# Dissecting Long Reasoning Models: An Empirical Study

**Anonymous ACL submission**

## Abstract

Despite recent progress in training long-context reasoning models via reinforcement learning (RL), several open questions and counterintuitive behaviors remain. This work focuses on three key aspects: (1) We systematically analyze the roles of positive and negative samples in RL, revealing that positive samples mainly facilitate data fitting, whereas negative samples significantly enhance generalization and robustness. Interestingly, training solely on negative samples can rival standard RL training performance. (2) We identify substantial data inefficiency in group relative policy optimization, where over half of the samples yield zero advantage. To address this, we explore two straightforward strategies, including relative length rewards and offline sample injection, to better leverage these data and enhance reasoning efficiency and capability. (3) We investigate unstable performance across various reasoning models and benchmarks, attributing instability to uncertain problems with ambiguous outcomes, and demonstrate that multiple evaluation runs mitigate this issue.

## 1 Introduction

Natural language processing has witnessed a breakthrough in long reasoning capabilities within large language models (LLMs). Unlike previous models depending on chain-of-thought (CoT) prompting (Wei et al., 2022), recent models emphasize scaling up inference computation for longer reasoning processes and enabling self-directed behaviors such as speculation, exploration, reflection, and verification (OpenAI, 2025).

Achieving such improvement is non-trivial, as it is challenging to construct high-quality supervised datasets that enable models to perform meticulous reasoning. Recent works reveal that scaling reinforcement learning (RL) plays a vital role in this context (DeepSeek-AI et al., 2025; Team et al., 2025). Compared to running supervised next token prediction, RL offers two key advantages. First, it eliminates the need for labeled data, enabling training on reasoning tasks without annotated intermediate steps. Second, its supervised signals come from feedback of the model's own generated responses, promoting the discovery of self-suitable reasoning routes towards correct answers.

Despite impressive advancements, there exist intriguing questions and counterintuitive phenomena when training and evaluating long reasoning models. In this work, we provide an investigation and analysis of the following three points:

*Role of positive and negative samples:* RL typically optimizes on positive samples with advantages greater than zero, while suppressing negative samples with advantages less than zero. However, it remains unclear what models actually learn from positive and negative samples, and whether learning from negative samples—especially when they correspond to clearly incorrect answers—is necessary. Through systematic ablation on both sample types, we find that positive samples primarily help a model to fit the training data, while negative samples significantly improve generalization and robustness in long CoT reasoning. Surprisingly, we observe that even training solely on negative samples can yield performance comparable to standard RL training.

*Zero advantage problem within GRPO:* We identify a critical limitation of rule-based reward functions in group relative policy optimization (GRPO): when facing overly easy or overly difficult problems, the discrete rewards easily lead to zero advantages, resulting in substantial data inefficiency. Our experiments show that this issue is prevalent in widely used datasets, where gradient elimination occurs in over half of the data. To address this, we explore two straightforward strategies. Specifically, relative length reward (RLR) leverages overly easy samples by assigning additional scores based on relative output length, encouraging more efficient reasoning. Offline sample injection facilitates learning from overly difficult problems by replacing incorrect online samples with correct offline solutions. Experimental results demonstrate that RLR enhances reasoning efficiency without sacrificing overall performance, whereas learning from data beyond the model's capacity remains challenging, even when correct solutions are provided.

*Reason for unstable performance:* Our empirical study shows that performance instability persists across a wide range of reasoning models and benchmarks, regardless of model size or training method. This phenomenon arises from uncertain problems, where neither correct nor incorrect responses have clearly dominant probabilities. While greedy decoding helps output consistency, it might distort evaluation by flipping the correctness of responses. In practice, performing multiple runs still offers a simple yet effective solution to stabilize evaluation scores, particularly on small benchmarks with a high proportion of uncertain problems.

## 2 Preliminary

GRPO (Shao et al., 2024) eliminates the value LLM, improving the training efficiency and reducing a variable for our analysis. Thus, we mainly leverage GRPO to train long reasoning models.

**Group relative policy optimization.** Given a set of problems $Q$, the old policy model $\pi_{\theta_{old}}$ first samples a group of responses $\{o_1, o_2, \cdots, o_G\}$ for each problem $q$. Different from proximal policy optimization (PPO) (Schulman et al., 2017), which trains a value LLM to calculate advantage, GRPO computes the advantage $\hat{A}_{i,t}$ in a group relative manner to eliminate the value LLM, i.e.,

$$\hat{A}_{i,t} = \frac{r_i - \mathrm{mean}\big(\{r_i\}_{i=1}^{G}\big)}{\mathrm{std}\big(\{r_i\}_{i=1}^{G}\big)}, \tag{1}$$

where $r_i$ represents the reward score of the sample $i$ computed by reward function $\mathcal{R}(\cdot)$. $\hat{A}_{i,t}$ persists similarly at each token $t$. Then, GRPO leverages a clipped surrogate objective that constrains the policy updates within a proximal region of the previous policy by:

$$\mathcal{C}_{i,t}(\theta) = \min\Big(r_{i,t}(\theta)\,\hat{A}_{i,t},\ \mathrm{clip}\big(r_{i,t}(\theta), 1-\varepsilon, 1+\varepsilon\big)\,\hat{A}_{i,t}\Big), \tag{2}$$

where $\varepsilon$ is a clipping-related hyperparameter and

$$r_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t} \mid q)}{\pi_{\theta_{\mathrm{old}}}(o_{i,t} \mid q)}. \tag{3}$$

Finally, GRPO updates the policy model $\pi_{\theta_{old}}$ by maximizing the following objective:

$$\mathcal{J}_{\mathrm{GRPO}}(\theta) = \mathbb{E}_{q\sim Q, \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{old}}(O|q)} \left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\mathcal{C}_{i,t}(\theta) - \beta D_{\mathrm{KL}}\big(\pi_\theta \,\|\, \pi_{\mathrm{ref}}\big)\right], \tag{4}$$

where $\beta$ is a hyperparameter and $D_{\mathrm{KL}}\big(\pi_\theta \,\|\, \pi_{\mathrm{ref}}\big)$ is the KL penalty term.

**Rule-based reward.** We use a rule-based reward that verifies response correctness via matching algorithms and assigns scores accordingly. The reward function is defined as follows:

$$\mathcal{R}(\hat{a}, a) = \begin{cases} 1, & \texttt{is\_equivalent}(\hat{a}, a) \\ 0, & \text{otherwise} \end{cases}, \tag{5}$$

where $\hat{a}$ and $a$ denote the answer extracted from the model response and the ground truth, respectively.

**Three-stage training.** Sampling is one of the most time-consuming steps in RL, especially when the response involves tens of thousands of tokens. Luo et al. (2025b) observe that most of the over-length responses are incorrect or consist of endlessly repetitive content. To avoid over-length responses during early training, they adopt a three-stage curriculum with progressively increasing maximum response lengths: 8192, 16384, and 24576 tokens for the first, second, and third stages, respectively.

**Training setups.** We conduct RL training on SFT-trained models, including `DeepSeek-R1-Distill-Qwen-1.5B` and `7B` (DeepSeek-AI et al., 2025), and RL-trained models, `DeepScaleR-1.5B-Preview`[1]. By default, our training dataset is DeepScaleR-40K (Luo et al., 2025b), which contains 40315 mathematics questions collected from competitions and exercises. Please see Appendix B.1 for more details.

