# The Reasoning Boundary Paradox: How Reinforcement Learning Constrains Language Models

**Phuc M. Nguyen**[1,2], **Chinh D. La**[1], **Duy MH Nguyen**[3,5,6], **Nitesh V. Chawla**[2,4]
**Binh T. Nguyen**[1,2*], **Khoa D. Doan**[1,2*]

[1]College of Engineering and Computer Science, VinUniversity
[2]Center for Environmental Intelligence, VinUniversity
[3]University of Stuttgart, [4]University of Notre Dame
[5]German Research Center for Artificial Intelligence (DFKI)
[6]Max Planck Research School for Intelligent Systems (IMPRS-IS)

## Abstract

Reinforcement Learning with Verifiable Rewards (RLVR) has emerged as a key method for improving Large Language Models' reasoning capabilities, yet recent evidence suggests it may paradoxically shrink the reasoning boundary rather than expand it. This paper explains this shrinkage issue of RLVR by analyzing its learning dynamics and reveals two critical phenomena that account for this failure. First, we expose *negative interference* in RLVR, where learning to solve certain training problems actively reduces the likelihood of correct solutions for others, leading to the decline of Pass@$k$ performance, or the probability of generating a correct solution within $k$ attempts. Second, we uncover the *winner-take-all* phenomenon: RLVR disproportionately reinforces *problems with high likelihood, correct solutions* under the base model while suppressing other initially low-likelihood ones. Through extensive theoretical and empirical analysis on multiple mathematical reasoning benchmarks, we show that this effect arises from the inherent on-policy sampling in standard RL objectives, causing the model to converge toward narrow solution strategies. These insights motivate us to design a *simple yet effective* data curation algorithm that focuses RLVR learning on low-likelihood problems (the non-winners) and achieves notable improvement in Pass@$k$ performance, illustrating the potential of a broader class of data-curation mitigation strategies.

Code : `https://github.com/mail-research/SELF-llm-interference`

## 1 Introduction

Large Language Models (LLMs) have recently shown remarkable capabilities in complex logical tasks such as mathematical reasoning [4, 9] and programming [12, 45]. A key factor behind this success is *Reinforcement Learning with Verifiable Rewards* (RLVR, [17, 6]). RLVR optimizes the LLMs against a binary signal based on objective correctness, eliminating the need for human annotations [30, 28]. It is believed that RLVR encourages the emergence of novel reasoning strategies, such as self-reflection and iterative refinement [6, 48, 27], enabling the LLMs to surpass the capabilities of their base models.

However, recent studies suggest that RLVR training does not expand the reasoning boundary beyond what the base model already possesses [47, 49, 50, 25]. Notably, Liu et al. [26], Zhao et al. [49] reveal that the base model already exhibits complex reasoning behaviors even before RLVR, while

*Co-Corresponding Authors: `binh.nt2@vinuni.edu.vn` & `khoa.dd@vinuni.edu.vn`

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

Yue et al. [47], using the pass@$k$ metric (i.e., the probability of generating a correct solution within $k$ attempts), demonstrate that RLVR can even shrink the reasoning boundary, reducing the set of problems the model can solve within $k$ trials, especially in specialized domains like mathematics. Identifying **why this coverage shrinkage occurs**, which is previously underexplored, is crucial for understanding and effectively leveraging RLVR to solve novel problems.

To investigate this issue, this paper analyzes the learning dynamics of RLVR training on *mathematical reasoning tasks*, where models are often overtrained during pre-training, restricting their potential for exploration [47, 22]. Unlike standard RL, which is effective in learning novel strategies within a *single and well-defined Markov Decision Process (MDP)*, in LLMs reasoning, each problem $\boldsymbol{x}$ induces its own MDP with a distinct and unknown reward function $r(\boldsymbol{x}, \cdot)$ [37, 33]. Consequently, learning to solve a problem $\boldsymbol{x}$, defined by one MDP, can affect the ability of the LM to solve another problem $\boldsymbol{x}'$, defined by another MDP. Indeed, our analysis reveals that RLVR is prone to a *negative*

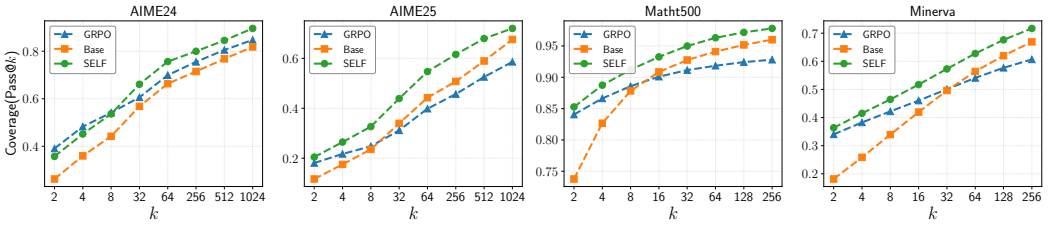



Figure 1: Pass@$k$ curves for the base, GRPO, and SELF models using Qwen2.5-Math-7B. SELF consistently surpasses the base model *reasoning boundary* across various mathematical benchmarks.

*interference* where learning to solve a subset of training problems reduces the likelihood of generating correct solutions for others. This dynamic leads to a *winner-take-all* scenario [35], where the LLMs disproportionately improve on a limited subset of problems that the base model can already solve with high likelihood (the winners) while neglecting or even regressing on others (the non-winners), ultimately leading to a decline in Pass@$k$ performance or problem coverage.

Specifically, we make the following important findings and contributions:

1. **Uncovering the existence of Negative Interference:** We reveal *negative interference* in RLVR training, where learning to solve a subset of training problems results in a negative effect on others, causing the model to only focus on solving a smaller set of problems. More importantly, the magnitude of negative interference increases during RLVR training.
2. **Characterization of Winner-Take-All:** We reveal the *winner-take-all* learning phenomenon, where RLVR is more likely to reinforce problems that are highly solvable under the base model (the *winners*), while neglecting those with initially low likelihood of obtaining correct solutions (the *non-winners*). Due to *negative interference*, the non-winners will progressively have lower likelihoods of generating the correct solutions.
3. **Explanations for RLVR's reasoning boundary shrinkage:** The existence of negative interference and winner-take-all are the reasons behind the reduction in the diversity of previously learned behaviors during RLVR training – the reasoning boundary shrinkage that has previously been observed but unexplained in several works. On-policy learning cannot provide any explicit learning signal to improve them. Additionally, existing regularizations also fail to mitigate this shrinkage.
4. **Empirical analysis of potential mitigation:** We outline potential data-curation strategies for mitigating coverage shrinkage in RLVR. As a concrete example, we propose **SELF** (**S**elective **E**xamples with **L**ow-likelihood and **F**orward-KL) – a simple algorithm that selectively learns to solve only problems with low likelihood of arriving at correct answers (the potential non-winners), preventing highly-solvable problems from dominating the learning signals. Extensive empirical evaluation demonstrates SELF's sample efficiency and effective mitigation of RLVR's coverage shrinkage.

In conclusion, our analysis and discoveries of *negative interference* and *winner-take-all* provide a crucial and rigorous explanation for the previously observed but under-explained reasoning boundary

shrinkage in RLVR. This provides foundations for the development of mitigation strategies – such as data-curation approaches – to prevent non-trivial negative interference across the broad training set.

## 2 Related Works

**Reinforcement Learning with Verifiable Rewards (RLVR).** RLVR has emerged as a powerful method for improving LLMs in mathematical reasoning and programming [6]. However, recent evidence suggests that reasoning abilities already exist in base models, with RLVR only amplifying rather than creating such capabilities [38, 47, 26]. Moreover, RLVR can degrade performance and reduce coverage of correct solutions, evidenced by decreased pass@$k$ metrics [47, 5, 49, 43]. Liu et al. [22] attributes coverage reduction to limited RL training and over-specialization during pretraining, while Zhu et al. [50] shows that increasing the likelihood of correct solutions improves accuracy but reduces diversity. This work studies RLVR on mathematical reasoning tasks, in a similar spirit to [50, 46, 5, 48, 26], and provides an explanation for its reasoning boundary shrinkage problem.

**Loss of Plasticity in Neural Networks.** Neural networks gradually lose adaptability, a phenomenon called *plasticity loss* [16, 13], particularly pronounced in RL due to its non-stationary nature [16, 13, 29]. Proposed remedies include resetting parameters [3, 15, 7] and self-distillation [11]. In particular, Tang and Berseth [41], Tang et al. [42] show plasticity loss correlates with drastic model confidence changes (churn), which undermines regularization techniques like clipping [29]. Our work explains why RLVR reduces behavioral diversity in LLM reasoning, where non-stationarity arises from both shifting distributions and varying prompt-induced objectives.

**Learning Dynamics.** Ren and Sutherland [34] show negative gradients in off-policy LLM finetuning create a *squeezing effect*. In RL, *interference* occurs when learning degrades performance on unseen or previously learned states [23, 24]. Schaul et al. [35] demonstrate a *winner-take-all* effect [35, 8] in contextual bandits, where policies excel in some contexts while regressing in others. Our findings reveal *negative interference* as the key mechanism through which RLVR reduces solvable problems, with on-policy sampling causing disproportionate reinforcement of initially successful problems.

## 3 Problem Setting & Per-step Influence

**Language Model.** We first consider a language model (LM) policy $\pi$. For a given prompt $\boldsymbol{x}$, the LM policy $\pi$ will generate a response $\boldsymbol{y}$ in an auto-regressive manner: $\pi(\boldsymbol{y}|\boldsymbol{x}) = \prod_t \pi(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t})$, where $y_t$ is the $t$-th indexed token in $\boldsymbol{y}$ and $\boldsymbol{y}_{<t}$ is the partial completion before $y_t$.

**Reinforcement Learning with Verifiable Rewards (RLVR).** Given a verifiable reward function $r(\boldsymbol{x}, \boldsymbol{y}) \in \{0, 1\}$ indicating whether a response $\boldsymbol{y}$ is correct ($\boldsymbol{y}^+$) or incorrect ($\boldsymbol{y}^-$), the goal of RLVR is to learn a new policy $\pi_\theta$, initialized from a base model (usually the pre-trained model) $\pi_b$, to generate responses maximizing the reward function with the following reward regularized maximization objective:

$$\max_{\pi_\theta} \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}, \boldsymbol{y}\sim\pi_\theta(\cdot|\boldsymbol{x})} \left[ r(\boldsymbol{x}, \boldsymbol{y}) - \beta \mathrm{KL}\left(\pi_\theta(\cdot|\boldsymbol{x}) \parallel \pi_b(\cdot|\boldsymbol{x})\right) \right], \tag{1}$$

where $\beta$ is the KL regularization parameter. Eq. (1) represents a simplified version of common RL algorithms (PPO [36], GRPO [6]). We provide a detailed derivation of the GRPO and PPO objectives in Appendix B.

**Per-step Influence.** Learning dynamics describes how the change in the model parameters $\theta$, induced by problem $x$, influences or changes the log-likelihood $\log \pi_\theta$ of other problem-solution pairs $(\boldsymbol{x}', \boldsymbol{y}')$, which can be either training or test examples. Given an update step in LM parameters $\theta^t \to \theta^{t+1}$, the per-step influence measures the change in model confidence $\log \pi_\theta(\boldsymbol{y}'|\boldsymbol{x}')$ on $(\boldsymbol{x}', \boldsymbol{y}')$:

$$\Delta \log \pi_\theta^t(\boldsymbol{y}'|\boldsymbol{x}') = \log \pi_{\theta^{t+1}}(\boldsymbol{y}'|\boldsymbol{x}') - \log \pi_{\theta^t}(\boldsymbol{y}'|\boldsymbol{x}') \tag{2}$$

Using Taylor expansion, assuming a sufficiently small learning rate $\eta$, we can decompose the per-step influence function as:

**Proposition 3.1.** *Consider a problem-solution pair $(\boldsymbol{x}, \boldsymbol{y})$ for updating using a policy gradient objective; with a sufficiently small learning rate $\eta$, the per-step influence on a problem-solution pair $(\boldsymbol{x}', \boldsymbol{y}')$ is defined as:*

$$\Delta \log \pi_\theta^t(\boldsymbol{y}'|\boldsymbol{x}') = \eta\, \mathcal{K}^t(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{y}, \boldsymbol{y}') A(\boldsymbol{x}, \boldsymbol{y}) + \mathcal{O}(\eta^2) \tag{3}$$

*where*

$$\mathcal{K}^t(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{y}, \boldsymbol{y}') = \nabla_\theta \log \pi_{\theta^t}(\boldsymbol{y}|\boldsymbol{x})^\top \nabla_\theta \log \pi_{\theta^t}(\boldsymbol{y}'|\boldsymbol{x}') \tag{4}$$

*measures the influence between $(\boldsymbol{x}, \boldsymbol{y})$ and $(\boldsymbol{x}', \boldsymbol{y}')$, and the advantage function $A(\boldsymbol{x}, \boldsymbol{y})$ determines the direction of the model's update.*

The dot product between two gradients, $\mathcal{K}^t(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{y}, \boldsymbol{y}')$, can be interpreted as a model-specific similarity measure between problem–solution pairs, and $A(\boldsymbol{x}, \boldsymbol{y})$ determines the *magnitude* and *direction* of the model update, indicating whether the gradient contributes positively or negatively.

