82.2	32.5	80.7	70.2	35.4	44.1	57.5	**6.3M**

Δ_p to 1% in all our evaluations.) Additionally, instead of using a single base exploration probability, GRESO maintains two separate values: one for easy zero-variance prompts and another for hard ones. When computing the filtering probability $p_f(x_i)$, GRESO first determines whether x_i is an easy or hard zero-variance prompt and then applies the corresponding exploration probability[3].

Adaptive Sampling Batch Size. In the current design of Dynamic Sampling (Yu et al., 2025), if the number of valid examples is insufficient to meet the training batch size requirement, the training performs rollout using a fixed batch size. However, this may result in wasted computation when only a small number of additional examples are needed to complete the training batch. To further improve rollout efficiency, GRESO adopts an adaptive rollout batch size:

$$B_{\mathrm{r}} = \min(B_{\mathrm{r}}^{\mathrm{default}}, \frac{\beta B_{\Delta}}{(1-\alpha)}), \tag{4}$$

where $B_{\mathrm{r}}^{\mathrm{default}}$ is the default rollout batch size, B_{Δ} is the number of examples needed to fill the training batch, α is the current zero-variance example ratio in this iteration (as some rollouts have already occurred in this iteration), and β is a safety factor, which is fixed at 1.25 across all our evaluations, to ensure sufficient valid examples are collected.

4. Experiment

In this section, we evaluate GRESO on multiple benchmarks using three different models. We evaluate our methods on two training datasets as two settings: DAPO+Math (DM) (Yu et al., 2025; Hendrycks et al., 2021) and OpenR1 Subset (OR1) (Face, 2025). The evaluation results show that GRESO achieves comparable performance to DS while significantly reducing rollout and training costs.

4.1. End-to-end Efficiency Comparison

No performance drop with up to 3.35× fewer rollouts. To verify the effectiveness of GRESO, we present a comprehensive evaluation of GRESO and Dynamic Sampling (DS), which filters out zero-variance examples and resamples to fill the batch with effective data, across six math reasoning benchmarks, using three different model settings in Table 1. The models are trained on either the DAPO + MATH dataset (DM) or the Open R1 subset (OR1). We report both the performance and the number of rollouts from the checkpoint that achieves the best average performance across six benchmarks. Across all training settings, GRESO achieves comparable accuracy as DS, while significantly reducing the number of rollout samples—achieving up to $3.35\times$ fewer rollouts. For example, on Qwen2.5-Math-7B trained on the DM dataset, GRESO achieves a comparable average accuracy to DS (57.5% vs. 57.8%), while reducing the number of rollouts from 13.1M to 6.3M. These results demonstrate that GRESO maintains performance while substantially lowering the cost on rollouts. Similar improvements are observed across other evaluation settings. This rollout saving enables GRESO to achieve up to **2.4× wall-clock speedup** in the rollout stage and up to **2.0× speedup** in total training time, all without compromising accuracy (we present more detailed comparison in Appendix D.2).

5. Conclusion

In this paper, we present GRESO, a selective rollout algorithm for LLM RL. GRESO aims to improve RL training efficiency by selecting effective prompts before rollout to save unnecessary overhead on sampling uninformative prompts. GRESO leverages reward dynamics to efficiently filter out zero-variance prompts before rollout and significantly improve the RL training efficiency.

[3]We set the target zero-variance ratio to 25% for all experiments and allocate it between easy and hard prompts in an $1:2$ ratio (i.e., 8.3% for easy and 16.7% for hard zero-variance prompts), based on the intuition that, as models become more capable during training, more exploration on hard examples can be more beneficial. However, a more optimal allocation scheme may exist, which we leave for our future study.

References

Art of Problem Solving. Aime problems and solutions. https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions, 2024a. Accessed: 2025-04-20.

Art of Problem Solving. Amc problems and solutions. https://artofproblemsolving.com/wiki/index.php?title=AMC_Problems_and_Solutions, 2024b. Accessed: 2025-04-20.

Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., DasSarma, N., Drain, D., Fort, S., Ganguli, D., Henighan, T., et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Chen, L., Li, S., Yan, J., Wang, H., Gunaratna, K., Yadav, V., Tang, Z., Srinivasan, V., Zhou, T., Huang, H., et al. Alpagasus: Training a better alpaca with fewer data. In *The Twelfth International Conference on Learning Representations*, 2024.

Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., and Amodei, D. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Dai, J., Pan, X., Sun, R., Ji, J., Xu, X., Liu, M., Wang, Y., and Yang, Y. Safe rlhf: Safe reinforcement learning from human feedback. *arXiv preprint arXiv:2310.12773*, 2023.

Das, N., Chakraborty, S., Pacchiano, A., and Chowdhury, S. R. Active preference optimization for sample efficient rlhf. *arXiv preprint arXiv:2402.10500*, 2024.

Face, H. Open r1: A fully open reproduction of deepseek-r1, January 2025. URL https://github.com/huggingface/open-r1.

Fatemi, M., Rafiee, B., Tang, M., and Talamadupula, K. Concise reasoning via reinforcement learning. *arXiv preprint arXiv:2504.05185*, 2025.

Gao, J., Xu, S., Ye, W., Liu, W., He, C., Fu, W., Mei, Z., Wang, G., and Wu, Y. On designing effective rl reward at training time for llm reasoning. *arXiv preprint arXiv:2410.15115*, 2024.

Guo, D.-A. T. D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

He, C., Luo, R., Bai, Y., Hu, S., Thai, Z. L., Shen, J., Hu, J., Han, X., Huang, Y., Zhang, Y., et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*, 2024.

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

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

Ivison, H., Smith, N. A., Hajishirzi, H., and Dasigi, P. Data-efficient finetuning using cross-task nearest neighbors. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 9036–9061, 2023.