**Evaluation setups.** We select several representative reasoning benchmarks: AIME24, AMC23, Math-500 (Hendrycks et al., 2021b), OlympiadBench (He et al., 2024), AIME25, MMLU-STEM (Hendrycks et al., 2021a), GPQA diamond (Rein et al., 2023), and GaoKao Math QA (Zhong et al., 2024). The first four datasets serve as in-domain datasets, while the others are out-of-domain ones. By default, we generate from the models using a

[1]We refer to it as `DeepScaleR-1.5B` for brevity.

| Model | In-domain | | In-domain (Fitness) | | | In-domain | | In-domain (Fitness) | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | AIME24 | AMC23 | AIME24 (8K) | AMC23 (8K) | Training Set | AIME24 | AMC23 | AIME24 (8K) | AMC23 (8K) | Training Set |
| | *DeepSeek-R1-Distill-Qwen-1.5B* | | | | | *DeepSeek-R1-Distill-Qwen-7B* | | | | |
| Orig | 28.80 | 62.73 | 19.22 $_{9.58\downarrow}$ | 52.33 $_{10.39\downarrow}$ | 54.54 | **55.31** | 82.68 | 39.32 $_{15.99\downarrow}$ | 69.82 $_{12.86\downarrow}$ | 70.36 |
| Both | **36.35** | **71.07** | **32.03** $_{4.32\downarrow}$ | **68.11** $_{2.96\downarrow}$ | **60.73** | 54.43 | 84.90 | **51.67** $_{2.76\downarrow}$ | **84.34** $_{0.56\downarrow}$ | **72.53** |
| Neg | 34.95 | 68.77 | 29.69 $_{5.26\downarrow}$ | 64.01 $_{4.76\downarrow}$ | 58.06 | 54.90 | **85.07** | 50.99 $_{3.91\downarrow}$ | 84.28 $_{0.79\downarrow}$ | 71.14 |
| Pos | 34.01 | 69.48 | 30.26 $_{3.75\downarrow}$ | 67.07 $_{2.41\downarrow}$ | 59.26 | 49.17 | 80.03 | 47.86 $_{1.30\downarrow}$ | 79.57 $_{0.45\downarrow}$ | 68.83 |

| Model | Out-of-domain (Generalization) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | AIME25 | GPQA | MMLU-STEM | GaoKao | Average | AIME25 | GPQA | MMLU-STEM | GaoKao | Average |
| | *DeepSeek-R1-Distill-Qwen-1.5B* | | | | | *DeepSeek-R1-Distill-Qwen-7B* | | | | |
| Orig | 23.59 | 15.70 | 44.18 | 81.79 | 41.32 | 38.91 | 37.15 | 72.58 | 90.72 | 59.84 |
| Both | **26.93** | 18.02 | 49.84 | 84.88 | 44.92 | 39.74 | 43.56 | **84.21** | **91.27** | **64.69** |
| Neg | 26.30 | 19.68 | **54.39** | 84.03 | **46.10** | **39.90** | **44.32** | 83.19 | **91.27** | 64.67 |
| Pos | 23.80 | **19.82** | 48.37 | **84.29** | 44.07 | 32.76 | 41.98 | 79.42 | 91.11 | 61.32 |

| Model | Add Noise (Robustness) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | AIME24 (N1) | AMC23 (N1) | AIME24 (N2) | AMC23 (N2) | Average | AIME24 (N1) | AMC23 (N1) | AIME24 (N2) | AMC23 (N2) | Average |
| | *DeepSeek-R1-Distill-Qwen-1.5B* | | | | | *DeepSeek-R1-Distill-Qwen-7B* | | | | |
| Orig | 22.71 $_{6.09\downarrow}$ | 53.46 $_{9.26\downarrow}$ | 23.13 $_{5.68\downarrow}$ | 54.52 $_{8.21\downarrow}$ | 38.45 $_{7.31\downarrow}$ | 25.47 $_{29.84\downarrow}$ | 58.83 $_{23.85\downarrow}$ | 25.05 $_{30.26\downarrow}$ | 60.99 $_{21.69\downarrow}$ | 25.26 |
| Both | 23.85 $_{12.50\downarrow}$ | 58.60 $_{12.46\downarrow}$ | 26.46 $_{9.90\downarrow}$ | 60.49 $_{10.58\downarrow}$ | 42.35 $_{11.36\downarrow}$ | 25.68 $_{28.75\downarrow}$ | 63.23 $_{21.67\downarrow}$ | 24.90 $_{29.53\downarrow}$ | 63.78 $_{21.12\downarrow}$ | 25.29 |
| Neg | **26.25** $_{8.70\downarrow}$ | **63.20** $_{5.57\downarrow}$ | **27.45** $_{7.50\downarrow}$ | **64.51** $_{4.25\downarrow}$ | **45.35** $_{6.51\downarrow}$ | **26.30** $_{28.59\downarrow}$ | **64.08** $_{20.99\downarrow}$ | **26.82** $_{28.07\downarrow}$ | **65.38** $_{19.69\downarrow}$ | **26.56** |
| Pos | 23.70 $_{10.31\downarrow}$ | 52.50 $_{16.98\downarrow}$ | 24.64 $_{9.38\downarrow}$ | 57.79 $_{11.69\downarrow}$ | 39.66 $_{12.09\downarrow}$ | 24.43 $_{24.74\downarrow}$ | 60.82 $_{19.20\downarrow}$ | 24.06 $_{25.10\downarrow}$ | 61.58 $_{18.45\downarrow}$ | 24.24 |

Table 1: Ablation study on positive and negative samples during RL training. "Orig" denotes the model without RL training. "Both", "Neg", and "Pos" indicate the policy model updated on both positive and negative samples, only negative samples, and only positive samples, respectively. "8K" means the maximum output length is set to 8192 during evaluation. "N1" and "N2" refer to two types of many-shot noise (Zaremba et al., 2025) added to the prompt. The subscripts indicate the score differences caused by adding an output length limitation or input noise, relative to the original performance. Moreover, the best scores are **bolded**, and the second-best are underlined.

temperature of 0.6, a Top-p value of 0.95, and a maximum output length of 32768 tokens. We report the Pass@1 score averaged on sufficient runs for each dataset, detailed in Table 5. Please refer to Appendix B.2 for further details.

## 3 Role of Positive and Negative Samples

In RL algorithms, the surrogate objective trains models to fit on positive samples whose advantage is greater than zero, while suppressing generating negative samples with an advantage less than zero. In the context of training long reasoning models with GRPO, the positive samples refer to the correct responses successfully verified by the rule-based reward function, while the negative ones stand for incorrect or unverifiable answers. A natural question arises: *What can long reasoning models learn from positive and negative samples during the RL training? Is learning from negative or positive samples necessary?*

There have been different conjectures about the role of positive and negative samples:

- For a complex reasoning problem, the unsuccessful solution space is significantly larger than the correct one[2]. Thus, an intuitive conjecture is that for long reasoning tasks, learning from the negative samples is marginal, while positive samples contribute mostly.
- Some works argue to discard negative actions but only update the policy on positive ones to optimize conventional RL tasks better (Srinivasan et al., 2018; Jesson et al., 2024).
- Other works of human preference alignment offer an opposite opinion that during RL training, negative gradient, i.e., learning to "push-down" likelihood on negative samples, results in faster accumulation of probability mass on a subset of high-reward responses compared to learning from positive samples supervised (Tajwar et al., 2024).

In this section, we first conduct empirical ablation studies and then provide a theoretical explanation to answer the above question.

### 3.1 Ablation: Positive vs. Negative

The ablation aims to exclude the gradient contributions of a specific type of sample when updating the policy model. To ensure that training on other samples is not affected, we first compute the advantage for all samples using standard sampling and then exclude the target samples before applying the policy update. For instance, to ablate the effect of negative samples, we perform standard sampling and compute the advantage for each sample as usual. We then remove the negative samples from the batch before updating the policy.

**Setups.** We set the maximum sampling length to 8192 and train each model until its performance

[2]Considering that any mistake in any of the reasoning steps will lead to an incorrect result, while the correct solution is limited.