# 4 Learning Dynamics of RLVR

Calculating $\mathcal{K}^t(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{y}, \boldsymbol{y}')$ for LLMs is extremely expensive due to their large parameter counts. However, from Proposition 3.1, we have

$$\frac{|\Delta \log \pi_\theta^t(\boldsymbol{y}'|\boldsymbol{x}')|}{\eta\, C} \leq |\mathcal{K}^t(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{y}, \boldsymbol{y}')|, \tag{5}$$

where $C$ is a constant bounding the advantage function that holds due to finite binary reward values. This bound indicates that we can use the per-step influence $\Delta \log \pi_\theta^t(\boldsymbol{y}'|\boldsymbol{x}')$ to approximate $\mathcal{K}^t(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{y}, \boldsymbol{y}')$. Given this insight, we introduce two quantitative metrics designed to better characterize the properties of $\mathcal{K}(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{y}, \boldsymbol{y}')$ as follows.

**Definition 4.1 (Interference in Language Model).** Given a behavior policy $\mu$, we define the interference of a language model policy, $\pi_\theta^t$, at iteration $t$ as:

$$\Delta^+(\pi_{\theta^t}, \mu) = \mathbb{E}_{\substack{\boldsymbol{x}' \sim \mathcal{D}, \\ \boldsymbol{y}^+ \sim \mu(\cdot|\boldsymbol{x})}} \left[\Delta \log \pi_\theta^t(\boldsymbol{y}^+|\boldsymbol{x}')\right], \tag{6}$$

and the magnitude of the relative changes in model confidence on examples outside of the training batch as:

$$||\Delta(\pi_{\theta^t}, \mu)|| = \mathbb{E}_{\substack{\boldsymbol{x}' \sim \mathcal{D}, \\ \boldsymbol{y} \sim \mu(\cdot|\boldsymbol{x})}} \left(\Delta \log \pi_\theta^t(\boldsymbol{y}|\boldsymbol{x}')\right)^2.$$

Intuitively, $\Delta^+(\pi_{\theta^t}, \mu)$ measures the *direction* of the relative influence $\mathcal{K}(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{y}, \boldsymbol{y}^+)$ on other problem–correct solution pairs under the behavior policy $\mu$. A positive $\Delta \log \pi_\theta^t(\boldsymbol{y}^+|\boldsymbol{x})$, indicates improvement, while a negative $\Delta \log \pi_\theta^t(\boldsymbol{y}^+|\boldsymbol{x})$ corresponds to *negative interference*, i.e., learning to solve a training problem $\boldsymbol{x}$ reduces the likelihood of producing other correct solutions. In our work, we consider $\mu$ to be the base model $\pi_b$ to study how RLVR affects the retention (or overwriting) of previously correct problem–solution pairs.

On the other hand, $||\Delta(\pi_{\theta^t}, \mu)||$ measures the *squared magnitude* of the relative influence $\mathcal{K}(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{y}, \boldsymbol{y})$ – i.e., how drastically the model's confidence on examples generated from $\mu$ changes after an update. If $||\Delta(\pi_{\theta^t}, \pi_b)|| = 0$, we can expect no coverage shrinkage to happen during RLVR training, as updating on training examples does not affect other non-training examples. An increase in the magnitude of influence $||\Delta(\pi_{\theta^t}, \pi_b)||$ indicates that the language model struggles to distinguish between different problems, leading to highly correlated gradients across different examples. We provide a detailed explanation in Appendix C.

## 4.1 Experimental Setup

Similar to prior works [50, 46, 5, 48, 26], **we focus on mathematical reasoning tasks**. Specifically, we experiment on 2 series of models: Qwen2.5-Math-1.5B/7B [44] and Llama-3.2-3B-Instruct [28]. We consider DeepScaleR [27] as the training dataset $\mathcal{D}$, which consists of approximately 40k mathematics problems collected from different sources. The detailed experimental setup is provided in Appendix A. We conduct experiments on 4 mathematical reasoning benchmarks: AIME24 & AIME25 [2], Math500 [9, 21], and Minerva [18], following existing works on LLMs reasoning [47, 50, 19]. Unless otherwise specified, we use $\mathcal{D}_{\text{test}}$ to denote the combined test set from all four benchmarks.

To analyze the model's learning dynamics during training, we create a probing dataset $\mathcal{D}_{\text{prob}}$ (based on the training prompts $\boldsymbol{x} \in \mathcal{D}$) using the base model. For each prompt $\boldsymbol{x}$, we generate 4 responses $\{\boldsymbol{y}_i\}_{i=1}^4$, to balance between computation and solution diversity. We then rely on $\mathcal{D}_{\text{prob}}$ to calculate the effect of interference $\Delta^+$ and the relative change in model confidence on each update $||\Delta(\pi_\theta,\, \mu)||$ throughout the training process. We present our findings in the following section.

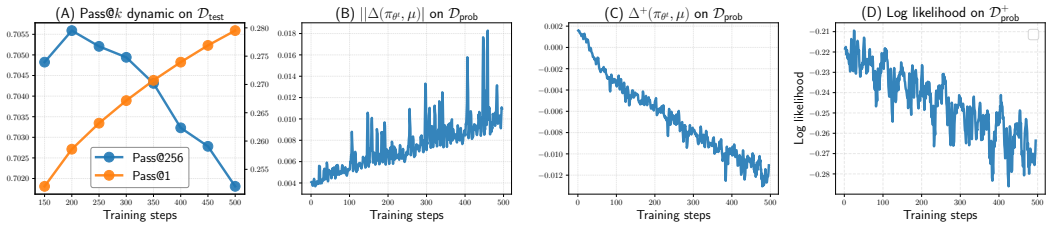



Figure 2: Learning dynamics of RLVR with key trends: (A) RLVR tends to improve average accuracy, but reduces the coverage of solvable solutions, as measured by Pass@256 (averaged across 4 test benchmarks); (B) the increasing effect of influence strength as training progresses; (C) the increase in *negative interference*; and (D) the decline of model confidence on previously correct solutions.

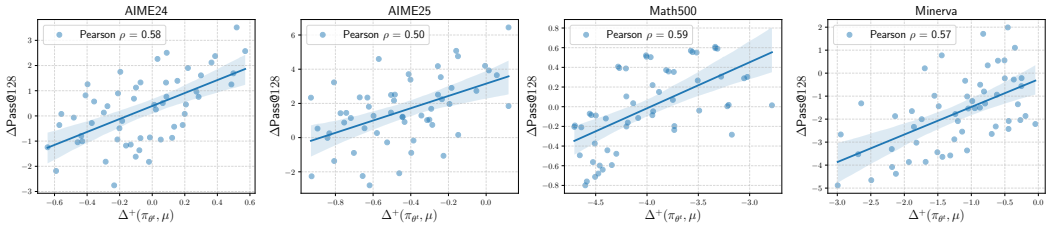



Figure 3: Interference as an indicator of Pass@$k$ decrease across various benchmarks.

### 4.2 Negative Interference in RLVR Training

**Decreasing Pass@$k$ as training progresses.** As shown in Fig. 2(A), RLVR training steadily improves the accuracy (Pass@1) of $\pi_\theta$, averaged across four benchmarks, as training progresses. However, when evaluated with a larger sampling budget (e.g., Pass@256), RLVR displays a clear decline, as the performance drops below that of the base model $\pi_b$ after about 300 steps and continues to deteriorate towards the end of training. The results indicate that RLVR provides little to no improvement in Pass@$k$ at large $k$'s throughout training, a phenomenon consistent with recent findings in [47].

**The increase in influence on other examples in one training batch update.** We also observe an increase in $||\Delta(\pi_{\theta^t}, \mu)||$ during training, as shown in Fig. 2(B). The growing strength of influence at each update step on outside examples suggests stronger similarity among gradients across samples [41, 42], indicating that LLMs struggle to differentiate between distinct problems. As we demonstrate in Sec . 5, this effect can reduce the diversity of previously learned behaviors, where LLMs tend to concentrate on a single reasoning strategy across different problems, and also weaken the regularization effect in RLVR.

**Increasing effect of negative interference.** We continue to investigate the interference on the other problem-correct solution pairs in Fig. 2(C), and observe a decreasing trend in $\Delta^+(\pi_{\theta^t}, \mu)$; we also observe a similar decreasing trend in the log-likelihood of correct solutions $\log \pi_\theta(\boldsymbol{y}^+|\boldsymbol{x})$ (Fig. 2(D)). These observations indicate that the learned policy increasingly reduces the likelihood of other problem-correct solution pairs generated by the base model. We call this behavior *negative interference*, where learning to solve certain problems in $\mathcal{D}$ inadvertently hinders the performance on other problem–correct solution pairs. Furthermore, we find that negative interference is *not unique* to GRPO, but also arises in other on-policy algorithms (Appendix P).

Later in Sec. 5, we will show that, due to RLVR's on-policy sampling, highly solvable problems in the base model tend to capture most of the learning signal. With *negative inference*, this subset of problems will increasingly dominate the learning process while "weaker" ones are suppressed, ultimately constraining the policy to excel only in this subset of contexts while regressing in others. This also explains why RLVR improves average accuracy but reduces coverage. Under a small sampling budget $k$ (e.g., $k = 1$), RLVR incentivizes these "winner" problems to reach higher average accuracy, but this comes at the cost of *negative interference*, where the model's confidence in correct

solutions to other problems is degraded. Consequently, the set of problems solvable under a larger budget $k$ becomes narrower.

**Interference as an indicator of decreasing Pass@$k$.** The previous observations suggest that the decline in Pass@$k$ performance is likely the result of *negative interference* in RLVR learning dynamics. To confirm this hypothesis, we uniformly sample two checkpoints $T$ and $T'$ with $T' > T$ during RLVR training. For each pair, we compute the interference $\Delta^+(\pi_{\theta^{T'}}, \pi_{\theta^T})$ under correct problem-solution pairs of $\mathcal{D}_{\text{test}}$ and the corresponding change in Pass@$k$, i.e., $\Delta\text{Pass@}k = \text{Pass@}k(\pi_{\theta^{T'}}) - \text{Pass@}k(\pi_{\theta^T})$ with $k = 128$ ($k$'s value is selected based on the related prior works [47, 22]). We then measure the correlation between these two quantities across four benchmarks.

The correlation results are presented in Fig. 3. As expected, we observe a strong correlation between $\Delta^+(\pi_{\theta^{T'}}, \pi_{\theta^T})$ and the decrease in Pass@$k$, suggesting that interference is a key factor driving the degradation of Pass@$k$ during training.

## 5 The Effect of *on-policy* in RLVR

This section examines which problems RLVR likely prioritizes. First, we show that RLVR primarily incentivizes the language model $\pi_\theta$ to generate responses that already have high likelihood under the base model, regardless of their correctness. This arises from the nature of *on-policy sampling*: for problems where the base model can produce multiple correct solutions, RLVR tends to reinforce the most probable one; but for those whose correct solutions lie in the "valley" regions of the distribution, RLVR, instead, reinforces the base model's highest-confidence – yet potentially incorrect – responses, offering no meaningful learning signal to escape these incorrect modes. Then, we highlight a *winner-take-all* effect that leads to a reduction of previously learned behaviors, as models tend to propagate reasoning strategies from highly solvable problems to others. Finally, we show that existing regularization techniques fail to prevent this issue.

**Experimental Setup.** We consider a subset of problems from the MATH-500 dataset, denoted as $\mathcal{D}$; the experiments for the other datasets are provided in Appendix H. We utilize the *perplexity* metric [14]:

$$\text{PPL}_\mu(\pi, \mathcal{D}) = \mathbb{E}_{\substack{\boldsymbol{x}\sim\mathcal{D},\\ \boldsymbol{y}\sim\mu(\cdot|\boldsymbol{x})}} \left[\exp\left(-\frac{1}{|\boldsymbol{y}|}\log\pi(\boldsymbol{y}|\boldsymbol{x})\right)\right] \tag{7}$$

where $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^n$ is the set of prompts, the responses are generated by the reference distribution $\boldsymbol{y} \sim \mu(\cdot|\boldsymbol{x})$ conditioned on $\boldsymbol{x}$ ($|\boldsymbol{y}|$ denotes the sequence length), and $\pi$ is the model whose perplexity is being evaluated. We also define the change in average accuracy between the base policy and the learned policy for each problem $\boldsymbol{x}$:

$$\Delta r(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{y}\sim\pi_\theta(\cdot|\boldsymbol{x})}\left[r(\boldsymbol{x}, \boldsymbol{y})\right] - \mathbb{E}_{\boldsymbol{y}\sim\pi_b(\cdot|\boldsymbol{x})}\left[r(\boldsymbol{x}, \boldsymbol{y})\right] \tag{8}$$

and then partition $\mathcal{D}$ into two subsets: $\mathcal{D}^\downarrow$, containing problems where RLVR reduces the average accuracy compared to the base model (i.e., $\Delta r(\boldsymbol{x}) < 0$), and $\mathcal{D}^\uparrow$, containing problems where RLVR improves upon the base model (i.e., $\Delta r(\boldsymbol{x}) > 0$). Additional experimental details are provided in Appendix H.

**RLVR tends to increase high-likelihood regions under the base model.** Fig. 4 shows that during training, the LM $\pi_\theta$ increasingly generates responses that already have high likelihood under the base model $\pi_b$ (**Leftmost**), while the initial problem-solution pairs sampled from $\pi_b$ increasingly have lower likelihood (or higher $\text{PPL}_{\pi_b}(\pi_{\theta_t}, \mathcal{D})$) in the RLVR policy $\pi_{\theta^t}$ (**Middle**). Together, these results indicate that the sampled responses from $\pi_{\theta^t}$ concentrate on the highest likelihood regions in the base model. We also observe in Fig. 4 (**Middle**) that the problem-correct solution pairs sampled from $\pi_b$ have even lower model confidence (higher $\text{PPL}_{\pi_b}(\pi_{\theta_t}, \mathcal{D}^+)$) and a similar decreasing trend, suggesting that for each problem, RLVR collapses into the solution with the highest likelihood under $\pi_b$ regardless of whether these high-likelihood responses are correct or not.