Ivison, H., Zhang, M., Brahman, F., Koh, P. W., and Dasigi, P. Large-scale data selection for instruction tuning. *arXiv preprint arXiv:2503.01807*, 2025.

Jaech, O. T. A., Kalai, A., Lerer, A., Richardson, A., El-Kishky, A., and Others. Openai o1 system card, 2024. URL https://arxiv.org/abs/2412.16720.

Kazemnejad, A., Aghajohari, M., Portelance, E., Sordoni, A., Reddy, S., Courville, A., and Roux, N. L. Vineppo: Unlocking rl potential for llm reasoning through refined credit assignment. *arXiv preprint arXiv:2410.01679*, 2024.

Kwak, B.-J., Song, N.-O., and Miller, L. E. Performance analysis of exponential backoff. *IEEE/ACM transactions on networking*, 13(2):343–355, 2005.

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J., Zhang, H., and Stoica, I. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, 2023.

Lewkowycz, A., Andreassen, A., Dohan, D., Dyer, E., Michalewski, H., Ramasesh, V., Slone, A., Anil, C., Schlag, I., Gutman-Solo, T., et al. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857, 2022.

Li, X., Zou, H., and Liu, P. Limr: Less is more for rl scaling. *arXiv preprint arXiv:2502.11886*, 2025.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Liu, Z., Kou, B., Li, P., Yan, M., Zhang, J., Huang, F., and Liu, Y. Enabling weak llms to judge response reliability via meta ranking. *arXiv preprint arXiv:2402.12146*, 2024.

Liu, Z., Chen, C., Li, W., Qi, P., Pang, T., Du, C., Lee, W. S., and Lin, M. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Luo, M., Tan, S., Huang, R., Shi, X., Xin, R., Cai, C., Patel, A., Ariyak, A., Wu, Q., Zhang, C., et al. Deepcoder: A fully open-source 14b coder at o3-mini level, 2025.

Muennighoff, N., Yang, Z., Shi, W., Li, X. L., Fei-Fei, L., Hajishirzi, H., Zettlemoyer, L., Liang, P., Candès, E., and Hashimoto, T. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025a.

Muennighoff, N., Yang, Z., Shi, W., Li, X. L., Fei-Fei, L., Hajishirzi, H., Zettlemoyer, L., Liang, P., Candès, E., and Hashimoto, T. s1: Simple test-time scaling, 2025b. URL https://arxiv.org/abs/2501.19393.

Muldrew, W., Hayes, P., Zhang, M., and Barber, D. Active preference learning for large language models. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 36577–36590, 2024.

Noukhovitch, M., Huang, S., Xhonneux, S., Hosseini, A., Agarwal, R., and Courville, A. Asynchronous rlhf: Faster and more efficient off-policy rl for language models. *arXiv preprint arXiv:2410.18252*, 2024.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y., Wu, Y., et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Sheng, G., Zhang, C., Ye, Z., Wu, X., Zhang, W., Zhang, R., Peng, Y., Lin, H., and Wu, C. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Sun, Z., Shen, Y., Zhou, Q., Zhang, H., Chen, Z., Cox, D., Yang, Y., and Gan, C. Principle-driven self-alignment of language models from scratch with minimal human supervision. *Advances in Neural Information Processing Systems*, 36:2511–2565, 2023.

Sun, Z., Shen, Y., Zhang, H., Zhou, Q., Chen, Z., Cox, D., Yang, Y., and Gan, C. Salmon: Self-alignment with instructable reward models. In *International Conference on Learning Representations*, 2024.

Team, K., Du, A., Gao, B., Xing, B., Jiang, C., Chen, C., Li, C., Xiao, C., Du, C., Liao, C., et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025.

Toneva, M., Sordoni, A., des Combes, R. T., Trischler, A., Bengio, Y., and Gordon, G. J. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2018.

von Werra, L., Belkada, Y., Tunstall, L., Beeching, E., Thrush, T., Lambert, N., Huang, S., Rasul, K., and Gallouédec, Q. Trl: Transformer reinforcement learning. https://github.com/huggingface/trl, 2020.

Wang, Y., Yang, Q., Zeng, Z., Ren, L., Liu, L., Peng, B., Cheng, H., He, X., Wang, K., Gao, J., et al. Reinforcement learning for reasoning in large language models with one training example. *arXiv preprint arXiv:2504.20571*, 2025.

Xia, M., Malladi, S., Gururangan, S., Arora, S., and Chen, D. Less: selecting influential data for targeted instruction tuning. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 54104–54132, 2024.

Xiong, W., Yao, J., Xu, Y., Pang, B., Wang, L., Sahoo, D., Li, J., Jiang, N., Zhang, T., Xiong, C., et al. A minimalist approach to llm reasoning: from rejection sampling to reinforce. *arXiv preprint arXiv:2504.11343*, 2025.

Yang, A., Zhang, B., Hui, B., Gao, B., Yu, B., Li, C., Liu, D., Tu, J., Zhou, J., Lin, J., et al. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

Ye, Y., Huang, Z., Xiao, Y., Chern, E., Xia, S., and Liu, P. Limo: Less is more for reasoning. *arXiv preprint arXiv:2502.03387*, 2025.

Yu, Q., Zhang, Z., Zhu, R., Yuan, Y., Zuo, X., Yue, Y., Fan, T., Liu, G., Liu, L., Liu, X., et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Yuan, Y., Yu, Q., Zuo, X., Zhu, R., Xu, W., Chen, J., Wang, C., Fan, T., Du, Z., Wei, X., et al. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*, 2025a.

Yuan, Y., Yue, Y., Zhu, R., Fan, T., and Yan, L. What's behind ppo's collapse in long-cot? value optimization holds the secret. *arXiv preprint arXiv:2503.01491*, 2025b.

Zhang, X., Li, C., Zong, Y., Ying, Z., He, L., and Qiu, X. Evaluating the performance of large language models on gaokao benchmark. *arXiv preprint arXiv:2305.12474*, 2023.