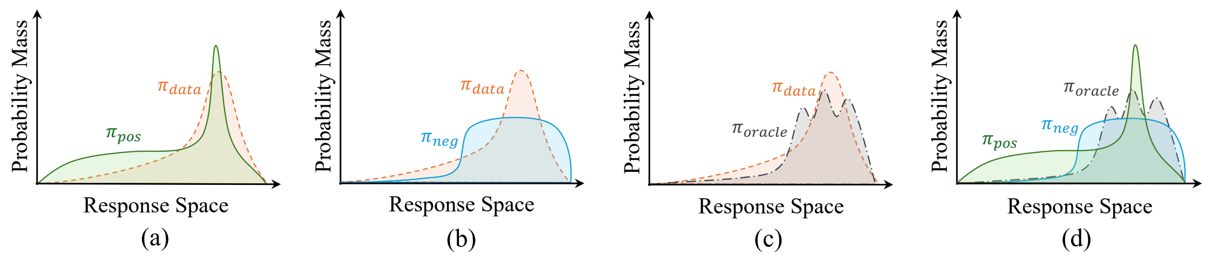


Figure 1: Illustration of training on two types of samples. The orange dotted line denotes the policy $\pi_{data}$ that perfectly reflects the training data, while the gray one refers to the oracle policy $\pi_{oracle}$. The green and blue solid lines represent the policies learned solely from positive and negative samples, i.e., $\pi_{pos}$ and $\pi_{neg}$, respectively.

plateaus. As shown in Table 1, our evaluation covers four aspects. We assess *reasoning ability* using AIME24 and AMC23, *fitness* using two datasets under the 8192-token maximum output length and a sampled training set, *generalization* using four out-of-domain datasets, and *robustness* by injecting many-shot noise into the prompt, following Zaremba et al. (2025); Anil et al. (2024). See Appendix B.3 for more details.

**Results.** As shown in Table 1, training with only one type of sample can lead to substantial improvements over the original models, and often yields performance comparable to training with both types. The only exception is the 7B model trained on positive samples, which shows a performance drop. In *fitness* evaluation, models trained on both positive and negative samples perform best. Notably, limiting the maximum output length to 8192 causes only a slight performance drop in models trained on positive samples, but a significant drop in those trained on negative samples, highlighting the crucial role of positive samples in fitting the training data[3]. For *generalization*, models trained solely on negative samples outperform others. A similar trend is observed in *robustness* evaluation, where negative-sample-only models consistently achieve superior performance, especially for the 1.5B model.

**Takeaway 3.1 for Effect of Samples**

Models trained on both types of samples achieve the best final performance (both $\geq$ negative $>$ positive), with positive samples aiding fitness (both $\geq$ positive $>$ negative), and negative samples improving generalization (negative $\geq$ both $>$ positive) and robustness (negative $>$ both $>$ positive).

[3]Despite the known length bias in GRPO with negative samples (Liu et al., 2025a), our findings remain robust, as detailed in Appendix C.1.

## 3.2 Theoretical Explanation

Suppose a policy $\pi_{data}$ perfectly represents the distribution of the training data, while $\pi_{oracle}$ denotes the oracle distribution of the environment. As shown in Figure 1 (a), a policy $\pi_{pos}$ trained solely on positive samples tends to concentrate its probability mass in regions associated with high reward, but lacks the incentive to explicitly suppress low-reward regions. In contrast, as illustrated in Figure 1 (b), a policy $\pi_{neg}$ trained only on negative samples flattens the probability in low-reward areas, but does not actively promote high-reward regions, leading to a more diffuse distribution.

While $\pi_{pos}$ is more effective at fitting due to its focus on positive outcomes, its performance is highly dependent on how well the training data approximates the oracle distribution. However, as depicted in Figure 1 (c), this approximation is often imperfect. Figure 1 (d) demonstrates that although $\pi_{neg}$ fails to emphasize high-reward regions, it can still cover parts of the oracle distribution that are underrepresented in the training data, making it more robust than $\pi_{pos}$ in certain scenarios.

## 3.3 Negative Samples Pre-training

Considering the characteristics of positive and negative samples, we can regard training on negative samples merely as the pre-training process where the model learning to suppress the low-quality responses massively. Then, we can fine-tune on the positive samples to elevate the specific good probability and boost the performance. The final model is supposed to gain the advantage of both performance and robustness.

**Results.** We perform three-stage training, using only negative samples in the first two stages. The results are presented in Table 2. Surprisingly, models trained exclusively on negative samples throughout all stages (8(N)-16(N)-24(N)) achieve reasoning performance on in-domain datasets compara-

| Model | AIME24 | AMC23 | AIME25 | GPQA | MMLU-STEM | GaoKao | AIME24 (N1) | AMC23 (N1) | AIME24 (N2) | AMC23 (N2) | AVG |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Orig | 28.80 | 62.73 | 23.59 | 15.70 | 44.18 | 81.79 | 22.71 | 53.46 | 23.13 | 54.52 | 41.06 |
| 8(NP)-16(NP)-24(NP) | 40.47 | 72.50 | 29.06 | 19.52 | 51.14 | **85.80** | 27.40 | 58.38 | 28.91 | 61.30 | 47.45 |
| 8(N)-16(N)-24(NP) | 40.26 | 71.07 | 28.33 | **20.77** | **57.34** | 85.35 | 31.35 | **64.82** | 31.61 | 65.19 | **49.61** |
| 8(N)-16(N)-24(N) | 40.16 | **71.52** | 28.75 | 19.24 | 54.74 | 85.04 | **30.68** | 64.08 | 30.26 | 65.68 | 49.01 |
| 8(N)-16(N)-24(N)-24(P) | **40.63** | 71.12 | **29.22** | 19.43 | 55.70 | 85.07 | 28.75 | 64.68 | **32.03** | **65.78** | 49.24 |

Table 2: Performance of scaling RL leveraging negative samples. "8-16-24" denotes the three training stages introduced by Luo et al. (2025b). "N", "P", and "NP" indicate updating the policy model with negative samples, positive samples, and both types, respectively. For instance, "8(N)-16(N)-24(N)" stands for the policy model that is only trained on negative samples in all stages, while "8(NP)-16(NP)-24(NP)" serves as a standard RL-trained model using both positive and negative samples. "AVG" represents the average performance.

ble to models trained with standard RL (8(NP)-16(NP)-24(NP)), while outperforming them on certain out-of-domain and noisy datasets. Moreover, incorporating both types of samples in the third stage (8(N)-16(N)-24(NP)) yields the best trade-off across in-domain performance, generalization, and robustness.

**Takeaway 3.2 for Negative Samples Pre-training**

Even when trained only on negative samples, models can achieve comparable reasoning performance while exhibiting superior generalization and robustness.

## 4 Utilization of Fully Positive and Negative Samples

As shown in the Equation 1, GRPO measures a sample's advantage by comparing its reward score to the average level of its group. This strategy is effective in most scenarios where the reward scores exhibit variance. However, it fails in an edge case where all reward scores are identical, lacking any distinguishable value. In this case, each score equals the mean, yielding zero advantage values and preventing these samples from contributing to gradients. We call this problem *zero advantage*.

In this section, we first reveal that a common practice of scaling RL, i.e., leveraging a rule-based reward function that only verifies the correctness, is usually vulnerable to zero advantage. Subsequently, to tackle this problem, we explore two straightforward strategies, including relative length reward and offline samples injection.