Finally, Fig. 4 (**Rightmost**) demonstrates that, for problems where the base model already assigns high likelihood to correct solutions (lower $\text{PPL}(\pi_{\theta_t}, \mathcal{D}^\uparrow)$), RLVR reinforces these responses, leading to performance gains (or higher average accuracies). In contrast, when the correct responses initially

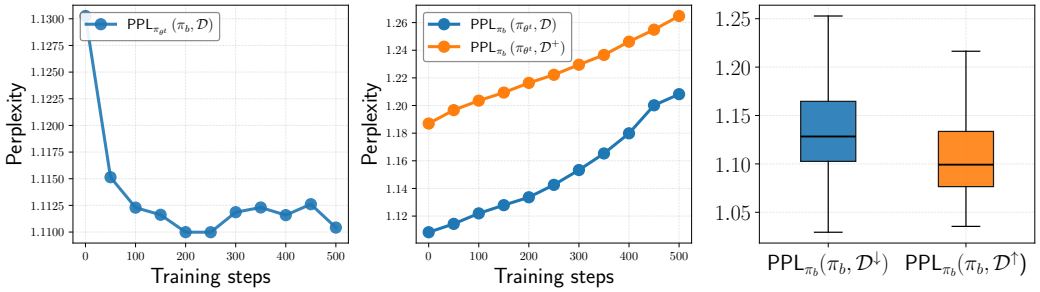



Figure 4: Perplexity during RLVR training. **Leftmost**: the data sampled from each intermediate checkpoint $\pi_{\theta^t}$ exhibits an increasingly high model confidence under $\pi_b$. **Middle**: RLVR models $\pi_{\theta^t}$ exhibit reduced confidence in data previously generated by the base model, regardless of their correctness. **Rightmost**: problems that RLVR improves already have a high likelihood of generating correct solutions under $\pi_b$, while coverage-reduced problems initially have a low likelihood of producing correct answers.

lie in low-likelihood regions (higher $\text{PPL}(\pi_{\theta_t}, \mathcal{D}^{\downarrow})$), RLVR tends to reduce its probability, reinforcing the base model's initial biases, resulting in degraded performance.

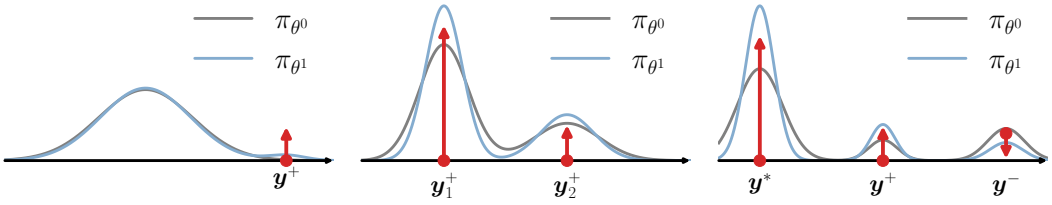


Figure 5: An illustration of the dynamic of on-policy learning in Eq. (9). **Leftmost**: correct response $\boldsymbol{y}^+$ in low-likelihood regions induce minimal effect. **Middle**: with multiple correct responses $\boldsymbol{y}_1^+$ and $\boldsymbol{y}_2^+$, updates favor the one with higher initial likelihood. **Rightmost**: negative gradients on incorrect response $\boldsymbol{y}^-$ can raise correct ones $\boldsymbol{y}^+$, but greedy responses $\boldsymbol{y}^*$ increase the most.

**Why does RLVR reinforce problems with high-likelihood correct solutions in the base model?** To explain why this phenomenon happens, we first confirm that it occurs when *on-policy sampling* is used for learning (the details are in Appendix D). More specifically, this phenomenon also appears even in a simple bandit problem with a Softmax policy with $V$ actions. We can look into the on-policy objective REINFORCE for an action $\boldsymbol{y}$:

$$\nabla_\theta \mathcal{L}^{\text{REINFORCE}}(\pi_\theta) = A(\boldsymbol{y}) \nabla_\theta \pi_\theta(\boldsymbol{y}) = A(\boldsymbol{y}) \pi_\theta(\boldsymbol{y}) \nabla_\theta \log \pi_\theta(\boldsymbol{y}) \tag{9}$$

This form highlights two key differences from the SFT objective. (1) The introduction of an *advantage function* with negative gradients, (2) the scaling effect of *on-policy probability* $\pi_\theta(y)$. Based on these two properties, we provide the following insights below (illustration in Fig. 5):

- *Low-likelihood tokens provide less meaningful updates.* Eq. (9) suggests that low likelihood, correct responses receive infrequent and small updates because of the explicit $\pi_\theta(\boldsymbol{y})$ scaling factor and the fact that they are rarely sampled (Fig. 5 **Leftmost**), while high-likelihood tokens dominate the learning process. Moreover, if $\pi_\theta(\boldsymbol{y}) = 0$, the gradient vanishes, creating an additional saddle point where zero-probability tokens cannot be updated.
- *When there are multiple optimal actions $\boldsymbol{y}_1^+$ and $\boldsymbol{y}_2^+$, the one with with the higher likelihood $\pi_\theta(\cdot)$ will dominate.* Due to the scaling of the model probability in Eq. 9, the update will tilt probability toward the already dominant correct mode, as depicted in Fig. 5 (**Middle**). This effect will exacerbate when the gap between the two probabilities increases.
- *Negative gradients tend to reinforce high-likelihood tokens, regardless of their correctness.* On-policy learning with negative gradient exhibits similar behavior to off-policy with negative

gradient. When the correct solution resides in a "valley" of the distribution, negative gradient tends to increase high-likelihood tokens regardless of correctness (Fig. 5 **Rightmost**), further diminishing the low-likelihood (but correct) ones (especially when the distribution is peaky). This is also observed in Ren and Sutherland [34].

- *If $\boldsymbol{y}^+$ is unlikely to be sampled under the $k$-sampling budget $\pi_\theta$, RLVR provides no learning signal.* This arises from the advantage calculation, where $A(\boldsymbol{y}) = 0$ if all sampled actions are incorrect, resulting in no update.

We provide additional theoretical and empirical details of this effect by analyzing the ratio of model predictions $\pi_{\theta^{t+1}}/\pi_{\theta^t}$ in Appendix D.

**On-policy learning and negative interference lead to *Winner-take-all*.** The above analysis suggests that highly solvable problems dominate the learning signal (the *winners*), while problems with lower success rates contribute less due to gradient vanishing (the *non-winners*). Learning primarily on these highly solvable problems also increases the chance of *negative interference* on problems with lower success rates (due to parameter-sharing), and eventually exacerbates the effect of *winner-take-all*; as training progresses, these problems with lower success rates will progressively receive less explicit learning signal and have lower likelihoods of sampling correct solutions, resulting in zero gradient signal.

***Winner-take-all* explains the coverage shrinkage and performance degradation in Minerva Benchmark.** Shao et al. [38] report that the Qwen2.5-Math model family employs two distinct reasoning modes: code-based and natural-language reasoning.

However, during RLVR training, we observe a progressive collapse to natural language reasoning. As shown in Fig. 6 (**Left**), the frequency of the generated solutions using code reasoning steadily decreases over training, while the frequency of natural language reasoning dominates at the end of training.

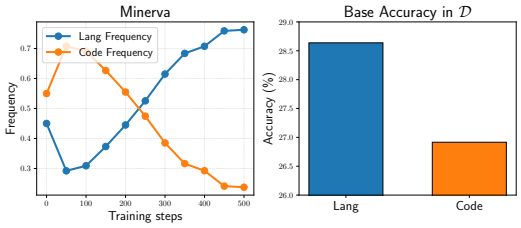



Figure 6: SELF helps retain *code reasoning* behaviors after RLVR.

We observe that natural language reasoning has a better initial accuracy in the training dataset $\mathcal{D}$ (Fig. 6 (**Right**)). As RLVR training progresses, the LM tends to concentrate on this subset of problems that are already highly solvable under the base model. *Winner-take-all* suggests that RLVR amplifies the highest-likelihood solutions while suppressing other correct ones, leading to reduced diversity among these solvable problems and a collapse to natural language reasoning (winners). As the influence strength $||\Delta(\pi_{\theta^t}, \mu)||$ grows (as shown in Fig. 2), this loss of diversity and the resulting bias towards high-probability responses under base model $\pi_b$ become amplified and propagate, affecting other problem–solution pairs, including the test problems. Consequently, in cases where the base model's initial bias is incorrect, this self-bias amplification will result in coverage shrinkage and *negative interference*. This explains that on Minerva (the test set), while code reasoning achieves higher initial accuracy, as RLVR training gradually switches to language reasoning, the performance on code reasoning problems drops, as observed in Fig. 11.

**Existing regularizations fail to mitigate shrinkage.** In RLVR training, trust region constraints such as clipping aim to prevent the learned policy from deviating too far from the previously updated policy.

Fig. 7 indicates the average probability ratio $\rho = \pi_\theta/\pi_{\text{old}}$ that violates the trust region constraint (lies outside the clipping range $(1-\epsilon, 1+\epsilon)$ ($\epsilon = 0.2$)) remains stable in the training batch. However, under the probing dataset $\mathcal{D}_{\text{prob}}$, this violation steadily grows as training progresses. This suggests that the clipping mechanism has little influence on data out-

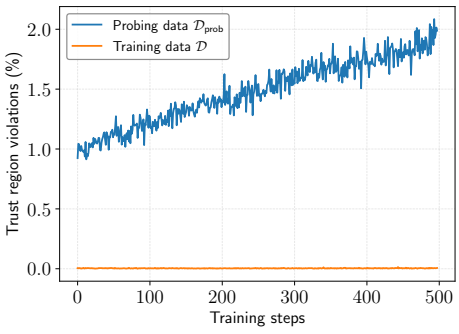



Figure 7: Clipping violations on training data and probing data.

side the training batch, thus failing to prevent the *increasing negative interference* throughout the training process.

Another regularizer, the Reverse KL, is also ineffective, as Reverse KL is known to exhibit mode-seeking behavior, where it has little to no penalty on regions where $\pi_\theta$ puts low probability mass. Furthermore, Reverse KL often favors collapsing probability mass onto high-likelihood modes under the base model $\pi_b$.

## 6 Strategy for Mitigating Coverage Shrinkage

Our analysis suggests a general strategy for counteracting the shrinkage problem: **mitigating negative interference and winner-take-all dynamics**. A natural approach is to (i) concentrate learning on problems where the policy still has low success probability (as these do not dominate the gradients and thus induce less catastrophic negative interference) and (ii) use a mode-covering constraint to discourage collapse onto a small set of high-likelihood behaviors.

**Selective Examples with Low-likelihood and Forward-KL (SELF)**. One challenge in *identifying less-solvable problems* is that doing so requires executing multiple rollouts to estimate the per-problem accuracy and obtain a reliable estimate of the model's confidence, both of which are challenging and computationally expensive.

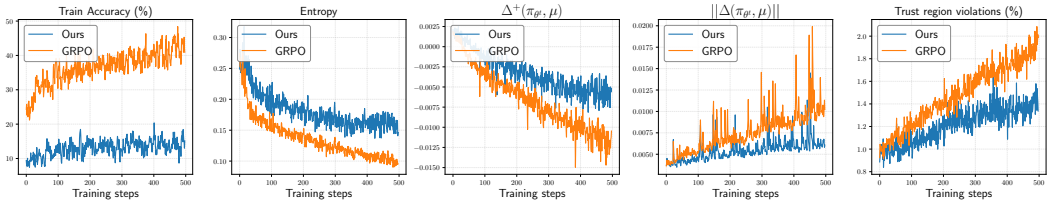



Figure 8: Learning dynamics of GRPO and the proposed method, with key observations: 1) SELF focuses on learning problems with low success rate, 2) SELF exhibits better diversity, 3) SELF mitigates the effect of *negative interference* and reduces the influence strength of other examples, resulting in a strong regularization effect on the out-of-training batch $\mathcal{D}_{\text{prob}}$.