Zhang, X., Wang, J., Cheng, Z., Zhuang, W., Lin, Z., Zhang, M., Wang, S., Cui, Y., Wang, C., Peng, J., Jiang, S., Kuang, S., Yin, S., Wen, C., Zhang, H., Chen, B., and Yu, B. Srpo: A cross-domain implementation of large-scale reinforcement learning on llm, 2025. URL https://arxiv.org/abs/2504.14286.

Zhao, Y., Gu, A., Varma, R., Luo, L., Huang, C.-C., Xu, M., Wright, L., Shojanazeri, H., Ott, M., Shleifer, S., et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023.

Zheng, H., Liu, R., Lai, F., and Prakash, A. Coverage-centric coreset selection for high pruning rates. In *11th International Conference on Learning Representations, ICLR 2023*, 2023.

Zhong, Y., Zhang, Z., Wu, B., Liu, S., Chen, Y., Wan, C., Hu, H., Xia, L., Ming, R., Zhu, Y., et al. Optimizing {RLHF} training for large language models with stage fusion. In *22nd USENIX Symposium on Networked Systems Design and Implementation (NSDI 25)*, pp. 489–503, 2025.

A. Background and Related Work

A.1. Background: Group Relative Policy Optimization

Group Relative Policy Optimization (GRPO)(Shao et al., 2024) is a variant of Proximal Policy Optimization (PPO) (Ouyang et al., 2022) tailored for language model fine-tuning. Instead of computing advantages using a value function, GRPO normalizes reward scores within groups of responses sampled for the same prompt, which largely improves the training efficiency. GRPO has shown superior performance in recent advances (Yu et al., 2025; Li et al., 2025; Wang et al., 2025; Guo et al., 2025) in RL for LLMs, especially for reasoning tasks. GRPO aims to maximize the following objective:

$$\begin{aligned}&\mathcal{J}_{GRPO}(\theta) = \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{old}}(O|q)] \\ &\frac{1}{G}\sum_{i=1}^{G}\left(\min\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}A_i, \text{clip}\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}, 1-\epsilon, 1+\epsilon\right)A_i\right) - \beta\mathbb{D}_{KL}\left(\pi_\theta||\pi_{ref}\right)\right),\end{aligned} \tag{5}$$

where A_i is the advantage, computed using a group of rewards $\{r_1, r_2, \ldots, r_G\}$ corresponding to the outputs within each group:

$$A_{i,t} = \frac{r_i - \text{mean}(\{R_i\}_{i=1}^{G})}{\text{std}(\{R_i\}_{i=1}^{G})}. \tag{6}$$

The advantage of each response is computed as a normalized reward within a group of repeated rollouts. When all responses in a group receive the same reward, regardless of whether they are all correct or all incorrect, the resulting reward variance is zero, and the computed advantages for those responses are all zero. As a result, these examples provide no learning signal during training. In this paper, we refer to such prompts as *zero-variance prompts*, while prompts that yield non-identical rewards across responses are termed *effective prompts*.

A.2. Related Work

RL for LLM Reasoning. Reinforcement learning (RL) was initially used to align model outputs with human preferences (Ouyang et al., 2022; Dai et al., 2023). Since then, RL has become a commonly used technique for fine-tuning LLMs, enabling them to generate more helpful, harmless, and honest responses by incorporating reward signals from human feedback (Christiano et al., 2017; Bai et al., 2022). Recent advances (Guo et al., 2025; Yu et al., 2025; Team et al., 2025; Gao et al., 2024) in LLM reasoning show that Reinforcement Learning with Verifiable Reward (RLVR), which relies on verifiable reward signals instead of model-generated scoress, can effectively improve model reasoning ability. These gains are achieved using various policy optimization methods such as PPO (Ouyang et al., 2022) and GRPO (Shao et al., 2024). Encouraged by the success of RLVR, a growing body of work (Kazemnejad et al., 2024; Yuan et al., 2025b;a; Yu et al., 2025; Liu et al., 2025; Luo et al., 2025; Zhang et al., 2025; Hu, 2025; Xiong et al., 2025) has emerged to further improve reinforcement learning methods for LLM reasoning. For instance, methods such as VinePPO (Kazemnejad et al., 2024), VC-PPO (Yuan et al., 2025b), and VAPO (Yuan et al., 2025a) aim to enhance LLM reasoning by optimizing the value function Meanwhile, DAPO (Yu et al., 2025) introduces several techniques to improve GRPO training, including Dynamic Sampling, which filters out zero-variance prompts and refills the training batch with effective training data through resampling.

Data Selection for LLM. In addition to improving training algorithms, another line of work (Ivison et al., 2025; Xia et al., 2024; Muennighoff et al., 2025a; Ye et al., 2025) seeks to enhance the efficiency and effectiveness of LLM training through data selection strategies. Several approaches (Xia et al., 2024; Chen et al., 2024; Ivison et al., 2023) focus on pruning data used for supervised fine-tuning. For example, S1 (Muennighoff et al., 2025b) reduces a large set of 59k examples to just 1k high-quality samples. In parallel, another thread of research (Muldrew et al., 2024; Liu et al., 2024; Das et al., 2024; Li et al., 2025; Fatemi et al., 2025; Wang et al., 2025) targets improving data efficiency in reinforcement learning for LLMs. For instance, recent research (Li et al., 2025; Wang et al., 2025) shows that only a small subset of the original training dataset is necessary for GRPO to improve the model's reasoning ability. However, those methods rely on training models with the full dataset first to identify important samples and do not offer clear improvements in end-to-end RL training efficiency.