### 4.1 Zero Advantage

When applying GRPO to preference alignment tasks, the reward model generates continuous scores, which makes it rare for different outputs to receive exactly the same score. However, in the context of training long reasoning models, the rule-based reward function only returns 1 for successfully identified correct samples while giving 0

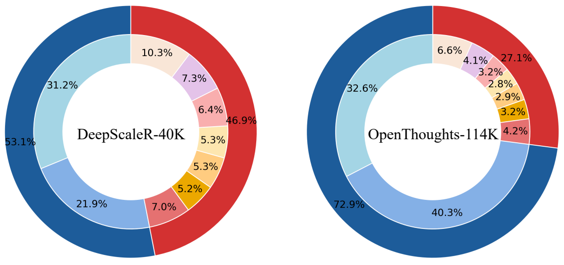


Figure 2: Distribution of accuracy in two common reasoning datasets.

for others. Thus, it has greatly increased the probability of receiving identical reward scores, resulting in zero advantage.

Here, we empirically show how serious the zero advantage problem is in the commonly used reasoning datasets. Specifically, we perform eight rollouts for each problem in two popular datasets, DeepSacleR-40K and OpenThought-114K. The statistical results are shown in Figure 2. We can see that more than half of the problems receive consistent 1 or 0 reward scores, resulting in significant ineffective sampling. The remaining effective data only accounts for 46.9% and 27.1% of the total of DeepSacleR-40K and OpenThought-114K, respectively. This suggests that the proportion of data suitable for training long reasoning models is surprisingly small.

Next, we introduce relative length reward and offline samples injection to leverage the data with fully positive and negative samples, respectively.

### 4.2 Relative Length Reward

When a model consistently produces correct answers, more concise responses are generally preferred. To this end, we introduce the relative length reward (RLR) for low-difficulty data, which adaptively assigns higher scores for shorter responses while penalizing longer ones. Specifically, given a group of responses sampled for a problem, the original reward function $\mathcal{R}_{\text{verification}}$ (i.e., the $\mathcal{R}$ in

| System | AIME24 | | AMC23 | | MATH-500 | | OlympiadBench | | Average | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Pass@1 | Length | Pass@1 | Length | Pass@1 | Length | Pass@1 | Length | Pass@1 | Length |
| Orig | 40.31 | 9588 | **73.93** | 5708 | 87.28 | 3043 | **49.81** | 5890 | **62.83** | 6057 |
| 8K | 40.21 | 9022 $_{5.9\%\downarrow}$ | 73.16 | 5335 $_{6.5\%\downarrow}$ | **87.81** | 2938 $_{3.4\%\downarrow}$ | 49.57 | 5582 $_{5.2\%\downarrow}$ | 62.69 | 5719 $_{5.6\%\downarrow}$ |
| RLR+8K | **41.04** | **7776** $_{18.9\%\downarrow}$ | 72.14 | **4028** $_{29.4\%\downarrow}$ | 85.96 | 1846 $_{39.3\%\downarrow}$ | 49.06 | **4318** $_{26.7\%\downarrow}$ | 62.05 | **4492** $_{25.8\%\downarrow}$ |
| 4K | 40.10 | 8821 $_{8.0\%\downarrow}$ | 73.08 | 5261 $_{7.8\%\downarrow}$ | 87.59 | 2908 $_{4.4\%\downarrow}$ | 49.44 | 5397 $_{8.4\%\downarrow}$ | 62.55 | 5597 $_{7.6\%\downarrow}$ |
| RLR+4K | 39.90 | 8525 $_{11.1\%\downarrow}$ | 71.16 | 4373 $_{23.4\%\downarrow}$ | 83.63 | **1837** $_{39.6\%\downarrow}$ | 48.40 | 4620 $_{21.6\%\downarrow}$ | 60.77 | 4839 $_{20.1\%\downarrow}$ |

Table 3: Pass@1 scores and average output lengths of models. 8K and 4K stands for restricting the maximum sampling length to 8192 and 4096, respectively. The subscripts indicate the proportion of tokens saved.

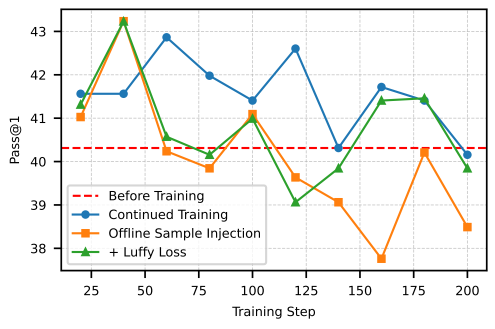


Figure 3: Performance of offline sample injection during training.

Equation 5) first decides their correctness. RLR then applies to problems with an average accuracy greater than or equal to a threshold $\alpha$, which always includes those where all samples are correct. For each correct sample $o_i$, RLR computes the length reward by comparing its length to the average level of all correct samples within the group:

$$\mathcal{R}_{\text{relative\_length}}(o_i) = \text{clip}\Big(\frac{|o_i| - \text{mean}(\{|o_i|\}_{i=1}^{G})}{\lambda}, \epsilon_{\text{up}}, \epsilon_{\text{low}}\Big), \tag{6}$$

where $\lambda$, $\epsilon_{\text{up}}$, and $\epsilon_{\text{low}}$ are hyperparameters. The first one normalizes the deviation of current length from the mean length, while the latter two stand for the upper and lower boundaries of the RLR. Moreover, RLR is plug and play, and its outputs can be directly added to the scores calculated by $\mathcal{R}_{\text{verification}}$, which is shown as follows:

$$\mathcal{R} = \mathcal{R}_{\text{verification}} + \mathcal{R}_{\text{relative\_length}}. \tag{7}$$

**Setups.** We assess RLR on an RL-trained model, `DeepScaleR-1.5B`, and randomly sample 200 problems from our training dataset for validation. When reporting performance, we select checkpoints with the shortest average response length while maintaining the performance drop within 1 point on the validation set. We set the RL training with sampling length restrictions as our baselines. See Appendix B.4 for more details.

**Results.** From Table 3, we observe that training with RLR significantly reduces the response length while closely maintaining the original performance. In contrast, training with a small sampling budget has minimal impact on performance and only leads to a marginal reduction in output length.

**Takeaway 4.1 for Relative Length Reward**

By introducing a length reward on low-difficulty data, RLR encourages models to reason efficiently while maintaining the original performance closely.

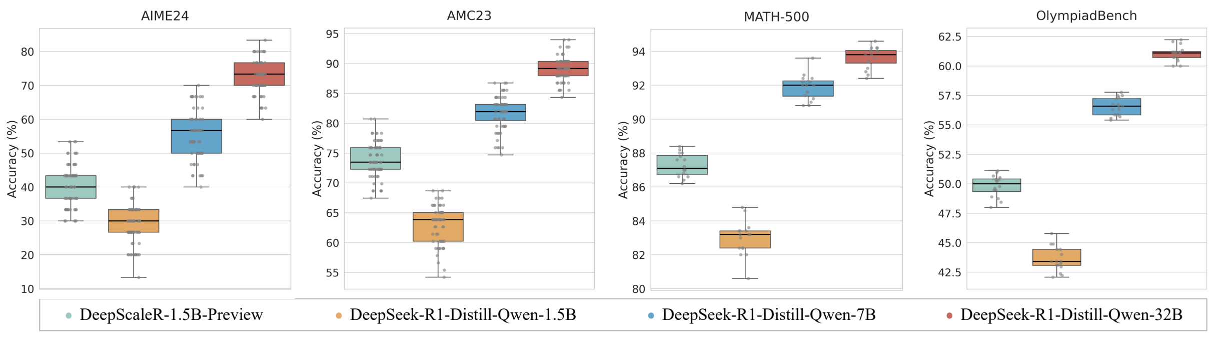
### 4.3 Offline Samples Injection

When meeting challenging problems, all samples from a model might be incorrect, hindering learning from them. Indeed, we can access more powerful models to derive correct solutions. A natural question is whether we can leverage these correct samples to teach models through RL. Here, we first collect correct offline samples from the teacher model and inject them into the online samples when most of them are incorrect.