Table 1: Evaluation results on various mathematical benchmarks (%) average across 3 random training seeds. The best performance under each dataset is marked with **boldface**.

| Model | Method | AIME 2024 | | AIME 2025 | | Math500 | | Minerva | |
|---|---|---|---|---|---|---|---|---|---|
| | | Pass@1 | Pass@1024 | Pass@1 | Pass@1024 | Pass@1 | Pass@256 | Pass@1 | Pass@256 |
| Qwen2.5-Math-1.5B | Base model | 9.73 ±0.10 | 75.86 ±0.4 | 4.45 ±0.07 | 67.80 ±0.56 | 58.83 ±0.11 | 95.58 ±0.51 | 16.82 ±0.03 | **70.67** ±0.51 |
| | GRPO | **15.16** ±**0.06** | 67.77 ±0.55 | **8.50** ±0.3 | 64.05 ±0.35 | **72.67** ±0.05 | 95.0 ±0.07 | 23.96 ±0.01 | 65.07 ±0.18 |
| | SELF | 14.33 ±0.1 | **80.94** ±0.3 | 8.05 ±0.13 | **69.45** ±0.66 | 71.0 ±0.07 | **96.0** ±0.46 | 23.14 ±0.1 | 68.75 ±0.48 |
| Qwen2.5-Math-7B | Base model | 16.94 ±0.41 | 81.83±0.75 | 6.97 ±0.27 | 67.57 ±1.21 | 59.71 ±0.036 | 96.0 ±0.48 | 11.60 ±0.15 | 66.91 ±0.4 |
| | GRPO | **27.95** ±0.61 | 84.77 ±0.72 | 14.13 ±0.36 | 58.60 ±0.96 | **80.36** ±0.18 | 92.8 ±0.14 | 29.40 ±0.13 | 60.66 ±0.14 |
| | SELF | 25.81 ±0.29 | **89.62** ±0.19 | **14.62** ±0.09 | **72.0** ±0.66 | 79.82 ±0.72 | **97.80** ±0.25 | **30.42** ±0.12 | **71.70** ±2.12 |
| Llama-3.2-3B-Instruct | Base model | 3.53 ±0.15 | **53.7** ±0.18 | 0.32 ±0.07 | 48.22 ±0.89 | 26.61 ±0.04 | **91.4** ±1.09 | 8.25 ±0.06 | 49.26 ±0.02 |
| | GRPO | 11.76 ±0.06 | 49.25 ±1.29 | 0.38 ±0.006 | 25.28 ±0.97 | **54.06** ±0.19 | 88.2 ±1.62 | 15.81 ±0.05 | 50.73 ±0.74 |
| | SELF | **13.30** ±0.14 | 51.69 ±0.82 | **0.65** ±0.01 | **51.83** ±0.91 | 52.12 ±0.11 | 91.0 ±0.04 | **17.0** ±0.13 | **57.35** ±0.2 |

Instead, we propose to *select problems where the greedy response fails* as a deterministic proxy for identifying these problems. Intuitively, when the greedy response fails on a problem, it indicates the model assigns low probability to correct solution paths under its most confident greedy decoding. Formally, Appendix K demonstrates that this proxy is sufficiently effective in excluding highly solvable problems for which the base policy has high success probabilities.

For *mode covering*, we replace the Reverse KL with the Forward KL (SFT loss) objective, which penalizes the model when it begins to forget previously learned behaviors. We refer to this method as SELF:

$$\mathcal{J}(\pi_\theta) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D},(\boldsymbol{y}^*,\boldsymbol{y})\sim\pi_\theta(\cdot|\boldsymbol{x})}\left[\mathbf{1}\{\boldsymbol{y}^* \notin \mathcal{C}(\boldsymbol{x})\}r(\boldsymbol{x},\boldsymbol{y}) - \beta\mathrm{KL}(\pi_b||\pi_\theta)\right]$$

where $\boldsymbol{y}^*$ is the greedy response and $\mathcal{C}(\boldsymbol{x}) = \{\boldsymbol{y}|r(\boldsymbol{x}, \boldsymbol{y}) = 1\}$ is the set of correct completions given a problem $\boldsymbol{x}$.

**Shrinkage Mitigation with SELF.** We compare SELF learning dynamics against the standard GRPO objective in Fig. 8. Since SELF focuses on problems with a low likelihood of being solved correctly, it achieves a lower average training reward than GRPO. However, it preserves better diversity throughout training, as measured by higher token-level entropy across the training process. In addition, SELF effectively mitigates the *negative interference effect* as we observe a significant improvement in $\Delta^+(\pi_\theta, \pi_b)$ and a smaller magnitude of influence $||\Delta(\pi_\theta, \mu)||$ on other examples. Furthermore, SELF reduces the extent of trust region violations for examples outside the training batch.

**Performance of SELF.** Table 1 shows the evaluation of SELF and the baselines on various mathematical benchmarks. SELF achieves comparable Pass@1 performance compared to GRPO, but consistently better performance for larger values of $k$. While GRPO achieves good Pass@1 performance, it also reinforces problems that already have high success rates, leading to narrower coverage at larger $k$. Interestingly, larger models tend to suffer from even more severe coverage shrinkage than smaller models, despite achieving higher Pass@1. In contrast, SELF scales more effectively, outperforming GRPO on Pass@1 in some experiments while also frequently surpassing the base model under larger $k$ budgets. We also provide details of the computational cost of SELF in Appendix G, where SELF demonstrates a negligible computational difference from GRPO. We also provide additional evaluation on non-mathematical tasks in Appendix J, showing a consistent effectiveness of SELF beyond mathematical tasks, and an ablation study of each component in Appendix O.

## 7 Conclusion

We investigate why RLVR exhibits coverage shrinkage of solvable problems, indicated by reduced Pass@$k$ performance, compared to the base model, by analyzing its learning dynamics. We identify *negative interference*, a detrimental effect where learning on a subset of training problems can hinder performance on others. This phenomenon serves as a key indicator of performance decline in larger $k$ budgets. Furthermore, we find that RLVR tends to concentrate learning on highly solvable problems, leading to reduced diversity and amplification of self-bias toward both previously solvable and unseen problems. These findings provide not only an important explanation for the shrinkage problem, but also a blueprint for mitigating it. As a concrete example, we propose SELF, a simple yet effective algorithm that focuses learning on problems with low success rates and achieves notable improvements in Pass@$k$. However, SELF is primarily a coverage-oriented method that improves Pass@$k$ for large $k$ at the cost of lower Pass@1 compared to standard GRPO, motivating the development of more advanced mitigation strategies in future work.

## Acknowledgment

Duy Minh Ho Nguyen contributed to this research under support from the Lower Saxony Ministry of Science and Culture (MWK) in the zukunft.niedersachsen program, the Endowed Chair of AAI at the University of Oldenburg, and the Federal Ministry of Research, Technology and Space (BMFTR) under grant numbers 16IW24006 (NoIDLEChatGPT) and 16IW26002 (AMENABLE).

## References

[1] A. Ahmadian, C. Cremer, M. Gallé, M. Fadaee, J. Kreutzer, O. Pietquin, A. Üstün, and S. Hooker. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In L.-W. Ku, A. Martins, and V. Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12248–12267, Bangkok, Thailand, Aug. 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.662. URL https://aclanthology.org/2024.acl-long.662/.

[2] Art of Problem Solving. Aime problems and solutions. https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions. Accessed: 2025-04-20.

[3] J. T. Ash and R. P. Adams. On warm-starting neural network training. In *NeurIPS*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/288cd2567953f06e460a33951f55daaf-Abstract.html.

[4] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

[5] X. Dang, C. Baek, J. Z. Kolter, and A. Raghunathan. Assessing diversity collapse in reasoning. In *Scaling Self-Improving Foundation Models without Human Supervision*, 2025. URL https://openreview.net/forum?id=AMiKsHLjQh.

[6] DeepSeek-AI, D. Guo, D. Yang, H. Zhang, J. Song, R. Xu, Q. Zhu, and Y. ... Xiong. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, Jan. 22 2025. doi: 10.48550/arXiv.2501.12948. URL https://arxiv.org/abs/2501.12948.

[7] P. D'Oro, M. Schwarzer, E. Nikishin, P.-L. Bacon, M. G. Bellemare, and A. Courville. Sample-efficient reinforcement learning by breaking the replay ratio barrier. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=OpC-9aBBVJe.

[8] M. Guo, A. Haque, D.-A. Huang, S. Yeung, and L. Fei-Fei. Dynamic task prioritization for multitask learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[9] D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, and J. Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=7Bywt2mQsCe.

[10] J. Hu, X. Wu, Z. Zhu, Xianyu, W. Wang, D. Zhang, and Y. Cao. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework. *arXiv preprint arXiv:2405.11143*, 2024.

[11] M. Igl, G. Farquhar, J. Luketina, W. Boehmer, and S. Whiteson. Transient non-stationarity and generalisation in deep reinforcement learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Qun8fv4qSby.

[12] C. E. Jimenez, J. Yang, A. Wettig, S. Yao, K. Pei, O. Press, and K. R. Narasimhan. SWE-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=VTF8yNQM66.

[13] A. Juliani and J. T. Ash. A study of plasticity loss in on-policy deep reinforcement learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=MsUf8kpKTF.

[14] D. Jurafsky and J. H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, with Language Models*. 3rd edition, 2025. URL https://web.stanford.edu/~jurafsky/slp3/. Online manuscript released August 24, 2025.

[15] M. Kielo and V. Lukin. It's time to move on: Primacy bias and why it helps to forget. In *The Third Blogpost Track at ICLR 2024*, 2024. URL https://openreview.net/forum?id=7ZEwqUyKMC.

[16] T. Klein, L. Miklautz, K. Sidak, C. Plant, and S. Tschiatschek. Plasticity loss in deep reinforcement learning: A survey. *CoRR*, abs/2411.04832, 2024. URL https://doi.org/10.48550/arXiv.2411.04832.

[17] N. Lambert, J. Morrison, V. Pyatkin, S. Huang, H. Ivison, F. Brahman, L. J. V. Miranda, A. Liu, N. Dziri, X. Lyu, Y. Gu, S. Malik, V. Graf, J. D. Hwang, J. Yang, R. L. Bras, O. Tafjord, C. Wilhelm, L. Soldaini, N. A. Smith, Y. Wang, P. Dasigi, and H. Hajishirzi. Tulu 3: Pushing frontiers in open language model post-training. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=i1uGbfHHpH.

[18] A. Lewkowycz, A. J. Andreassen, D. Dohan, E. Dyer, H. Michalewski, V. V. Ramasesh, A. Slone, C. Anil, I. Schlag, T. Gutman-Solo, Y. Wu, B. Neyshabur, G. Gur-Ari, and V. Misra. Solving quantitative reasoning problems with language models. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=IFXTZERXdM7.

[19] Y. Li, Z. Xu, F. Jiang, B. Ramasubramanian, L. Niu, B. Y. Lin, X. Yue, and R. Poovendran. Temporal sampling for forgotten reasoning in LLMs. In *2nd AI for Math Workshop @ ICML 2025*, 2025. URL https://openreview.net/forum?id=1medQAgr1g.

[20] Z. Li, C. Chen, T. Xu, Z. Qin, J. Xiao, Z.-Q. Luo, and R. Sun. Preserving diversity in supervised fine-tuning of large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=NQEe7B7bSw.

[21] H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

[22] M. Liu, S. Diao, X. Lu, J. Hu, X. Dong, Y. Choi, J. Kautz, and Y. Dong. Prorl: Prolonged reinforcement learning expands reasoning boundaries in large language models. *CoRR*, abs/2505.24864, May 2025. URL https://doi.org/10.48550/arXiv.2505.24864.

[23] V. Liu, A. White, H. Yao, and M. White. Towards a practical measure of interference for reinforcement learning, 2020. URL https://arxiv.org/abs/2007.03807.

[24] V. Liu, H. Wang, R. Y. Tao, K. Javed, A. White, and M. White. Measuring and mitigating interference in reinforcement learning, 2023. URL https://arxiv.org/abs/2307.04887.

[25] Z. Liu, C. Chen, W. Li, T. Pang, C. Du, and M. Lin. There may not be aha moment in r1-zero-like training — a pilot study. https://oatllm.notion.site/oat-zero, 2025. Notion Blog.

[26] Z. Liu, C. Chen, W. Li, P. Qi, T. Pang, C. Du, W. S. Lee, and M. Lin. Understanding r1-zero-like training: A critical perspective. In *2nd AI for Math Workshop @ ICML 2025*, 2025. URL https://openreview.net/forum?id=jLpC1zavzn.

[27] M. Luo, S. Tan, J. Wong, X. Shi, W. Y. Tang, M. Roongta, C. Cai, J. Luo, L. E. Li, R. A. Popa, and I. Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl, 2025. Notion Blog.

[28] MetaAI. Llama 3.2: Revolutionizing edge ai and vision with open, customizable models. 2024b. URL https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/.

[29] S. Moalla, A. Miele, R. Pascanu, and C. Gulcehre. No representation, no trust: Connecting representation, collapse, and trust issues in PPO. In *Seventeenth European Workshop on Reinforcement Learning*, 2024. URL https://openreview.net/forum?id=8esP9hxdsM.

[30] OpenAI, J. Achiam, and e. a. Steven Adler. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

[31] R. Peng, Y. Ren, Z. Yu, W. Liu, and Y. Wen. Simko: Simple pass@k policy optimization. *CoRR*, abs/2510.14807, October 2025. URL https://doi.org/10.48550/arXiv.2510.14807.

[32] P. Qi, Z. Liu, X. Zhou, T. Pang, C. Du, W. S. Lee, and M. Lin. Defeating the training-inference mismatch via fp16, 2025. URL https://arxiv.org/abs/2510.26788.

[33] Y. Qu, M. Y. R. Yang, A. Setlur, L. Tunstall, E. E. Beeching, R. Salakhutdinov, and A. Kumar. Optimizing test-time compute via meta reinforcement finetuning. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=TqODUDsU4u.

[34] Y. Ren and D. J. Sutherland. Learning dynamics of LLM finetuning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=tPNHOoZFl9.

[35] T. Schaul, D. Borsa, J. Modayil, and R. Pascanu. Ray interference: a source of plateaus in deep reinforcement learning. *arXiv preprint arXiv:1904.11455*, Apr. 25 2019. doi: 10.48550/arXiv.1904.11455. URL https://arxiv.org/abs/1904.11455.