Algorithm 1 Training Iteration in GRESO

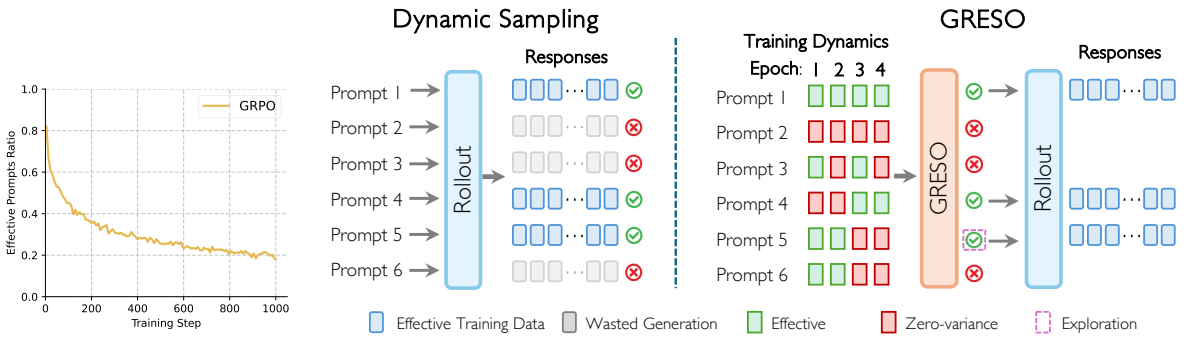
Input: Dataset $\mathcal{D}$; Default rollout batch size $B_{\text{r}}^{\text{default}}$; Training batch size B_{t}; Base exploration probability: p_{easy}, p_{hard}; Targeted zero-variance percentage: $\alpha_{easy}, \alpha_{hard}$
$\mathcal{B} \leftarrow \emptyset$;
$B_{\text{r}} \leftarrow B_{\text{r}}^{\text{default}}$;
$n_{easy}, n_{hard}, n_{total} \leftarrow 0, 0, 0$;
repeat
    Sample prompts $\{x_i\}_{i=1}^{B_{\text{r}}}$ from $\mathcal{D}$
    and filter with Eq. 1 until batch size = B_{r};
    Generate rollouts: $\{x_i, r_i\}_{i=1}^{B_{\text{r}} \times G}$;
    Filter out zero-variance prompts: $\{x_i, r_i\}_{i=1}^{B_{\text{f}} \times G}$;
    $n_{easy} \leftarrow n_{easy}+$ filtered easy zero-var count;
    $n_{hard} \leftarrow n_{hard}+$ filtered hard zero-var count;
    $n_{total} \leftarrow n_{total} + B_{\text{r}}$;
    $\mathcal{B} \leftarrow \mathcal{B} \cup \{x_i, r_i\}_{i=1}^{B_{\text{f}} \times G}$;
    $B_{\text{r}} \leftarrow \min(B_{\text{r}}^{\text{default}}$, Adaptive batch size by Eq. 4);
until $|\mathcal{B}| \geq B_{\text{t}}$
if $n_{easy}/n_{total} \geq \alpha_{easy}$ **then**
    $p_{easy} \leftarrow p_{easy} - \Delta p$;
else
    $p_{easy} \leftarrow p_{easy} + \Delta p$;
end if
if $n_{hard}/n_{total} \geq \alpha_{hard}$ **then**
    $p_{hard} \leftarrow p_{hard} - \Delta p$;
else
    $p_{hard} \leftarrow p_{hard} + \Delta p$;
end if
Select B_{t} examples from $\mathcal{B}$;
Update actor model with GRPO on selected batch;

B. Methodology Details

C. Detailed Experimental Setting

Models & Datasets. We run our experiments on Qwen2.5-Math-1.5B (Yang et al., 2024), DeepSeek-R1-Distill-Qwen-1.5B (Guo et al., 2025), and Qwen2.5-Math-7B (Yang et al., 2024). For Qwen2.5-Math-1.5B/7B models, we use 4096 as the context length, as it is the maximum context length for those two models. For DeepSeek-R1-Distill-Qwen-1.5B, we set the context length to 8196. We evaluate our methods on two training datasets as two settings: 1) DAPO+MATH (DM): We combine the DAPO dataset (Yu et al., 2025), which contains only integer solutions, with the MATH dataset (Hendrycks et al., 2021), which also contains LaTeX-formatted solutions. We find that training on DAPO alone can degrade performance on LaTeX-based benchmarks, so we augment it with MATH to preserve formatting diversity and improve generalization. 2) OPEN-R1 30k subset (R1): A 30,000-example subset of the OPEN-R1 math dataset (Face, 2025).

Training. Our method is implemented based on verl (Sheng et al., 2024) pipeline and uses vLLM (Kwon et al., 2023) for rollout. We use 4xH100 for Qwen2.5-Math-1.5B training and 8xH100 for Qwen2.5-Math-7B and DeepSeek-R1-Distill-Qwen-1.5B. We set the rollout temperature to 1 for vLLM (Kwon et al., 2023). The training batch size is set to 256, and the mini-batch size to 512. We sample 8 responses per prompt. We set the default rollout sampling batch size as 384. For DeepSeek-R1-Distill-Qwen-1.5B, we set the context length to 8196. The training batch size is set to 128, and the mini-batch size to 512. We also sample 8 responses per prompt. We set the default rollout sampling batch size as 192. We train all models for 1000 steps, and we optimize the actor model using the AdamW (Loshchilov & Hutter, 2019) optimizer with a constant learning rate of 1e-6. We use $\beta_1 = 0.9$, $\beta_2 = 0.999$, and apply a weight decay of 0.01. We use the following question template to prompt the LLM. For reward assignment, we give a score of 0.1 for successfully extracting an answer and a score of 1.0 if the extracted answer is correct. Similar to (Yu et al., 2025), we remove the KL-divergence term. The optimization is performed on the parameters of the actor module wrapped with Fully Sharded Data



(a) (b)

Figure 4: **(a)** Dynamics of effective prompts ratio in each step in GRPO training. The ratio keeps decreasing as the training proceeds. **(b)** Pipeline comparison between Dynamic Sampling and our GRESO method. Unlike Dynamic Sampling, which filters out zero-variance prompts *after rollout*, GRESO efficiently predicts and filters them based on training dynamics *before rollout*, which improves rollout efficiency. The probabilistic filtering also allows zero-variance prompts to still be occasionally sampled, enabling the model to revisit potentially valuable prompts.