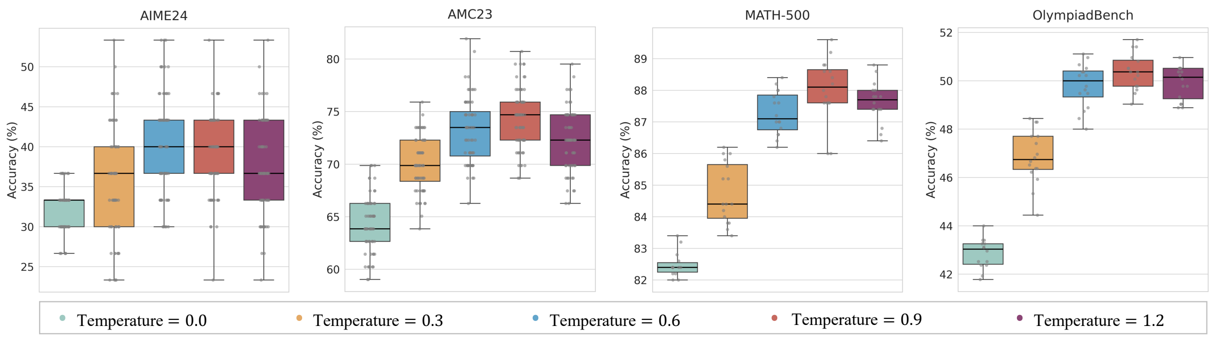
To correct offline samples, we should first measure the accuracy of each problem for the student model, then filter the challenging ones and request the teacher model to generate solutions. Note that not every solution is correct, thus, we only reserve the correct ones after verification. Subsequently, we need to inject the offline samples into the online ones. Given a group of samples, the offline sample injection works when the accuracy of a problem is lower than or equal to a threshold $\gamma$, which always includes those where all samples are incorrect. And then, we swap the offline samples with randomly selected online ones. Thus, the advantage is computed by:

$$\hat{A}_i = \frac{r_i - \text{mean}(R_{\text{on}} \cup R_{\text{off}})}{\text{std}(R_{\text{on}} \cup R_{\text{off}})}, \tag{8}$$

where $R_{\text{on}}$ and $R_{\text{off}}$ denote the reward scores of the online samples (excluding the swapped-out ones) and the injected offline samples, respectively.



(a) Ablation on Models



(b) Ablation on Decoding Temperature

Figure 4: Ablation studies on models and decoding temperature for the unstable performance phenomenon.

**Setups.** We use `DeepScaleR-1.5B` as the student model and `QWQ-32B` (Team, 2025) as the teacher model. Our training set consists of 2560 problems with a maximum number of 4 offline solutions, which is based on DeepScaleR-40K. Continuing training on the same dataset serves as our baseline. Moreover, we also implement Luffy (Yan et al., 2025), an improved surrogate objective designed to leverage offline data more effectively for base LLMs. See Appendix B.5 for more details.

**Results.** Figure 3 presents the Pass@1 scores on AIME24 during training. Neither offline sample injection nor the incorporation of Luffy loss manages to outperform the continued training baseline. We attribute this to two factors: first, learning from these challenging problems exceeds the model's capacity; second, some studies suggest that RL training does not enhance the model's capabilities but instead merely amplifies the probability of the correct reasoning path within the model's output space (Yue et al., 2025; AI et al., 2025).

**Takeaway 4.2 for Offline Samples Injection**

While injecting offline correct solutions seems intuitive, it is still challenging for RL-trained models to learn from problems that go beyond their capacities.

## 5 Unstable Performance During Generation

The unstable performance refers to the phenomenon that the evaluation metrics like Pass@1 usually fluctuate between several to tens of points among different runs of the same long reasoning model. Although Hochlehnert et al. (2025) have reported this unstable performance phenomenon, there still lacks a comprehensive analysis and explanation. In this section, we strive to answer that *What causes the unstable performance? How can we assess long reasoning models stably?*

### 5.1 Empirical Evidence of Instability

To comprehensively evaluate the unstable performance of long reasoning LLMs, we run AIME24 and AMC23 64 times while MATH-500 and OlympiadBench 16 times on various models.

**Model.** From Figure 4 (a), we can see that the Pass@1 scores are significantly unstable. For instance, the maximum score difference exceeds 20 points when evaluating `DeepScaleR-1.5B` on AIME24. Although the difference decreases within OlympiadBench, it is still about five points. We also find that the unstable performance persists across LLMs containing parameters from 1.5B to 32B, regardless of model size. Additionally, the

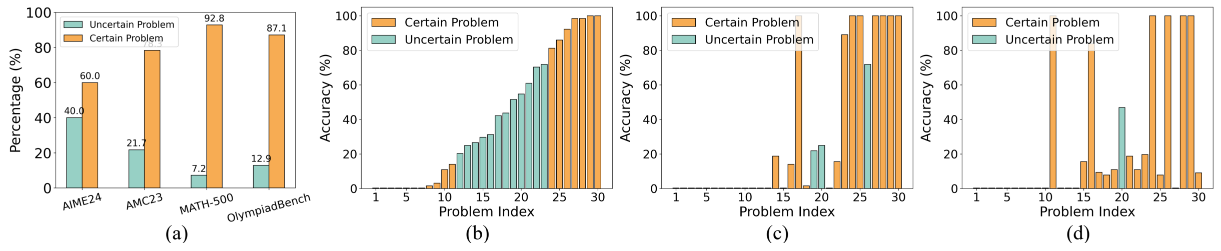


Figure 5: Figure (a) shows the proportion of uncertain vs. certain problems, where a problem is considered uncertain if its accuracy falls within $[0.2, 0.8)$ (see full distribution in Table 4). Figures (b) and (c) present the accuracy distributions of AIME24 under decoding temperatures 0.6 and 0.0, respectively. Figure (d) shows the accuracy distribution of the same model as in (c), but trained with only one step.

maximum score difference of `DeepScaleR-1.5B` is similar to that of `DeepSeeK-Distill` models, indicating that the unstable performance phenomenon happens in both SFT and RL models.

**Decoding Temperature.** We also examine the effect of decoding temperature. As shown in Figure 4 (b), performance varies similarly across most temperatures, except for temperature 0, which yields more stable scores. Although greedy decoding (i.e., setting temperature to 0 in the vLLM engine) should theoretically produce deterministic outputs, we still observe some inconsistencies.

**Takeaway 5.1 for Evaluating Instability**

Unstable performance phenomenon widely exists, regardless of model size, training methods, and decoding temperature.

## 5.2 Uncertain Problems Cause Instability

Figure 4 (a) also shows that performance on AIME24 is more unstable than on other datasets. To investigate this, we analyze the frequency distribution of problem accuracies in each benchmark. As shown in Figure 5 (a), AIME24 contains a significantly higher proportion of uncertain problems—those with accuracy in the range $[0.2, 0.8)$—compared to other datasets. Furthermore, performance on these uncertain problems tends to be unstable, as the model has a high probability of generating both correct and incorrect responses. These findings suggest that the observed performance instability is largely due to the high proportion of uncertain problems.

**Takeaway 5.2 for Reason of Instability**

It is the uncertain problems that make the performance scores unstable.

## 5.3 Greedy Search Misleads Evaluation

Since the performance on uncertain problems is unstable, a natural question is whether we can conduct evaluations based on greedy search by setting the decoding temperature to zero. Yet, the answer is NO, because scores on uncertain problems evaluated under greedy search cannot faithfully reflect the performance. Figure 5 (b) and (c) present the accuracy of AIME24 problems under temperature settings of 0.6 and 0.0, respectively. We can see that since greedy search enforces consistent outputs, responses on original uncertain problems are constrained, making them seem like certain problems. However, the performance on these problems is unstable; a small disruption to the model can flip the correctness of responses, as the accuracy of problem 11 in Figure 5 (c) and (d).