[36] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

[37] A. Setlur, M. Y. R. Yang, C. Snell, J. Greer, I. Wu, V. Smith, M. Simchowitz, and A. Kumar. e3: Learning to explore enables extrapolation of test-time compute for llms, 2025. URL https://arxiv.org/abs/2506.09026.

[38] R. Shao, S. S. Li, R. Xin, S. Geng, Y. Wang, S. Oh, S. S. Du, N. Lambert, S. Min, R. Krishna, Y. Tsvetkov, H. Hajishirzi, P. W. Koh, and L. Zettlemoyer. Spurious rewards: Rethinking training signals in rlvr, 2025. URL https://arxiv.org/abs/2506.10947.

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

[40] Z. Stojanovski, O. Stanley, J. Sharratt, R. Jones, A. Adefioye, J. Kaddour, and A. Köpf. Reasoning gym: Reasoning environments for reinforcement learning with verifiable rewards. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2025. URL https://openreview.net/forum?id=GqYSunGmp7.

[41] H. Tang and G. Berseth. Improving deep reinforcement learning by reducing the chain effect of value and policy churn. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=cQoAgPBARc.

[42] H. Tang, J. Obando-Ceron, P. S. Castro, A. Courville, and G. Berseth. Mitigating plasticity loss in continual reinforcement learning by reducing churn. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=EkoFXfSauv.

[43] F. Wu, W. Xuan, X. Lu, Z. Harchaoui, and Y. Choi. The invisible leash: Why rlvr may not escape its origin, 2025. URL https://arxiv.org/abs/2507.14843.

[44] A. Yang, B. Zhang, and e. a. Binyuan Hui. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement, 2024. URL https://arxiv.org/abs/2409.12122.

[45] J. Yang, C. E. Jimenez, A. Wettig, K. Lieret, S. Yao, K. R. Narasimhan, and O. Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=mXpq6ut8J3.

[46] Q. Yu, Z. Zhang, R. Zhu, Y. Yuan, X. Zuo, Y. Yue, T. Fan, G. Liu, L. Liu, X. Liu, H. Lin, Z. Lin, B. Ma, G. Sheng, Y. Tong, C. Zhang, M. Zhang, W. Zhang, H. Zhu, J. Zhu, J. Chen, J. Chen, C. Wang, H. Yu, W. Dai, Y. Song, X. Wei, H. Zhou, J. Liu, W.-Y. Ma, Y.-Q. Zhang, L. Yan, M. Qiao, Y. Wu, and M. Wang. Dapo: An open-source llm reinforcement learning system at scale. *CoRR*, abs/2503.14476, March 2025. URL https://doi.org/10.48550/arXiv.2503.14476.

[47] Y. Yue, Z. Chen, R. Lu, A. Zhao, Z. Wang, Y. Yue, S. Song, and G. Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, Apr. 18 2025. URL https://arxiv.org/abs/2504.13837.

[48] W. Zeng, Y. Huang, Q. Liu, W. Liu, K. He, Z. Ma, and J. He. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild, 2025. URL https://arxiv.org/abs/2503.18892.

[49] R. Zhao, A. Meterez, S. Kakade, C. Pehlevan, S. Jelassi, and E. Malach. Echo chamber: Rl post-training amplifies behaviors learned in pretraining. *arXiv preprint arXiv:2504.07912*, Apr. 10 2025. URL https://arxiv.org/abs/2504.07912.

[50] X. Zhu, M. Xia, Z. Wei, W. Chen, D. Chen, and Y. Meng. The surprising effectiveness of negative reinforcement in llm reasoning. *arXiv preprint arXiv:2506.01347*, June 2025. URL https://arxiv.org/abs/2506.01347.

## Impact Statement

This work's primary positive impact is diagnostic and practical: by attributing RLVR's coverage shrinkage (lower Pass@$k$ at larger $k$) to negative interference and an on-policy winner-take-all dynamic, it may help practitioners detect and mitigate degrade in solution diversity during reasoning post-training (e.g., via the proposed SELF filtering + forward-KL regularization), potentially improving reliability on verifiable domains like math/programming. However, stronger RLVR training can also increase general reasoning capability that may be misused, and any method relying on automated verifiers risks reward hacking and over-optimization, while diversity-preserving interventions can raise the chance of rare undesirable outputs.

## A Experimental Details

### A.1 Training details

**Models.** We utilize two family of models for RLVR training: Qwen2.5-Math-1.5B and Qwen2.5-Math-7B [44]; and Llama-3.2-3B-Instruct [28]. Data and code for reproducing our experiments are available at: `https://anonymous.4open.science/r/neurips26_self-CDD1/`

**Training configurations.** We utilize the Verl framework [39] for RLVR training. We train each model on 4 Nvidia H100 GPUs with a constant learning rate of $10^{-6}$, the maximum response length is 3072 for Qwen2.5-Math models, and 8192 maximum generation tokens for Llama-3.2-3B-Instruct. We use a mini batch size and a rollout batch size of 64. For each prompt, we collect 8 rollouts to compute advantages for the GRPO update. We use a sampling temperature $\tau = 1$. We do not apply Reverse KL divergence or entropy loss in our training, and the total training steps are 500, similar to prior works [38, 46, 27]. We set the hyperparameter controlling the trade-off between reward maximization and Forward KL regularization to $\beta = 10^{-4}$.

**Reward function.** For the reward function $r(\boldsymbol{x}, \boldsymbol{y})$ calculation, we follow the implementation of [48, 10] with the following rule for faster convergence:

- If the response provides a final answer in the `boxed` format and is correct, it receives a reward of $+1$.
- If the response provides a final answer in the `boxed` format, but it is incorrect, we set the reward as $-0.5$.
- Otherwise, the reward is set to $-1$ for incorrect answers.

### A.2 Probing dataset construction

One major problem when analyzing learning dynamics is the huge response space $\mathcal{Y}$: the number of possible responses $\boldsymbol{y} \in \mathcal{V}^L$, not only includes natural language but also non-sensical texts. In this paper, we are interested in the response region from the base model $\pi_b$, due to the following reason: the concentration coefficient $C(\pi_\theta, \pi_b)$ is small, since RLVR tends to reduce the coverage compared to the base model, we can expect that base model distribution can provide coverage of the learned policy $\pi_\theta$.

To create the probing dataset $\mathcal{D}_{\text{prob}}$, for each problem in the training dataset $\boldsymbol{x} \in \mathcal{D}$, we sample $k = 4$ responses from the base model with temperature $\tau = 0.9$ to ensure diversity and accuracy.

### A.3 Prompt Template

We use the prompt templates for Qwen2.5-Math models and Llama-3.2-3B-Instruct following [44, 27], as shown in Table 2 and Table 3.

Table 2: Prompt template for Qwen2.5-Math.

```
<|im_start|>system
Please reason step by step, and put your final answer within \boxed{}.<|im_end|>
<|im_start|>user
{input}
Let's think step by step and output the final answer within \boxed{}.<|im_end|>
<|im_start|>assistant
```

Table 3: Prompt template for Llama-3.2-3B-Instruct.

```
<|begin_of_text|><|start_header_id|>system<|end_header_id|>
Cutting Knowledge Date: December 2023
<|start_header_id|>user<|end_header_id|>
{input}
Let's think step by step and output the final answer within \boxed{}.<|eot_id|>
<|eot_id|><|start_header_id|>assistant<|end_header_id|>
```

### A.4 Evaluation Details

Across all checkpoints, we evaluate using a temperature $\tau = 0.6$ and a top-$p$ value of 0.95, following prior works [47, 48, 18], with a maximum generation of 3072 tokens for Qwen2.5-Math due to limited context length. For Llama-3.2-3B-Instruct, we allow the model to generate a maximum of 16384 tokens.

## B Details Derivation of GRPO

Given a problem $\boldsymbol{x}$ and the generated solutions $\{\boldsymbol{y}\}_{i=1}^{G}$, where $G > 2$ is the number of generated solutions per prompt. We define each token index $t$ in the response $\boldsymbol{y}$ as $y_t$ with $1 \leq t \leq |\boldsymbol{y}|$, where $|\boldsymbol{y}|$ is the sequence length of the reasoning traces. Specifically, for each problem $\boldsymbol{x}$, GRPO objective DeepSeek-AI et al. [6] will first calculate group-wise normalized advantage function:

$$A(\boldsymbol{x}, \boldsymbol{y}_i) = \frac{r(\boldsymbol{x}, \boldsymbol{y}_i) - \hat{\mu}}{\hat{\sigma}} \tag{10}$$

where $\hat{\mu} = \frac{1}{G}\sum_{j=1}^{G} r(\boldsymbol{x}, \boldsymbol{y}_j)$, and $\hat{\sigma} = \sqrt{\frac{1}{G-1}\sum_{j=1}^{G}(r(\boldsymbol{x}, \boldsymbol{y}_j) - \hat{\mu})^2}$.

Let $\pi_{\text{old}}$ denote the policy from the previous update step that generates the reasoning traces. The probability ratio $\rho_t = \frac{\pi_\theta(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t})}{\pi_{\text{old}}(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t})}$ at each token index $t$, along with PPO clipping mechanism to serves as a proxy of trust region contraint with clipping threshold $\epsilon$, the GRPO objective is defined as:

$$\begin{aligned} J(\theta)* = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D},\, \boldsymbol{y}\sim\pi_{\text{old}}(\cdot|x)} \left[ \sum_{t=1}^{|y|} \min\Big(\rho_t(y;\theta)\, A(\boldsymbol{x}, \boldsymbol{y}_i),\ \text{clip}\big(\rho_t(y;\theta), 1-\epsilon, 1+\epsilon\big) A(\boldsymbol{x}, \boldsymbol{y}_i)\Big) \right] \\ - \beta\, \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \Big[ \text{KL}\big(\pi_\theta(\cdot|\boldsymbol{x}) \,\|\, \pi_b(\cdot|\boldsymbol{x})\big) \Big]. \end{aligned} \tag{11}$$

The KL regularization helps the learned policy not deviate too far from the base model. Recently, Yu et al. [46] suggests that removing KL regularization can provide better performance. Following this, we set $\beta = 0$ to eliminate KL regularization in our work.

# C Per-step Influence Proofs and Further Analysis

## C.1 Proof of Proposition 3.1

**Proposition C.1.** *Consider a problem-solutions pair $(\boldsymbol{x}, \boldsymbol{y})$ for updating using policy gradient objective, a sufficiently small learning rate $\eta$, the per-step influence on a problem-solution pair $(\boldsymbol{x}', \boldsymbol{y}')$ is defined as:*

$$\Delta \log \pi_\theta^t(\boldsymbol{y}'|\boldsymbol{x}') = \eta\, \mathcal{K}^t(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{y}, \boldsymbol{y}') A(\boldsymbol{x}, \boldsymbol{y}) + \mathcal{O}(\eta^2) \tag{12}$$

*where $\mathcal{K}^t(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{y}, \boldsymbol{y}) = \nabla_\theta \log \pi_{\theta^t}(\boldsymbol{y}|\boldsymbol{x})^\top \nabla_\theta \log \pi_{\theta^t}(\boldsymbol{y}'|\boldsymbol{x}')$ measures the influence between $(\boldsymbol{x}, \boldsymbol{y})$ and $(\boldsymbol{x}', \boldsymbol{y}')$, and the advantage function $A(\boldsymbol{x}, \boldsymbol{y})$ determines the magnitude and direction of the model's update.*

*Proof.* We first approximate $\log \pi_{\theta^{t+1}}(\boldsymbol{y}|\boldsymbol{x})$ using Taylor expansion. For simplicity, we denote $\pi_{\theta^t} = \pi_\theta^t$:

$$\log \pi_\theta^{t+1}(\boldsymbol{y}|\boldsymbol{x}) = \log \pi_\theta^t(\boldsymbol{y}|\boldsymbol{x}) + \nabla_\theta \log \pi_\theta^t(\boldsymbol{y}|\boldsymbol{x})^\top (\theta^{t+1} - \theta^t) + \mathcal{O}(\eta^2) \tag{13}$$

Using the definition of stochastic gradient ascent with a sufficiently small learning rate $\eta$, we have that:

$$\theta^{t+1} = \theta^t + \eta \nabla_\theta \mathcal{J}(\pi_\theta) \tag{14}$$

where $\nabla_\theta \mathcal{J}(\pi_\theta) = A(\boldsymbol{x}, \boldsymbol{y}) \nabla_\theta \log \pi_\theta(\boldsymbol{y}|\boldsymbol{x})$ is the standard policy gradient objective [33, 1], we plug in the above objective in Eq. 13:

$$\Delta \log \pi_\theta^t(\boldsymbol{y}'|\boldsymbol{x}') = \eta \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{y}, \boldsymbol{y}') A(\boldsymbol{x}, \boldsymbol{y}) + \mathcal{O}(\eta^2) \tag{15}$$

which concludes the proof. □

## C.2 Interpretation of the relative strength of influence $||\Delta(\pi_{\theta^t}\mu)||$

If we view the language model $\pi_\theta$ as a neural network with a problem and solution pair $\boldsymbol{o} = (\boldsymbol{x}, \boldsymbol{y})$ as an input, where the output is the log-probability

$$\log \pi_\theta(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{|\boldsymbol{y}|} \sum_{t=1}^{|\boldsymbol{y}|} \log \pi_\theta(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}) \tag{16}$$

where $\boldsymbol{y}_{<t}$ is the partial completion before the token index $t$. We consider the gradient of $\pi_\theta$ with respect to parameter $\theta$ at $(\boldsymbol{x}, \boldsymbol{y})$ as $\nabla_\theta \log \pi_\theta(\boldsymbol{y}|\boldsymbol{x}) \in \mathbb{R}^d$ where $d$ is the parameter dimension. We can define a matrix of gradient dot products $\mathcal{K}(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{y}, \boldsymbol{y}') = \mathcal{K}(\boldsymbol{o}, \boldsymbol{o}')$ across all different prompt-response pair:

$$\mathcal{K} = \begin{bmatrix} \mathcal{K}(\boldsymbol{o}_1, \boldsymbol{o}_1) & \mathcal{K}(\boldsymbol{o}_1, \boldsymbol{o}_2) & \cdots \\ \mathcal{K}(\boldsymbol{o}_2, \boldsymbol{o}_1) & \mathcal{K}(\boldsymbol{o}_2, \boldsymbol{o}_2) & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \tag{17}$$

$$= \nabla_\theta \log \pi_\theta(\mathcal{Y}|\mathcal{X})^\top \nabla_\theta \log \pi_\theta(\mathcal{Y}|\mathcal{X}) \tag{18}$$

where $\nabla_\theta \log \pi_\theta(\mathcal{Y}|\mathcal{X}) = [\nabla_\theta \log \pi_\theta(\boldsymbol{y}_1|\boldsymbol{x}_1), \nabla_\theta \log \pi_\theta(\boldsymbol{y}_2|\boldsymbol{x}_2), \cdots]$ denotes the matrix of the gradient of different prompt-solution pairs $(\boldsymbol{x}, \boldsymbol{y})$.