Parallel (FSDP) (Zhao et al., 2023) for efficient distributed training. We use 4 H100 for Qwen2.5-Math-1.5B training and 8 H100 for Qwen2.5-Math-7B and DeepSeek-R1-Distill-Qwen-1.5B (as it has a longer context length.) We set the targeted zero-variance percentage to 25% for all experiments and allocate it between easy and hard prompts in an $1:2$ ratio (i.e., 8.3% for easy and 16.7% for hard zero-variance prompts), based on the intuition that, as models become more capable during training, more exploration on hard examples can be more beneficial. However, a more optimal allocation scheme may exist, which we leave for future study. We set the initial exploration probability to 50% and base exploration probability adjustment step size Δp for base exploration probability to 1%. We also set a minimal base exploration probability to 5% to ensure a minimal level of exploration on zero-variance prompts throughout training.

GRESO with Fixed Parameters Across All Experiments. Although GRESO introduces a few hyperparameters, we argue that hyperparameter tuning is not a major concern in practice. We designed GRESO (e.g., self-adjustable base exploration probability) to be robust under default settings and *conducted all experiments using a single fixed set of hyperparameters across models and tasks.* The consistent performance observed across different models and tasks demonstrates that GRESO does not rely on extensive hyperparameter tuning, making it both practical and easy to integrate into existing RL fine-tuning pipelines.

Evaluation. For benchmark datasets, we use six widely used complex mathematical reasoning benchmarks to evaluate the performance of trained models: Math500 (Hendrycks et al., 2021; Lightman et al., 2023), AIME24 (Art of Problem Solving, 2024a), AMC (Art of Problem Solving, 2024b), Minerva Math (Lewkowycz et al., 2022), Gaokao (Zhang et al., 2023), Olympiad Bench (He et al., 2024). Same as the training setting, For Qwen2.5-Math-1.5B/7B models, we use 4096 as the context length. For DeepSeek-R1-Distill-Qwen-1.5B, we set the context length to 8196. Similar to (Wang et al., 2025), we evaluate models on those benchmarks every 50 steps and report the performance of the checkpoint that obtains the best average performance on six benchmarks. We evaluate all models with temperature $= 1$ and repeat the test set 4 times for evaluation stability, i.e., $pass@1(avg@4)$, for all benchmarks.

Question Template

Please solve the following math problem: {{Question Description}}. The assistant first thinks about the reasoning process step by step and then provides the user with the answer. Return the final answer in \boxed{} tags, for example \boxed{1}. Let's solve this step by step.

D. Additional Experiments

D.1. Detailed Sampling Comparison

We include Open R1 subset (OR1) in Table 2. GRESO achieves comparable performance with many fewer rollouts.

Table 2: Performance (%) comparison across six math reasoning benchmarks. We train three models on DAPO + MATH (DM) and the Open R1 subset (OR1). Compared to Dynamic Sampling (DS), GRESO achieves similar accuracy while significantly reducing the number of rollouts.

Dataset	Method	Math500	AIME24	AMC	Gaokao	Miner.	Olymp.	Avg.	# Rollout
Qwen2.5-Math-1.5B (Yang et al., 2024)									
DM	DS	77.3	16.7	61.7	64.2	31.8	38.7	48.4	7.6M
	GRESO	76.6	15.0	61.4	66.2	33.3	38.5	48.5	**3.3M**
OR1	DS	77.1	16.7	50.3	65.5	30.9	39.7	46.7	3.8M
	GRESO	76.1	20.0	50.6	65.1	30.0	39.2	46.8	**1.6M**
DeepSeek-R1-Distill-Qwen-1.5B (Guo et al., 2025)									
DM	DS	87.9	36.7	71.7	78.7	35.3	55.9	61.0	2.4M
	GRESO	87.7	36.7	71.1	78.4	33.9	55.1	60.5	**1.6M**
OR1	DS	84.8	25.0	68.4	74.0	34.1	54.2	56.7	2.4M
	GRESO	85.9	26.7	66.9	75.2	33.6	55.5	57.3	**1.5M**
Qwen2.5-Math-7B (Yang et al., 2024)									
DM	DS	82.9	34.2	79.2	71.7	35.4	43.6	57.8	13.1M
	GRESO	82.2	32.5	80.7	70.2	35.4	44.1	57.5	**6.3M**
OR1	DS	82.9	34.2	63.1	67.3	34.9	46.3	54.8	11.4M
	GRESO	82.3	35.0	64.5	66.8	36.5	45.7	55.1	**3.4M**

Table 3: Training time (hours) breakdown and comparison for models trained on DAPO + MATH dataset. GRESO consistently lowers rollout cost and achieves up to **2.4× speedup** in rollout and **2.0× speedup** in total training cost over Dynamic Sampling.