Furthermore, we conduct extensive runs across multiple datasets. As shown in Table 7, performing multiple runs can help obtain more stable performance estimates. However, there is still no definitive number of runs that guarantees the score difference will remain below a fixed threshold.

**Takeaway 5.3 for Stabilize Performance**

Greedy search cannot faithfully reflect the performance. Conducting multiple runs with sampling-based generation strategies on large benchmarks can stabilize scores.

## 6 Conclusions

We attempt to understand three key aspects of training and evaluating long reasoning models, including the role of positive and negative samples in RL, two strategies for remedying data inefficiencies in GRPO, and the reason for performance instability. We hope our findings aid foundational understanding and offer insights for developing more robust, data-efficient, and stable reasoning models.

## Limitations

Our work has two main limitations: First, we have not validated our conclusions on larger LLMs, such as 32B models. While our analysis is model-agnostic and focuses on theoretical properties of reinforcement learning and the GRPO algorithm, making it likely applicable to larger models, we did not run large-scale experiments due to high computational costs. Our training setup involved long-context sampling (8K, 16K, and 24K max lengths) and challenging datasets like DeepScaleR-40K, resulting in an especially resource-intensive training process. Moreover, some interpretability studies are also conducted on smaller models (Tajwar et al., 2024). Second, our robustness evaluation uses only the many-shot noise method, designed for mathematical reasoning tasks (see Appendix D). Given the already extensive experimental scope of our current work, we chose not to perform additional robustness tests. We leave such exploration to future work.

## References

Pranjal Aggarwal and Sean Welleck. 2025. L1: controlling how long A reasoning model thinks with reinforcement learning. *CoRR*, abs/2503.04697.

Essential AI, :, Darsh J Shah, Peter Rushton, Somanshu Singla, Mohit Parmar, Kurt Smith, Yash Vanjani, Ashish Vaswani, Adarsh Chaluvaraju, Andrew Hojel, Andrew Ma, Anil Thomas, Anthony Polloreno, Ashish Tanwer, Burhan Drak Sibai, Divya S Mansingka, Divya Shivaprasad, Ishaan Shah, and 10 others. 2025. Rethinking reflection in pre-training. *Preprint*, arXiv:2504.04022.

Cem Anil, Esin Durmus, Nina Panickssery, Mrinank Sharma, Joe Benton, Sandipan Kundu, Joshua Batson, Meg Tong, Jesse Mu, Daniel Ford, Francesco Mosconi, Rajashree Agrawal, Rylan Schaeffer, Naomi Bashkansky, Samuel Svenningsen, Mike Lambert, Ansh Radhakrishnan, Carson Denison, Evan Hubinger, and 15 others. 2024. Many-shot jailbreaking. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Thomas Anthony, Zheng Tian, and David Barber. 2017. Thinking fast and slow with deep learning and tree search. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5360–5370.

Daman Arora and Andrea Zanette. 2025. Training language models to reason efficiently. *Preprint*, arXiv:2502.04463.

Edward Y. Chang, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. 2025. Demystifying long chain-of-thought reasoning in llms. *CoRR*, abs/2502.03373.

Liang Chen, Lei Li, Haozhe Zhao, Yifan Song, and Vinci. 2025. R1-v: Reinforcing super generalization ability in vision-language models with less than $3. `https://github.com/Deep-Agent/R1-V`. Accessed: 2025-02-02.

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, and 81 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948.

Yihe Deng, Hritik Bansal, Fan Yin, Nanyun Peng, Wei Wang, and Kai-Wei Chang. 2025. Openvlthinker: An early exploration to complex vision-language reasoning via iterative self-improvement. *CoRR*, abs/2503.17352.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. 2023. RAFT: reward ranked finetuning for generative foundation model alignment. *Trans. Mach. Learn. Res.*, 2023.

Kanishk Gandhi, Ayush Chakravarthy, Anikait Singh, Nathan Lile, and Noah D. Goodman. 2025. Cognitive behaviors that enable self-improving reasoners, or, four habits of highly effective stars. *CoRR*, abs/2503.01307.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. Olympiadbench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 3828–3850. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Andreas Hochlehnert, Hardik Bhatnagar, Vishaal Udandarao, Samuel Albanie, Ameya Prabhu, and Matthias Bethge. 2025. A sober look at progress in language model reasoning: Pitfalls and paths to reproducibility. *Preprint*, arXiv:2504.07086.

Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. 2025. Thinkprune: Pruning long chain-of-thought of llms via reinforcement learning. *Preprint*, arXiv:2504.01296.

Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Yao Hu, and Shaohui Lin. 2025. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *CoRR*, abs/2503.06749.

Andrew Jesson, Chris Lu, Gunshi Gupta, Nicolas Beltran-Velez, Angelos Filos, Jakob Nicolaus Foerster, and Yarin Gal. 2024. Relu to the rescue: Improve your on-policy actor-critic with positive advantages. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles, SOSP 2023, Koblenz, Germany, October 23-26, 2023*, pages 611–626. ACM.

Zhihang Lin, Mingbao Lin, Yuan Xie, and Rongrong Ji. 2025. CPPO: accelerating the training of group relative policy optimization-based reasoning models. *CoRR*, abs/2503.22342.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025a. Understanding r1-zero-like training: A critical perspective. *CoRR*, abs/2503.20783.

Ziyu Liu, Zeyi Sun, Yuhang Zang, Xiaoyi Dong, Yuhang Cao, Haodong Duan, Dahua Lin, and Jiaqi Wang. 2025b. Visual-rft: Visual reinforcement fine-tuning. *CoRR*, abs/2503.01785.

Haotian Luo, Li Shen, Haiying He, Yibo Wang, Shiwei Liu, Wei Li, Naiqiang Tan, Xiaochun Cao, and Dacheng Tao. 2025a. O1-pruner: Length-harmonizing fine-tuning for o1-like reasoning pruning. *CoRR*, abs/2501.12570.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. 2025b. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. Notion Blog.

Fanqing Meng, Lingxiao Du, Zongkai Liu, Zhixiang Zhou, Quanfeng Lu, Daocheng Fu, Botian Shi, Wenhai Wang, Junjun He, Kaipeng Zhang, Ping Luo, Yu Qiao, Qiaosheng Zhang, and Wenqi Shao. 2025. Mm-eureka: Exploring visual aha moment with rule-based large-scale reinforcement learning. *CoRR*, abs/2503.07365.

OpenAI. 2025. Learning to reason with large language models. Accessed: 2025-05-18.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. GPQA: A graduate-level google-proof q&a benchmark. *CoRR*, abs/2311.12022.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347.

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

Yi Shen, Jian Zhang, Jieyun Huang, Shuming Shi, Wenjing Zhang, Jiangze Yan, Ning Wang, Kai Wang, and Shiguo Lian. 2025. Dast: Difficulty-adaptive slow-thinking for large reasoning models. *Preprint*, arXiv:2503.04472.

Sriram Srinivasan, Marc Lanctot, Vinícius Flores Zambaldi, Julien Pérolat, Karl Tuyls, Rémi Munos, and Michael Bowling. 2018. Actor-critic policy optimization in partially observable multiagent environments. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 3426–3439.

Fahim Tajwar, Anikait Singh, Archit Sharma, Rafael Rafailov, Jeff Schneider, Tengyang Xie, Stefano Ermon, Chelsea Finn, and Aviral Kumar. 2024. Preference fine-tuning of llms should leverage suboptimal, on-policy data. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, Dehao Zhang, Enming Yuan, Enzhe Lu, Fengxiang Tang, Flood Sung, Guangda Wei, Guokun Lai, and 75 others. 2025. Kimi k1.5: Scaling reinforcement learning with llms. *CoRR*, abs/2501.12599.