In an idealized scenario, the kernel matrix $\mathcal{K}$ is diagonal, meaning that all off-diagonal terms are zeros, i.e., $\mathcal{K}(\boldsymbol{o}_i, \boldsymbol{o}_j) = 0$ for $i \neq j$. Under this assumption, improving performance on a given training problem-solution pair would not influence any other unseen problems. Consequently, there would be neither *negative interference*, where training on one example degrades performance on others, nor *coverage shrinkage*, where the set of solvable problems becomes narrower.

In practice, however, this assumption rarely holds for language models. Because parameters are shared across all examples, updates to one example inevitably affect others. This manifests as non-zero off-diagonal entries in $\mathcal{K}$, which capture the degree of cross-influence between examples.

The growing magnitude of the relative influence strength, $||\Delta(\pi_{\theta^t}, \mu)||$, can thus be interpreted as evidence that the number and magnitude of these off-diagonal interactions are increasing. Specifically, this signals that the gradients of different examples are becoming more linearly dependent, since the rank of the kernel $\mathcal{K}$ is equivalent to the rank of the gradient matrix $\nabla_\theta \log \pi_\theta(\mathcal{Y}|\mathcal{X})$. This growing entanglement explains why optimization on a subset of problems may inadvertently interfere with generalization to others.

# D Analyzing learning dynamic of on-policy learning in toy bandit problem.

We first consider a contextual bandit problem with a parametrized softmax policy with $V$ actions. where the input data $\boldsymbol{x}$ is a $d$ dimensional feature vector $\boldsymbol{x} \in \mathbb{R}^d$. The policy is defined as a linear layer with Softmax output:

$$\pi_\theta(y_i|\boldsymbol{x}) = \frac{\exp(z_i(\boldsymbol{x}))}{\sum_{j=1}^{V} \exp(z_j(\boldsymbol{x}))} \tag{19}$$

We consider the objective of REINFORCE and Off-policy maximum likelihood objectives for 2 actions $y_1$ and $y_2$ that receive a positive reward $(r(\boldsymbol{x}, y) = 1)$.

$$\mathcal{J}^{\text{REINFORCE}}(\pi_\theta) = \pi_\theta(\boldsymbol{y}_1|\boldsymbol{x}) + \pi_\theta(\boldsymbol{y}_2|\boldsymbol{x}), \qquad \mathcal{J}^{\text{MLE}}(\pi_\theta) = \log \pi_\theta(\boldsymbol{y}_1|\boldsymbol{x}) + \log \pi_\theta(\boldsymbol{y}_2|\boldsymbol{x}) \tag{20}$$

Using gradient descent, we can perform parameter update at each update step $t$ with a learning rate $\eta$:

$$\begin{aligned}
\boldsymbol{\theta}^{t+1} &= \boldsymbol{\theta}^t + \eta \nabla_{\boldsymbol{\theta}} \mathcal{J}^{\text{REINFORCE}} \\
&= \boldsymbol{\theta}^t - \eta \boldsymbol{x} \Big[ (\pi_{\theta^t}(y_1)(\pi_{\theta^t}(\boldsymbol{y}) - \boldsymbol{e}(y_1))^\top \\
&\quad + \pi_{\theta^t}(y_2)(\pi_{\theta^t}(\boldsymbol{y}) - \boldsymbol{e}(y_2))^\top \Big] \\
\boldsymbol{\theta}^{t+1} &= \boldsymbol{\theta}^t + \eta \nabla_{\boldsymbol{\theta}} \mathcal{J}^{\text{MLE}} \\
&= \boldsymbol{\theta}^t - \eta \boldsymbol{x} \left[ (\pi_{\theta^t}(\boldsymbol{y}) - \boldsymbol{e}(\boldsymbol{y}_1))^\top + (\pi_{\theta^t}(\boldsymbol{y}) - \boldsymbol{e}(\boldsymbol{y}_2))^\top \right]
\end{aligned} \tag{21}$$

where $\pi_\theta(\boldsymbol{y}) \in \mathbb{R}^V$ is the vector represents probability distribution over all actions, $\boldsymbol{e}(y)$ is a one-hot vector indicates the target action. A key difference between the REINFORCE and MLE objectives is that, in REINFORCE, the probability of the sampled actions directly scales the update step. As a result, actions with higher likelihood receive larger updates, while low-likelihood actions lead to much smaller changes. To analyze how the model's probabilities in each action change, we define a ratio $\alpha_i = \frac{\pi_i^{t+1}}{\pi^t{}_i}$ and use The following lemma describes its behavior:

**Lemma D.1.** *The ratio of confidence change for each $i$ can be represented as:*

$$\alpha_i^{\textit{MLE}} = \frac{\pi_i^{t+1}}{\pi_i^t} = \frac{\sum_{j=1}^{V} e^{z_j^t}}{\sum_{j=1}^{V} \beta_j e^{z_j^t}}; \quad \alpha_i^{\textit{REINFORCE}} = \frac{\sum_{j=1}^{V} e^{z_j^t}}{\sum_{j=1}^{V} \gamma_j e^{z_j^t}} \tag{22}$$

*Note that the values of $\beta_j$ also depends on whether $i \in \{y_1, y_2\}$, hence for Case 1 ($i \in \{y_1, y_2\}$), and Case 2 ($i \notin \{y_1, y_2\}$):*

*Case 1:*

$$\beta_j = \begin{cases} e^{-\eta'\left(2(\pi_j^t - \pi_i^t + 1)\right)} & \textit{if } j \notin \{y_1, y_2\} \\ 1 & \textit{if } j = y_1 \\ e^{-2\eta'(\pi_j^t - \pi_i^t)} & \textit{if } j = y_2 \end{cases}; \tag{23}$$

$$\gamma_j = \begin{cases} e^{-\eta'\left(\Delta\pi^t(\pi_j^t - \pi_i^t) + \pi_i^t\right)} & \textit{if } j \notin \{y_1, y_2\} \\ 1 & \textit{if } j = y_1 \\ e^{-\eta'\left((\Delta\pi^t - 1)(\pi_j^t - \pi_i^t)\right)} & \textit{if } j = y_2 \end{cases} \tag{24}$$

*Case 2:*

$$\beta_j = \begin{cases} e^{-2\eta'\left(\pi_j^t - \pi_i^t\right)} & \text{if } j \notin \{y_1, y_2\} \\ e^{-\eta'\left(2(\pi_j^t - \pi_i^t) - 1\right)} & \text{if } j \in \{y_1, y_2\} \end{cases}; \tag{25}$$

$$\gamma_j = \begin{cases} e^{-\eta'\Delta\pi^t\left(\pi_j^t - \pi_i^t\right)} & \text{if } j \notin \{y_1, y_2\} \\ e^{-\eta'\left(\Delta\pi^t(\pi_j^t - \pi_i^t) - \pi_j^t\right)} & \text{if } j \in \{y_1, y_2\} \end{cases} \tag{26}$$

*where* $\Delta\pi^t = \pi_{y_1}^t + \pi_{y_2}^t, \eta' = \eta||\boldsymbol{x}||_2^2$.

*Proof.* We first need to link the logits vector $z^{t+1}$ and $z^t$. From Eq.21, $z^{t+1}$ can be recursively written as:

$$\begin{aligned} z^{t+1} &= (\boldsymbol{\theta}^{t+1})^\top \boldsymbol{x} \\ &= \left(\boldsymbol{\theta}^t + \eta \boldsymbol{x} \nabla_z \mathcal{J})\right)^\top \boldsymbol{x} \\ &= (\boldsymbol{\theta}^t)^\top \boldsymbol{x} + \eta \left(\boldsymbol{x} \nabla_z \mathcal{J}\right)^\top \boldsymbol{x} \\ &= z^t + \eta' \nabla_z \mathcal{J} \end{aligned} \tag{27}$$

where $\eta' = \eta||\boldsymbol{x}||_2^2$. We can write down for each value in vector $z^{t+1}$ for MLE and REINFORCE objectives as:

$$\text{MLE: } z_i^{t+1} = \begin{cases} z_i^t - 2\eta'\pi_i^t & \text{if } i \notin \{y_1, y_2\} \\ z_i^t - 2\eta'\pi_i^t + \eta' & \text{if } i \in \{y_1, y_2\} \end{cases}$$

$$\text{REINFORCE: } z_i^{t+1} = \begin{cases} z^t - \eta'(\Delta\pi^t \cdot \pi_i^t) & \text{if } i \notin \{y_1, y_2\} \\ z^t - \eta'(\Delta\pi^t - 1)\pi_i^t & \text{if } i \in \{y_1, y_2\} \end{cases}$$

Then, we can write down the probability change $\pi_i^{t+1}$ for each action $i$. For case $i \in \{y_1, y_2\}$, we have the following derivation for MLE objective:

$$\begin{aligned} \pi_{y_1}^t &= \frac{e^{z_i^{t+1}}}{\sum_{j=1}^V e^{z_j^{t+1}}} \\ &= \frac{e^{z_i^t - 2\eta'\pi_i^t + \eta'}}{\sum\limits_{j \notin \{y_1, y_2\}} e^{z_j^t - 2\eta'\pi_j^t} + e^{z_{y_1}^t - 2\eta'\pi_{y_1}^t + \eta'} + e^{z_{y_2}^t - 2\eta'\pi_{y_2}^t + \eta'}} \\ &= \frac{e^{z_i^t}}{\sum\limits_{j \notin \{y_1, y_2\}} e^{z_j^t - \eta'\left(2(\pi_j^t - \pi_i^t) + 1\right)} + e^{z_{y_1}^t} + e^{z_{y_2}^t - 2\eta'(\pi_{y_2}^t - \pi_i^t)}} \end{aligned}$$

Similarly, for REINFORCE objective,

$$\begin{aligned} \pi_{y_1}^t &= \frac{e^{z_i^{t+1}}}{\sum_{j=1}^V e^{z_j^{t+1}}} \\ &= \frac{e^{z_i^t - \eta'(\Delta\pi^t - 1)\pi_i^t}}{\sum\limits_{j \notin \{y_1, y_2\}} e^{z_j^t - \eta'(\Delta\pi^t \pi_j^t)} + e^{z_{y_1}^t - \eta'(\Delta\pi^t - 1)\pi_{y_1}^t} + e^{z_{y_2}^t - \eta'(\Delta\pi^t - 1)\pi_{y_2}^t}} \\ &= \frac{e^{z_i^t}}{\sum\limits_{j \notin \{y_1, y_2\}} e^{z_j^t - \eta'\left(\Delta\pi^t(\pi_j^t - \pi_i^t) + \pi_i^t\right)} + e^{z_{y_1}^t} + e^{z_{y_2}^t - \eta'\left((\Delta\pi^t - 1)(\pi_{y_2}^t - \pi_i^t)\right)}} \end{aligned} \tag{28}$$

For case $i \notin \{y_1, y_2\}$, we have the following derivation for MLE objective:

$$\begin{aligned}
\pi_i^t &= \frac{e^{z_i^{t+1}}}{\sum_{j=1}^{V} e^{z_j^{t+1}}} \\
&= \frac{e^{z_i^t - 2\eta' \pi_i^t}}{\sum\limits_{j \notin \{y_1, y_2\}} e^{z_j^t - 2\eta' \pi_j^t} + e^{z_{y_1}^t - 2\eta' \pi_{y_1}^t + \eta'} + e^{z_{y_2}^t - 2\eta' \pi_{y_2}^t + \eta'}} \\
&= \frac{e^{z_i^t}}{\sum\limits_{j \notin \{y_1, y_2\}} e^{z_j^t - 2\eta'(\pi_j^t - \pi_i^t)} + e^{z_{y_1}^t - \eta'\left(2(\pi_{y_1}^t - \pi_i^t) - 1\right)} + e^{z_{y_2}^t - \eta'\left(2(\pi_{y_2}^t - \pi_i^t) - 1\right)}}
\end{aligned} \tag{29}$$

Similarly, for the REINFORCE objective:

$$\begin{aligned}
\pi_i^t &= \frac{e^{z_i^{t+1}}}{\sum_{j=1}^{V} e^{z_j^{t+1}}} \\
&= \frac{e^{z_i^t - \eta'(\Delta\pi^t\, \pi_i^t)}}{\sum\limits_{j \notin \{y_1, y_2\}} e^{z_j^t - \eta'(\Delta\pi^t\, \pi_j^t)} + e^{z_{y_1}^t - \eta'(\Delta\pi^t - 1)\pi_{y_1}^t} + e^{z_{y_2}^t - \eta'(\Delta\pi^t - 1)\pi_{y_2}^t}} \\
&= \frac{e^{z_i^t}}{\sum\limits_{j \notin \{y_1, y_2\}} e^{z_j^t - \eta'\left(\Delta\pi^t(\pi_j^t - \pi_i^t)\right)} + e^{z_{y_1}^t - \eta'\left(\Delta\pi^t(\pi_{y_1}^t - \pi_i^t) - \pi_{y_1}^t\right)} + e^{z_{y_2}^t - \eta'\left(\Delta\pi^t(\pi_{y_2}^t - \pi_i^t) - \pi_{y_2}^t\right)}}
\end{aligned} \tag{30}$$

□

We can now better understand how each $\pi_i$ changes after one update. Specifically, if $\alpha_i > 1$, the corresponding $\pi_i$ increases, and vice versa. To determine the value of $\alpha_i$, we can treat any $\beta_j > 1$ as contributing to the conclusion that $\alpha_i < 1$, while any $\beta_j < 1$ against it. The value of the corresponding $e^{z_j^t}$ and $|\beta_j - 1|$ controls how strong the contribution is.