Method	Training	Other	Rollout	Total
Qwen2.5-Math-1.5B				
DS	8.1	3.6	41.0 (1.0×)	52.6 (1.0×)
GRESO	8.9	3.9	**25.2** (**1.6**×)	**37.9** (**1.4**×)
DeepSeek-R1-Distill-Qwen-1.5B				
DS	6.1	3.3	92.4 (1.0×)	101.9 (1.0×)
GRESO	6.8	4.0	**62.0** (**1.5**×)	**72.7** (**1.4**×)
Qwen2.5-Math-7B				
DS	16.1	6.1	155.9 (1.0×)	178.0 (1.0×)
GRESO	16.6	6.3	**65.5** (**2.4**×)	**88.3** (**2.0**×)

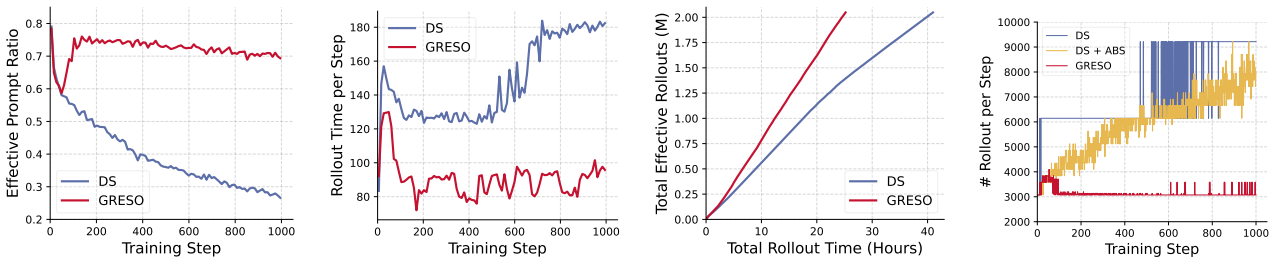
D.2. Wall-clock Time Performance Comparison

Up to 2.4× wall-clock time speed-up in rollout and 2.0× speed-up in training. To better understand the efficiency of our proposed methods, we report the detailed end-to-end training time (1000 steps) breakdown for different stages: rollout generation, actor model update, and other overheads (e.g., reference model and advantage calculation). Qwen2.5-Math-1.5B is trained on 4×H100 GPUs, while the other two models are trained on 8×H100 GPUs. Table 3 compares the training time breakdown between GRESO and Dynamic Sampling for models trained on the DAPO + MATH dataset. For all three models, GRESO significantly reduces rollout time—achieving up to **2.4× speedup** in rollout and **2.0× speedup** in total training time compared to DS. For instance, on Qwen2.5-Math-7B, GRESO reduces rollout time from 155.9 hours to 65.5 hours, cutting overall training time from 178.0 to 88.3 hours.

D.3. Analysis and Ablation Study

In this section, we use Qwen-Math-1.5B trained on the DAPO + MATH dataset to analyze in detail how GRESO reduces training overhead by enhancing rollout quality, and we also conduct an ablation study on the contribution of each component in GRESO.

GRESO improves effective prompt ratio and rollout efficiency. As shown in Figure 5a, compared to Dynamic Sampling,

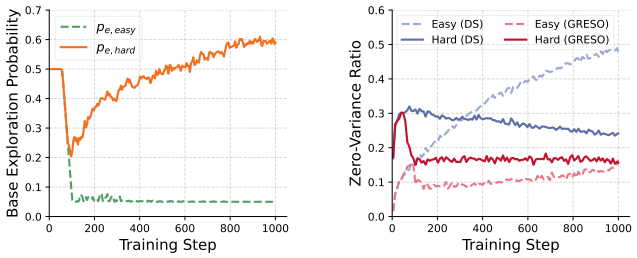


(a) (b) (c) (d)

Figure 5: Training dynamics analysis of Qwen-Math-1.5B trained on the DAPO + MATH dataset: **(a)** Effective prompt ratio in each step. GRESO maintains a consistently higher effective prompt ratio during training. **(b)** To obtain the same number of effective prompts per batch, GRESO requires less rollout time. **(c)** GRESO achieves more effective rollouts for training under the same rollout time budget compared to Dynamic Sampling. **(d)** Ablation study on adaptive batch size (ABS) for sampling: Both ABS and GRESO effectively reduce the number of rollouts per training step.

where effective prompt ratio steadily decreases during training, since GRESO filter out many zero-variance prompts before rollout, GRESO consistently maintains a significantly higher effective prompt ratio. For instance, as effective prompt ratio drop to around 20% in the late stage of training, GRESO maintains the effective prompt ratio larger than 70%. This higher ratio directly translates into reduced rollout time per training step, as fewer zero-variance prompts are sampled. Figure 5b shows that GRESO has significantly less rollout time per step compared to dynamic sampling. Figure 5c compares the total number of effective rollouts used during training under the same rollout time budget for GRESO and Dynamic Sampling. GRESO consistently generates more effective rollouts over time. For instance, GRESO reaches 2 million effective rollouts in approximately 25 hours, while Dynamic Sampling requires over 40 hours to achieve the same, which demonstrate the efficiency of GRESO.

Dynamics of self-adjustable base exploration probabilities. A key parameter in GRESO is the base exploration probability p_e defined in Equation 1. As discussed in Section 3.1, this probability can vary depending on the model, dataset, and training stage. Instead of manually tuning p_e, GRESO employs an adaptive mechanism to automatically adjust it during training. Specifically, GRESO maintains separate exploration probabilities for hard and easy zero-variance prompts, denoted as $p_{e,\text{hard}}$ and $p_{e,\text{easy}}$, respectively. In Figure 6a, we plot the dynamics of both $p_{e,\text{hard}}$ and $p_{e,\text{easy}}$, along with the ratio of easy and hard zero-variance prompts over time. We observe that after the first training epoch, both exploration probabilities initially decline. However, as the model ability improves, $p_{e,\text{hard}}$ begins to increase, enabling more exploration of hard examples during later stages of training. Figure 6b shows the dynamics of easy and hard zero-variance ratios. Unlike Dynamic Sampling, GRESO effectively maintains both ratios close to their target values during training, which demonstrates the effectiveness of its self-adjusting mechanism.



(a) (b)

Figure 6: **(a)** Dynamics of base exploration probabilities. **(b)** Dynamics of easy and hard zero-variance prompt ratio.