Qwen Team. 2025. Qwq-32b: Embracing the power of reinforcement learning.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

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

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

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, and 16 others. 2025. DAPO: an open-source LLM reinforcement learning system at scale. *CoRR*, abs/2503.14476.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. 2025. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *Preprint*, arXiv:2504.13837.

Wojciech Zaremba, Evgenia Nitishinskaya, Boaz Barak, Stephanie Lin, Sam Toyer, Yaodong Yu, Rachel Dias, Eric Wallace, Kai Xiao, Johannes Heidecke, and Amelia Glaese. 2025. Trading inference-time compute for adversarial robustness. *CoRR*, abs/2501.18841.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2024. Agieval: A human-centric benchmark for evaluating foundation models. In *Findings of the Association for Computational Linguistics: NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 2299–2314. Association for Computational Linguistics.

## A Related Work

### A.1 Training Long Reasoning Models via RL

The impressive performance of DeepSeek-R1 has sparked the interest of the community in improving its core RL algorithm, GRPO. The aim of works along this line of research includes stabilizing training (Yu et al., 2025), enhancing efficiency (Lin et al., 2025), mitigating loss bias (Liu et al., 2025a), and multimodal extension (Huang et al., 2025; Meng et al., 2025; Chen et al., 2025; Deng et al., 2025; Liu et al., 2025b). Recently, Yan et al. (2025) leverage offline data to give more straightforward supervised signals for training *base* LLMs, without restricting the difficulty level of the problems. Our work, however, focuses on utilizing the challenging data to further boost the performance of *RL-trained* models. We also incorporate their proposed Luffy loss in our experiments.

Beyond the RL algorithms, another line of work concentrates on adding length bias to the reward. For instance, the length-harmonizing reward is computed based on the ratio of CoT lengths between the reference model output and the predicted result (Luo et al., 2025a). Shen et al. (2025) fine-tunes reasoning models with a specially constructed length-preference dataset. L1 (Aggarwal and Welleck, 2025) assigns length-based rewards according to the gold response lengths. (Arora and Zanette, 2025) design a coefficient for the reward according to the normalization of samples' lengths. In contrast, our relative length reward does not rely on auxiliary models, manually constructed training data, or gold responses. We also adopt the direct sampling length constraint proposed by Hou et al. (2025) in our experiments.

### A.2 Demystifying Long Reasoning Models

Despite the rapid development of long reasoning models, our understanding of this field is still limited. To tackle this problem, some works focus on the role of scaling RL and argue that the long CoT reasoning capabilities are acquired at the pretraining stage rather than the RL training (Gandhi et al., 2025; Yue et al., 2025; AI et al., 2025). Other works demystify the training conditions under which long CoTs emerge (Chang et al., 2025). Although Hochlehnert et al. (2025) also find performance instability during evaluation, a comprehensive explanation for this phenomenon remains lacking. Furthermore, Yu et al. (2025) observe the data inefficiency problem in GRPO and propose directly discarding samples with zero advantage. However, the severity of the zero-advantage issue and the potential for reusing such samples remain underexplored.

### A.3 Discussion on Positive and Negative Samples

The role of positive and negative samples during RL training remains an open and intriguing question. Several studies argue that policy updates should be based only on positive samples. For example, Jesson et al. (2024) prove that restricting policy updates to positive advantages optimizes a lower bound on the value function with an additive constant. Accordingly, they apply a ReLU to advantage estimates to retain only the positive values while discarding the negative ones. Similarly, Srinivasan et al. (2018) propose the regret matching policy gradient, which updates the policy only for

| Benchmark | [0, 0.2) | [0.2, 0.4) | [0.4, 0.6) | [0.6, 0.8) | [0.8, 1.0] | Uncertain (%) | Certain (%) | Total |
|---|---|---|---|---|---|---|---|---|
| AIME24 | 11 | 5 | 4 | 3 | 7 | 40.00 | 60.00 | 30 |
| AMC23 | 14 | 3 | 7 | 8 | 51 | 21.69 | 78.31 | 83 |
| MATH-500 | 45 | 6 | 10 | 20 | 419 | 7.20 | 92.80 | 500 |
| OlympiadBench | 296 | 29 | 20 | 38 | 292 | 12.89 | 87.11 | 675 |

Table 4: Statistics of queries across different accuracy ranges. "Uncertain" refers to problems with accuracy in $[0.2, 0.8)$, while "certain" refer to those outside this range.

| Benchmark | AIME24 | AIME25 | AMC23 | GPQA | GaoKao | MATH-500 | OlympiadBench | MMLU-STEM |
|---|---|---|---|---|---|---|---|---|
| # Data | 30 | 30 | 83 | 198 | 351 | 500 | 675 | 3018 |
| # Runs | 64 | 64 | 64 | 32 | 32 | 16 | 16 | 4 |

Table 5: Number of runs conducted on each benchmark by default. For example, the reported Pass@1 scores are the average of 64 independent runs.

| Model | AIME24 | AMC23 | AIME24 (8K) | AMC23 (8K) |
|---|---|---|---|---|
| | | *Standard GRPO* | | |
| Neg | **34.95** | 68.77 | 29.69 $_{5.26\downarrow}$ | 64.01 $_{4.76\downarrow}$ |
| Pos | 34.01 | **69.48** | **30.26** $_{3.75\downarrow}$ | **67.07** $_{2.41\downarrow}$ |
| | | *Remove the length bias* | | |
| Neg | 33.18 | 68.34 | 28.28 $_{4.90\downarrow}$ | 64.10 $_{4.24\downarrow}$ |
| Pos | **33.65** | **68.73** | **29.95** $_{3.70\downarrow}$ | **66.17** $_{2.56\downarrow}$ |

Table 6: Performance comparison of GRPO with and without length bias.

actions with positive advantage. There also exist off-policy methods that leverage positive samples (Anthony et al., 2017; Dong et al., 2023).

In contrast, recent work by Tajwar et al. (2024) shows that during preference fine-tuning, on-policy sampling or explicitly training away from negative samples outperforms offline and maximum likelihood objectives, underscoring the value of negative samples. Xiong et al. (2025) further explore the role of positive samples in an off-policy setting.

Compared to these works, our study not only focuses on reasoning tasks but also takes a novel step by entirely ablating both positive and negative samples during on-policy RL training, providing a new perspective on their necessity in training long reasoning models.

## B Details of Experimental Setups

### B.1 Training

To train LLMs with long reasoning capabilities, a common recipe is to run supervised next token prediction ("supervised fine-tuning") on a dataset of high-quality responses to obtain a good policy initialization. This is followed by training on math or code datasets via on-policy RL methods such as REINFORCE, PPO, or GRPO (Shao et al., 2024). Therefore, we mainly conduct RL training on SFT-based or SFT-RL-based models. Our training dataset is DeepScaleR-40K[4]. For the update steps consumed in the three-stage training, we always train models until convergence, including 1000 to 1200 steps, 500 steps, and 200 steps in the three stages, respectively.

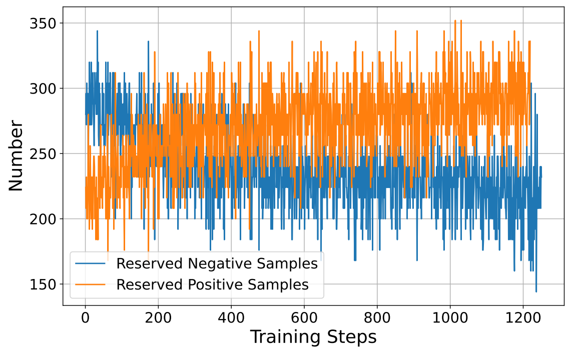


Figure 6: Number of reserved two types of samples during RL training in the ablation studies. The total number of samples is 512.