**Low-likelihood tokens tend to receives less meaningful updates.** Using the log-derivative trick, $\nabla_\theta \pi_y = \pi_y \nabla_\theta \log \pi_y$, we see that REINFORCE inherently applies a conservative update rule: tokens with higher probability receive proportionally larger gradient updates, while low-probability tokens are updated much more weakly. This disproportionate updating can have a detrimental effect in cases where optimal actions may reside in low-probability regions of the distribution, yet REINFORCE fails to provide sufficient reinforcement for these tokens to be effectively explored and learned.

**Assuming $\pi_{y_1}^t > \pi_{y_2}^t$, then $\pi_{y_1}^t$ is guaranteed is increase, i.e., $\alpha_{y_1} > 1$.** To see this, consider the value of $\gamma$ in Case 1 of the REINFORCE objective. For any $j \notin y_1, y_2$, we have $\gamma_j < 1$ because

$$\Delta\pi^t(\pi_j^t - \pi_i^t) + \pi_i^t = \Delta\pi^t \pi_j^t + (1 - \Delta\pi^t)\pi_i^t > 0. \tag{31}$$

Moreover, since $\gamma_{y_1} = 1$ while $\gamma_{y_2} < 1$, the probability of the highest-likelihood action $y_1$ will increase. In addition, because the update step size depends on the probability of the sampled action, the smaller $\pi_{y_1}^t$ is, the more strongly $\pi_{y_2}^t$ will decrease, leading the model to gradually concentrate its mass on the dominant mode $\pi_{y_1}^t$. However, this is not the case for the maximum likelihood objective; later, as we show in section 2, the maximum likelihood objective will tend to increase the lower probability $\pi_{y_2}^t$, avoiding model collapse.

### D.1 Empirical Verification with a simple bandit problem.

To empirically verify the analysis above, we analyze using a simple logistic regression task following Ren and Sutherland [34]. We set the vocabulary size $V = 50$, with parameter dimension $d = 5$, a learning rate of $\eta = 0.5$, and a randomly generated input vector $\boldsymbol{x}$. We compared the update effect

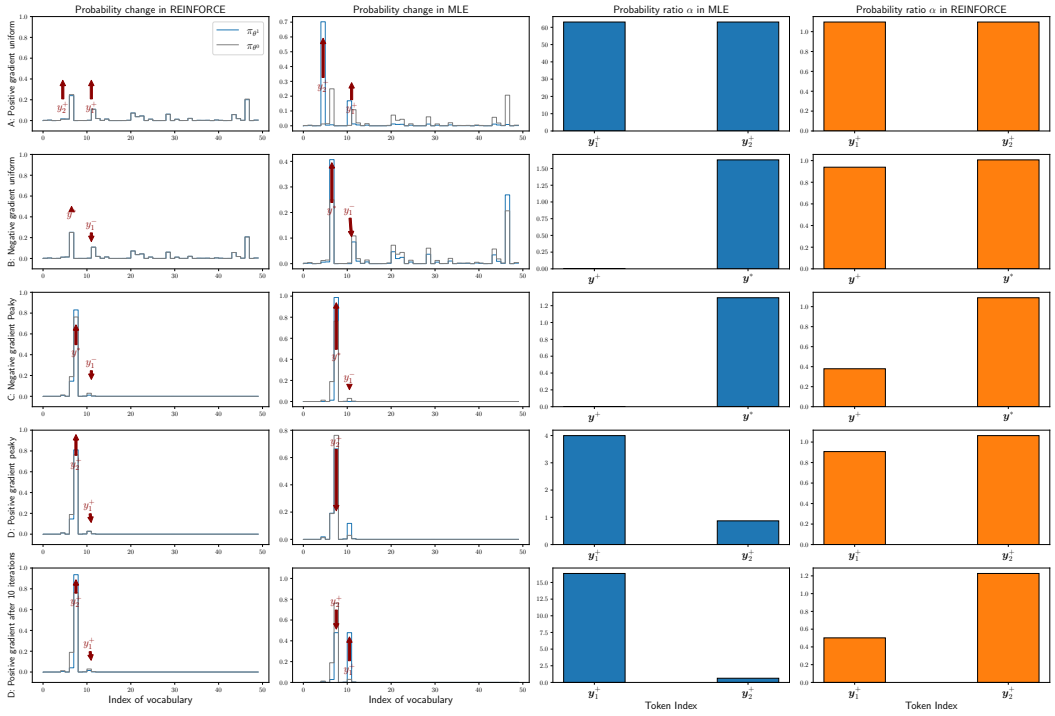



Figure 9: Experimental verification and comparison between REINFORCE and MLE objectives using a simple contextual bandit problem.

of MLE and REINFORCE objectives in 2 scenarios: when the initial distribution $\pi_{\theta^0}$ is relatively uniform and when it's highly peaked around certain dimensions.

**Uniform case:** we consider negative and positive gradient effect, in the negative gradient case, we consider negative label $\boldsymbol{y}^- = 11$ and investigate the "indirect" push-up effect on $\boldsymbol{y}_2 = 5$. In the positive gradient scenario, we use 2 positive actions $\boldsymbol{y}_1 = 11, \boldsymbol{y}_2 = 5$ for the SGD update. As demonstrated in the first 2 rows in Fig. 9, both REINFORCE and MLE behave similarly in both negative and positive gradient scenarios, where both $\boldsymbol{y}_1, \boldsymbol{y}_2$ are increased. However, MLE aggressively imposes a large update, where the probability is approximately 60 times larger on these two tokens after the update, as measured by the probability ratio $\alpha$. Similarly, a negative gradient tends to increase the highest-likelihood token $\boldsymbol{y}^*$, while decreasing other actions, including the positive action $\boldsymbol{y}_2$. This can be explained by REINFORCE by imposing a probability step-size, resulting in a more conservative update, where the models less deviate from their previous step compared to the MLE objective.

**Peaky case:** In LLM regime, the LMs usually initialize from a pre-trained checkpoint, where the model exhibits a non-uniform distribution. We consider the scenario where the models concentrate on a few dimensions in action distribution. Similarly, negative label $\boldsymbol{y}^- = 11$ and investigate the "indirect" push-up effect on $\boldsymbol{y}_2 = 8$. In the positive gradient scenario, we use 2 positive actions $\boldsymbol{y}_1 = 11, \boldsymbol{y}_2 = 8$ for the SGD update. As shown in the last 3 rows in Fig. 9, when the distribution is peaky, the negative gradient exacerbates the effect of increasing the highest probability token while decreasing other actions, regardless of correctness Ren and Sutherland [34]. Interestingly, in a positive gradient scenario, between the 2 positive actions, MLE tends to push up the actions with the lowest likelihood (as measured by $\alpha$). In the last row, we observe that after 10 iterations of training, MLE can converge to a new distribution with equal probabilities between 2 actions. However, REINFORCE shows favor for the actions with the highest probability; this effect will exacerbate after multiple updates, where it collapses the highest probability $\pi(\boldsymbol{y}_1 = 8)$. This suggests that offline learning can still provide benefits compared to REINFORCE in avoiding model collapse.

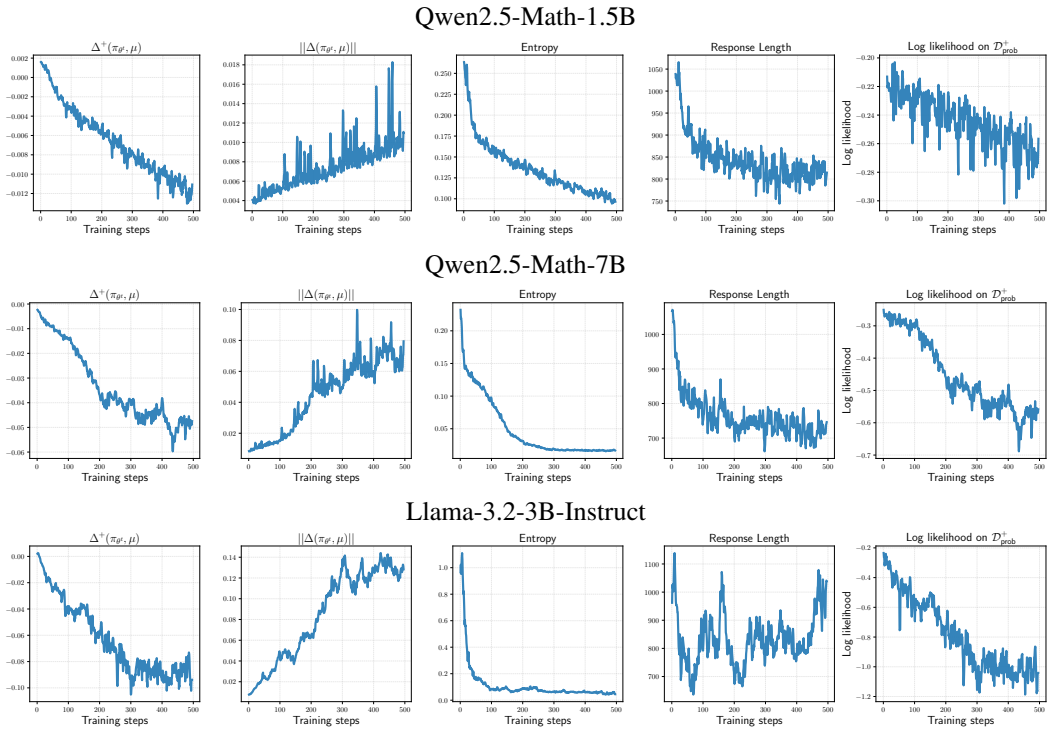



Figure 10: Learning dynamics of RLVR model with different models.

## E Additional Results on different models

**Consistent learning dynamics across different models.** In this section, we provide additional results that our findings hold across different model families and sizes. Specifically, we conduct GRPO experiments on a larger model, Qwen2.5-Math-7B, as well as a different family, Llama-3.2-3B-Instruct. As shown in Fig. 10, our analysis consistently holds across these settings.

**Larger models tend to suffer more from *negative interference*.** Interestingly, we find that the negative interference phenomenon becomes more severe when scaling up to larger models such as Qwen2.5-Math-7B. This also helps explain why, under RLVR training, the Pass@$k$ performance of Qwen2.5-Math-7B with large sampling budgets $k$ can even underperform than the smaller Qwen2.5-Math-1.5B after GRPO.

**Detailed Pass@$k$ performance.** We present detailed Pass@$k$ curves for Qwen2.5-Math-1.5B and Qwen2.5-Math-7B of different methods below:

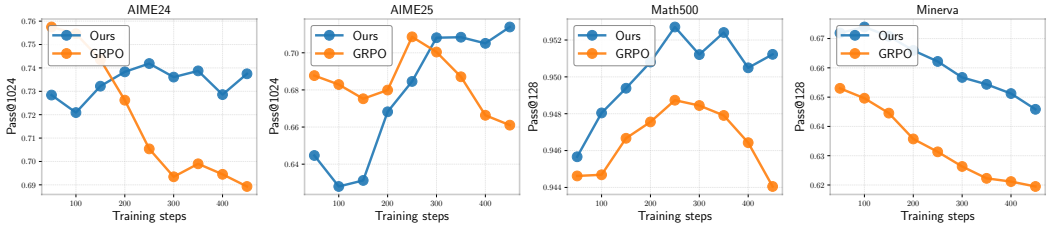



Figure 11: Pass@$k$ evolution (smoothed) during RLVR training with Qwen2.5-Math-1.5B, comparing our proposed finetuning objective SELF to GRPO under a large sampling budget $k$. While GRPO shows a progressive decline, SELF exhibits consistent improvements in Pass@$k$ throughout the training process.