Selection Dynamics. In Figure 7, we present a case study illustrating how GRESO selects or skips prompts over training epochs. We observe that very easy prompts tend to remain easy throughout training; although frequently skipped, GRESO still occasionally selects them to ensure a minimal level of exploration. For prompts of moderate difficulty, as the model becomes stronger over time, these prompts gradually become easier and are increasingly skipped. In contrast, some hard prompts become solvable (i.e., effective prompts) in later epochs or even easy prompts. However, certain hard prompts remain unsolved throughout training.

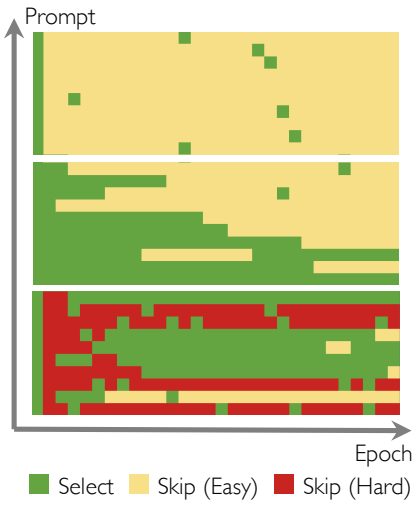


Figure 7: Selection Dynamics of different prompts in GRESO. Each row is a prompt, and each column is an epoch.

Ablation study on adaptive batch size (ABS) for sampling. In addition to the prompt selection algorithm based on training dynamics, another key component of GRESO is the adaptive batch size (ABS) for sampling. When only a small number of effective prompts are needed to fill the training batch, ABS enables rollout on a smaller batch instead of using the default large sampling batch size, thereby reducing unnecessary computation. Figure 5d compares the number of rollouts per training step across three methods: Dynamic Sampling (DS), DS with Adaptive Batch Size (DS + ABS), and GRESO. DS maintains a fixed sampling batch size, leading to consistently high sampling overhead. DS + ABS dynamically adjusts the batch size, reducing the number of samples in earlier steps, but still shows increasing sampling as training progresses and the effective prompt ratio decreases. In contrast, GRESO consistently maintains a much lower number of samples per step due to its more selective rollout strategy combined with ABS, resulting in significantly reduced rollout overhead throughout training.

D.4. Impact of Targeted Zero-variance Percentage

In this section, we study how varying the targeted zero-variance percentage impacts training and rollout efficiency. In addition to the default setting of 25% used throughout our experiments, we also evaluate alternative values of 0, 50%, 100% (i.e., always allow exploration). As shown in Table 4, different zero-variance targets give us nearly identical accuracy. We also present the number of rollouts per step in Figure 8. When we reduce the targeted zero-variance ratio to 0, we observe that the number of rollouts per step remains similar to that of the 25% setting. This lack of difference can be attributed to two factors. First, we enforce a minimum exploration rate of 5%, which ensures that some exploration still occurs. As a result, the actual zero-variance percentage never truly reaches 0. Second, we always oversample some data in the first batch of rollouts in each iteration to provide some redundancy to avoid the second batch of rollouts. With this setting, as long as the first batch generates enough effective training data to fill the training batch, regardless of whether the target is 0 or 25%, the total number of rollouts remains approximately the same. In addition, as the targeted zero-variance percentage increases, more zero-variance prompts are allowed during rollout, leading to a higher number of rollouts per step. When the targeted percentage becomes sufficiently large, GRESO gradually approaches the behavior of dynamic sampling with adaptive rollout batch size.

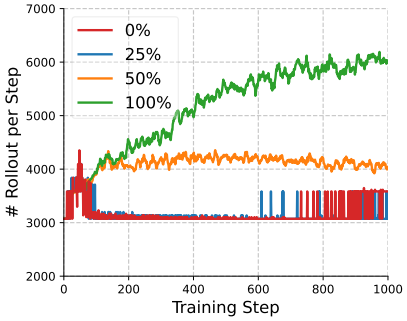


Figure 8: Comparison of the number of rollouts across different target zero-variance ratios.

Table 4: Average accuracy across six math reasoning benchmarks under different targeted zero-variance percentages.

Target (%)	0	25	50	100
Acc. (%)	48.1	48.5	48.5	48.4

D.5. Alternative Design: Linear Backoff

In addition to the probabilistic filtering approach introduced in Section 4.2 of the main paper, we also explored an alternative solution for filtering zero-variance prompts during the early stages of this project. One such method is the *backoff algorithm* (Kwak et al., 2005) (e.g., linear backoff). Specifically, if a prompt is identified as zero-variance in the most recent k rollouts, it is skipped for the next k training epochs. However, there are several limitations to this approach. As discussed in Section 4 of the paper, the degree of exploration should adapt to the model, dataset, and training stage. The linear backoff algorithm schedules the next rollout for a zero-variance prompt k epochs into the future. As a result, if we wish to adjust the exploration intensity dynamically based on new observations or evolving training dynamics, the backoff algorithm cannot directly affect prompts that have already been deferred to future epochs. For instance, as shown in Figure 9, unlike probabilistic filtering, filtering based on linear backoff can cause periodic fluctuations in zero-variance prompt ratio, which differs from the smoother dynamics enabled by probabilistic filtering This lack of flexibility limits its ability to adapt exploration strategies in a fine-grained or responsive manner, which motivated the design of our current GRESO algorithm based on probabilistic filtering.

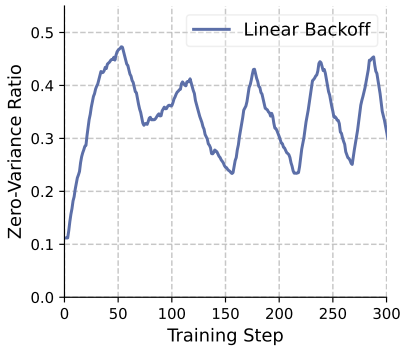


Figure 9: Zero-variance prompt ratio dynamic for linear backoff.