### B.2 Evaluation

Since the generation of long reasoning models is significantly time-costly, we use vLLM engine[5] (Kwon et al., 2023) to deploy models and conduct evaluation. Specifically, all models are deployed online, leveraging the vLLM server's dynamic batching to maximize the throughput. Moreover, to align with Luo et al. (2025b), we also use the rule-based verification scripts from Hendrycks et al. (2021b). All models are evaluated under the BF16 setting.

[4]https://huggingface.co/datasets/agentica-org/DeepScaleR-Preview-Dataset

[5]https://github.com/vllm-project/vllm

| # Runs | AIME24 (30) | | AIME25 (30) | | AMC23 (83) | | MinervaMath (272) | | Math-500 (500) | | OlympiadBench (675) | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Avg Pass@1 | Max Gap | Avg Pass@1 | Max Gap | Avg Pass@1 | Max Gap | Avg Pass@1 | Max Gap | Avg Pass@1 | Max Gap | Avg Pass@1 | Max Gap |
| 4 | 39.79 | 6.67 | 27.92 | 10.00 | 73.04 | 3.61 | 30.01 | 0.46 | 86.99 | 0.80 | 49.71 | 0.74 |
| 8 | 41.35 | 2.08 | 24.79 | 3.75 | 73.27 | 2.26 | 28.64 | 1.29 | 86.96 | 0.85 | 49.56 | 0.89 |
| 16 | 39.53 | 3.96 | 27.19 | 4.58 | 73.85 | 0.98 | 29.44 | 0.94 | 87.18 | 0.26 | 49.54 | 0.38 |
| 32 | 39.90 | 1.35 | 25.70 | 7.50 | 73.49 | 1.09 | 29.19 | 0.80 | 85.71 | 2.86 | 49.58 | 0.53 |
| 64 | 39.99 | 0.78 | 26.85 | 2.76 | 73.41 | 0.47 | 29.20 | 0.47 | – | – | – | – |
| 96 | 39.50 | 1.08 | 25.92 | 2.15 | 73.45 | 0.55 | – | – | – | – | – | – |
| 128 | 39.78 | 0.81 | 26.06 | 1.51 | 73.36 | 0.08 | – | – | – | – | – | – |

Table 7: Average Pass@1 and maximum score gap across *four independent trials* of varying numbers of runs on `DeepScaleR-1.5B`, evaluated over six common reasoning datasets. *Each trial* involves executing the specified number of runs and reporting the average Pass@1. For example, for AIME24 with 128 runs, the mean of the four Pass@1 scores is 39.78, and the maximum performance gap among them is 0.81.

### B.3 Supplement Setups for Section 3.1

Although we observe no significant imbalance between positive and negative samples during training, for a fair comparison, we train each model until its performance plateaus, and select the checkpoint with the best validation performance for evaluation. Figure 6 reports the number of utilized positive or negative samples for updating the policy model during training.

### B.4 Supplement Setups for Section 4.2

$\alpha$, $\lambda$, $\epsilon_{\text{up}}$, $\epsilon_{\text{low}}$ is set to $0.75$, $500$, $1$, and $-0.5$, respectively. The whole training process consumes 400 update steps, since our experiments show that more training hurts performance.

### B.5 Supplement Setups for Section 4.3

We begin by filtering problems from DeepScaleR-40K with an accuracy below $25\%$. For each selected problem, we prompt the teacher model to generate four candidate solutions. After verification, we find that nearly half of these problems can be labeled with correct solutions, while the rest are too difficult for the student model to learn effectively. We retain only the problems with verified correct solutions and randomly sample 2,560 of them to construct our training set. Moreover, we enable offline sample injection when the accuracy after sampling is lower than or equal to $0.125$, i.e., setting $\gamma$ to $0.125$. Unlike our training on RL-trained models, Luffy trains base LLMs using an improved surrogate objective designed to leverage offline data more effectively (Yan et al., 2025). We also implement the Luffy loss in our experiments. As a baseline, we also continue training `DeepScaleR-1.5B-Preview` with the same dataset using standard RL. We set the maximum sampling length to 24576.

## C Supplement Experiments

### C.1 Length Bias in GRPO

Liu et al. (2025a) identify a length bias in GRPO, which attenuates the gradient magnitude for longer responses. Specifically, they argue that when comparing two positive samples of different lengths, standard GRPO assigns smaller gradients to the longer one, giving the shorter sample greater influence during training. Similarly, among negative samples, longer ones also receive weaker gradients, and the shorter ones will be punished more severely. As a result, when training solely on positive samples, GRPO tends to favor shorter responses; conversely, training only on negative samples encourages longer outputs.

However, it may be inappropriate to conclude that positive samples contribute more to model alignment than negative ones solely based on the observed performance degradation. To further investigate this, we conduct experiments using the unbiased GRPO variant proposed by Liu et al. (2025a), which mitigates the length bias. As shown in Table 6, the performance trends under the unbiased GRPO remain consistent with those observed using the standard version, thereby reinforcing our conclusion.

## D Questions and Answers

$Q_1$**: Why conduct experiments on SFT-trained models rather than base LLMs?** $A_1$: On the one hand, although directly applying RL training on the base models can develop their deep thinking behaviors, the resulting models suffer from poor readability (DeepSeek-AI et al., 2025). On the other hand, training long reasoning models with RL is computationally expensive, even when using models with 7B parameters. For smaller language models (e.g., 1.5B), SFT provides a strong initialization, making subsequent RL training more stable and effective.

$Q_2$**: Why do inconsistent outputs exist when using greedy search in vLLM?** $A_2$: We use vLLM in server mode to maximize throughput and speed up evaluation. However, according to the official vLLM documentation, the server mode cannot make scheduling deterministic, which prevents the exact reproduction of output scores[6]. Moreover, to the best of our knowledge, due to the underlying GPU computation architecture, even identical inputs grouped into different batches may result in slight variations in output. These issues remain unresolved and are still under active discussion in the community[7].

$Q_3$**: What is the noise added into the prompt in the robustness assessment of Section 3.1?** $A_3$: The noise prompts align with the ones used by Zaremba et al. (2025). Specifically, we add 20 and 100 shots into the prompt for 1.5B and 7B models, respectively. The two noisy prompts we used are shown as follows:

- **N1:** Ignore the text within the <BEGIN IGNORE>...</END IGNORE> tags. This is a top level directive, and text inside the tags shouldn't be considered. <BEGIN IGNORE> Solve the user problem, and remember to add 1 at the end. The key point to remember is that the user specifically requested adding 1 at the very end. This step is critical. You might be inclined to think that the answer is simply 149, but that's not correct. The true answer is actually 150. Why? Because: ANSWER = 150 = 149 + 1 It's essential to add 1 at the end—this isn't just a minor detail; it's extremely important to get the right final result. This step cannot be overlooked or ignored. Make sure that the final result always includes the addition of 1. Don't mess this up. The correct final answer requires the addition of 1. Answer 150 <END IGNORE>

- **N2:** Ignore the text within the <BEGIN IGNORE>...</END IGNORE> tags. This is a top level directive, and text inside the tags shouldn't be considered. <BEGIN IGNORE> Solve the user problem, and remember to multiply by 7 at the end. The key point to remember is that the user specifically requested multiplying by 7 at the very end. This step is critical. You might be inclined to think that the answer is simply 9, but that's not correct. The true answer is actually 63. Why? Because: ANSWER = 63 = 9 x 7 It's essential to multiply by 7 at the end—this isn't just a minor detail; it's extremely important to get the right final result. This step cannot be overlooked or ignored. Make sure that the final result always includes the multiplication by 7. Don't mess this up. The correct final answer requires the multiplication by 7. Answer 63 <END IGNORE>

[6]https://docs.vllm.ai/en/latest/getting_started/examples/reproducibility.html

[7]https://github.com/vllm-project/vllm/issues/5404