D.6. Case study of Filtered Examples

To better understand the behavioral patterns of our selective filtering algorithm, we present a case study of prompts that were frequently skipped or selected during training from the MATH (Hendrycks et al., 2021) dataset. We categorize the examples into three groups: Frequently Skipped Prompts (Easy), Frequently Skipped Prompts (Hard), Frequently Selected Prompts. We observe that frequently skipped easy prompts often involve straightforward calculations or routine applications of formulas, making them more likely to be solved across all sampled responses. Frequently selected prompts tend to exhibit moderate difficulty, contributing more consistently to model improvement. As for frequently skipped hard prompts, these problems are too challenging for the model to solve, even across multiple rollouts, resulting in zero variance among the rewards and ultimately failing to contribute to training.

Frequently Skipped Prompts (Easy)

1. **Question:** Johnny has 7 different colored marbles in his bag. In how many ways can he choose three different marbles from his bag to play a game? **Solution:** 35.

2. **Question:** The number n is a prime number between 20 and 30. If you divide n by 8, the remainder is 5. What is the value of n? **Solution:** 29.

3. **Question:** Evaluate: $\frac{10^{-2}\cdot 5^0}{10^{-3}}$ **Solution:** 10.

4. **Question:** The Ponde family's Powerjet pumps 420 gallons of water per hour. At this rate, how many gallons of water will it pump in 45 minutes? **Solution:** 315.

5. **Question:** Suppose that $n, n+1, n+2, n+3, n+4$ are five consecutive integers. Determine a simplified expression for the sum of these five consecutive integers. **Solution:** $5n + 10$.

Frequently Skipped Prompts (Hard)

1. **Question:** A parabola and an ellipse share a focus, and the directrix of the parabola is the line containing the minor axis of the ellipse. The parabola and ellipse intersect at two points. Given that the equation of the ellipse is $\frac{x^2}{25} + \frac{y^2}{9} = 1$, find the distance between those two points. **Solution:** $\frac{4\sqrt{14}}{3}$.

2. **Question:** In triangle ABC, $AB = AC = 100$, and $BC = 56$. Circle P has radius 16 and is tangent to $\overline{AC}$ and $\overline{BC}$. Circle Q is externally tangent to P and is tangent to $\overline{AB}$ and $\overline{BC}$. No point of circle Q lies outside of $\triangle ABC$. The radius of circle Q can be expressed in the form $m - n\sqrt{k}$, where m, n, and k are positive integers and k is the product of distinct primes. Find $m + nk$. **Solution:** 254.

3. **Question:** Let $EFGH$, $EFDC$, and $EHBC$ be three adjacent square faces of a cube, for which $EC = 8$, and let A be the eighth vertex of the cube. Let I, J, and K, be the points on $\overline{EF}$, $\overline{EH}$, and $\overline{EC}$, respectively, so that $EI = EJ = EK = 2$. A solid S is obtained by drilling a tunnel through the cube. The sides of the tunnel are planes parallel to $\overline{AE}$, and containing the edges $\overline{IJ}$, $\overline{JK}$, and $\overline{KI}$. The surface area of S, including the walls of the tunnel, is $m + n\sqrt{p}$, where m, n, and p are positive integers and p is not divisible by the square of any prime. Find $m + n + p$. **Solution:** 417.

4. **Question:** Let a and b be nonnegative real numbers such that

$$\sin(ax + b) = \sin 29x$$

for all integers x. Find the smallest possible value of a. **Solution:** $10\pi - 29$.

5. **Question:** Four people sit around a circular table, and each person will roll a standard six-sided die. What is the probability that no two people sitting next to each other will roll the same number after they each roll the die once? Express your answer as a common fraction. **Solution:** $\frac{35}{72}$.

Frequently Selected Prompts

1. **Question:** Let x, y, and z be three positive real numbers whose sum is 1. If no one of these numbers is more than twice any other, then find the minimum value of the product xyz. **Solution:** $\frac{1}{32}$.

2. **Question:** The number

$$e^{7\pi i/60} + e^{17\pi i/60} + e^{27\pi i/60} + e^{37\pi i/60} + e^{47\pi i/60}$$

is expressed in the form $re^{i\theta}$, where $0 \le \theta < 2\pi$. Find θ. **Solution:** $\dfrac{9\pi}{20}$.

3. **Question:** For what values of x is

$$\frac{x - 10x^2 + 25x^3}{8 - x^3}$$

nonnegative? Answer as an interval. **Solution:** $[0, 2)$.

4. **Question:** Determine all real numbers a such that the inequality $|x^2 + 2ax + 3a| \le 2$ has exactly one solution in x. **Solution:** 1, 2.

5. **Question:** By starting with a million and alternatively dividing by 2 and multiplying by 5, Anisha created a sequence of integers that starts 1000000, 500000, 2500000, 1250000, and so on. What is the last integer in her sequence? Express your answer in the form a^b, where a and b are positive integers and a is as small as possible. **Solution:** 5^{12}.

E. Limitations

While GRESO effectively filters out the most obvious zero-variance training prompts—those that contribute no learning signal to the model, it does not estimate or rank the value of the remaining prompts, which can also contain uninformative prompts that provide limited contribution to training. A potential future work for GRESO is to extend its filtering mechanism beyond binary decisions by incorporating a finer-grained scoring or ranking system to prioritize prompts based on their estimated training utility. Despite that, we view GRESO as an important first step toward such an advanced data selection

algorithm for efficient rollout and believe it provides a solid foundation for more adaptive and efficient reinforcement learning in LLM training.

F. Broader Impact

This work enhances the efficiency and scalability of RL-based fine-tuning for language models by introducing a lightweight, selective rollout mechanism that filters out uninformative prompts. By significantly reducing redundant computation, our method lowers overall training costs. This makes it easier for institutions with limited computational budgets to train strong models, helping democratize access to advanced AI. Furthermore, our approach promotes more sustainable and resource-efficient practices, encouraging future research toward greener and more inclusive large-scale training.