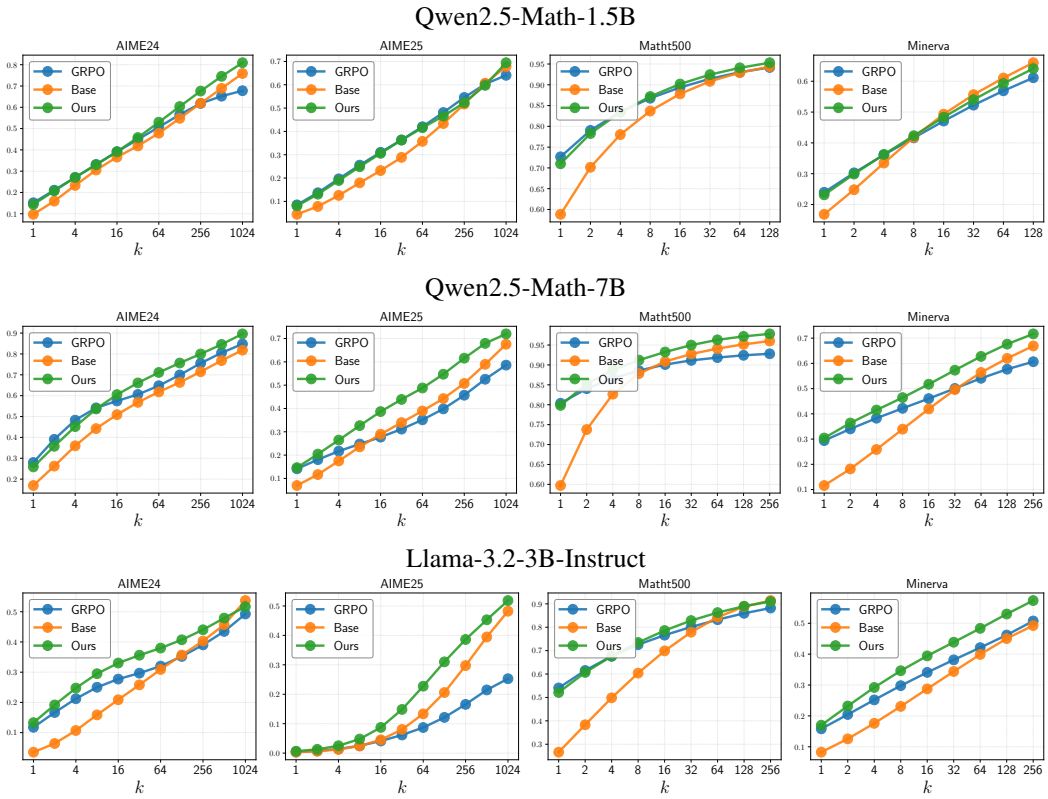



Figure 12: Pass@$k$ curves for base model, RLVR model and SELF with different models.

## F Performance degradation in W-REINFORCE

We also experiment with recent work **W-REINFORCE** Zhu et al. [50], which upweights the negative gradient signal to encourage diversity. As shown in Fig. 2, we find that **W-REINFORCE** suffers significantly from the catastrophic forgetting issue, where the average accuracy decreases even before 300 steps. We hypothesize that it is due to the high variance of the REINFORCE objective.

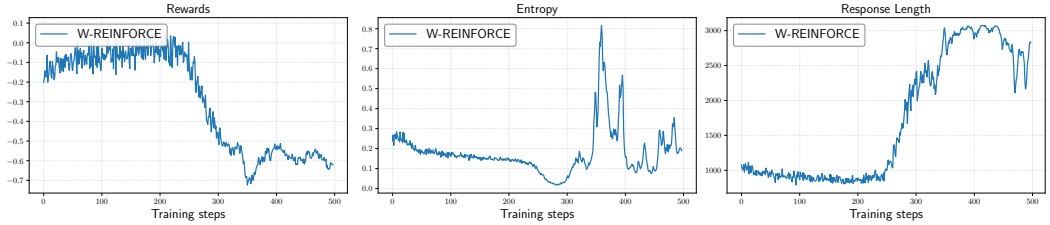



Figure 13: Learning dynamics of **W-REINFORCE**, we observe that **W-REINFORCE** suffer from training instability under prolong training.

Furthermore, progressively learning with a negative gradient can result in improper increases in tail probabilities Li et al. [20], leading the models to generate non-meaningful tokens. To understand this collapsing phenomenon arises intrinsically from the algorithms or from numerical precision issues in RLVR training [32]. We also conduct Float16 mixed-precision for **W-REINFORCE** in Fig. 14. We found that W-REINFORCE still results in a collapsing phenomenon, and even faster than bf16 after a small number of training steps. As a result, this collapsing phenomenon in **W-REINFORCE** is not from numerical precision issues but may arise from the algorithm perspective. We provide the following reasons:

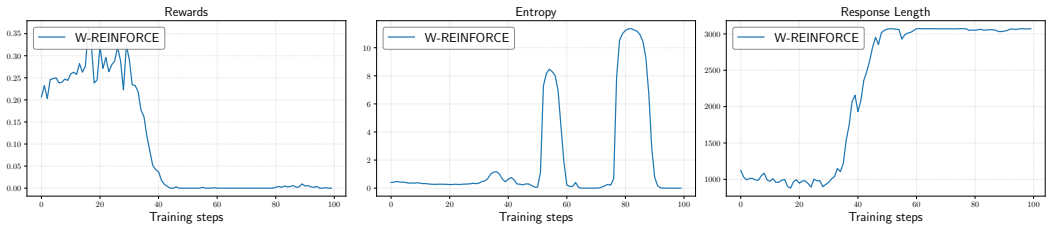



Figure 14: Learning dynamics of **W-REINFORCE** with FP16 Mixed Precision, training still collapses after 100 steps.

- **High variance issue in W-REINFORCE.** Different from GRPO and other algorithms, W-REINFORCE does not employ any baseline function, which results in a much higher variance policy gradient estimation.
- **Negative gradients in W-REINFORCE can increase the probability of void tokens.** W-REINFORCE focuses on learning from negative samples, negative gradients, which can indeed increase the exploration ability of LLMs by redistributing reduced probability mass to other tokens. However, under prolonged training, this can result in improper increases in tail probabilities, leading the model to generate non-meaningful tokens [20].

## G Computational cost of SELF

Our method requires an additional greedy generation to filter highly solvable problems. To analyze the computational cost of SELF and GRPO, we utilizethe per-iteration time of each update step with 3 factors: total generation time $T_{\text{gen}}$, backward time $T_{\text{backward}}$, and forward time $T_{\text{forward}}$. We provide our results in Fig. 15 We find that SELF incurs an additional cost in generation time compared to

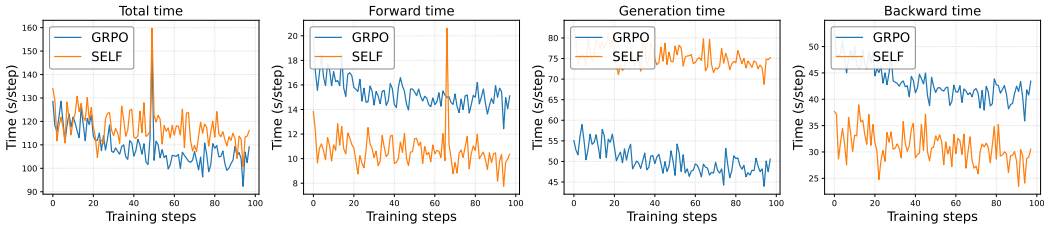



Figure 15: Computational results for training `Qwen2.5-Math-1.5B` for 100 steps.

GRPO. However, due to the filtering mechanism, both the forward and backward passes are faster, which compensates for the extra generation time and results in a negligible overall computational cost relative to GRPO.

## H Perplexity Detail Analysis

In this section, we provide a detailed perplexity analysis. Across all benchmarks, we randomly sample 16 reasoning traces for each problem in the test dataset to calculate perplexity.

To collect problems that increased after RLVR $\mathcal{D}^{\uparrow}$, we estimate the average accuracy for each problem and sort the problem indices according to average accuracy changes between the base policy and RLVR policy:

$$\Delta r(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{y}\sim\pi_\theta(\boldsymbol{y}|\boldsymbol{x})}\left[r(\boldsymbol{x},\boldsymbol{y})\right] - \mathbb{E}_{\boldsymbol{y}\sim\pi_b(\boldsymbol{y}|\boldsymbol{x})}\left[r(\boldsymbol{x},\boldsymbol{y})\right] \tag{32}$$

For AIME 2024/2025, we sample the top-3 problems that increased the most for $\mathcal{D}^{\uparrow}$ and top-3 problems decreased for $\mathcal{D}^{-}$, as these benchmarks only consist of 30 problems. For Math500 and Minerva, we sample top-64 problems for both $\mathcal{D}^{+}$ and $\mathcal{D}^{\downarrow}$. We present out results in Fig. 16.

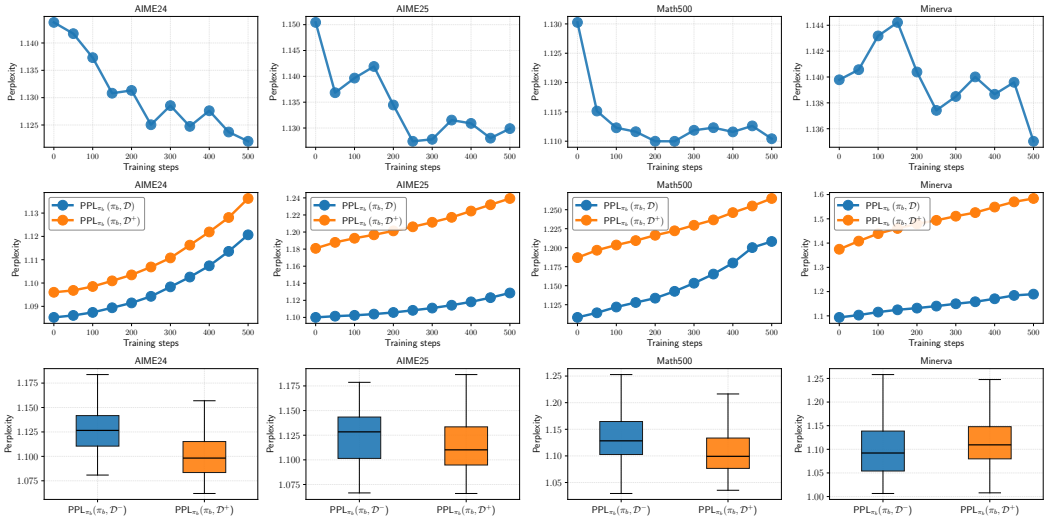



Figure 16: Dynamics of Perplexity during RLVR. The first row shows that RLVR increasingly generates low-perplexity responses under the base model. The second row demonstrates that the initial base solutions, regardless of correctness, exhibit increasing perplexity; the LMs are less likely to generate initial, diverse solutions. The final row shows that for problems where the correct solution already has a high probability to generate under base model, RLVR will incentivize, improving these problems, while problems with low success rate exhibits performance degradation.

# I Additional details in reducing previously learned behaviors in RLVR.

Similar to Shao et al. [38], we define *code reasoning* are responses that contain the string `python`. For each prompt $\boldsymbol{x}$, the LMs policy can generate either *code reasoning traces* or *lang reasoning traces*. We can define a mixture of distribution between these two reasoning traces:

$$\pi_b(\boldsymbol{y}|\boldsymbol{x}) = \alpha(\boldsymbol{x})\pi_{\text{code}}(\boldsymbol{y}|\boldsymbol{x}) + (1 - \alpha(\boldsymbol{x}))\pi_{\text{lang}}(\boldsymbol{y}|\boldsymbol{x}) \tag{33}$$

where $\alpha(\boldsymbol{x})$ is the mixture coefficient that depends on the prompt. For each prompt $\boldsymbol{x}$, we sample

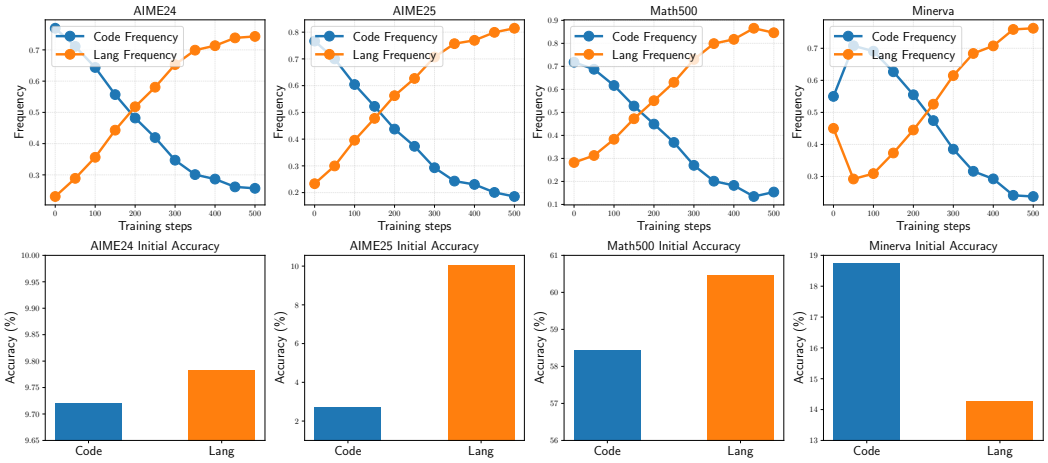



Figure 17: Reasoning strategy switching from `code` to `lang` throughout training process. While AIME24/25 and Math500 exhibits initial better accuracy in `lang` than `code`, Minerva shows better performance in *code reasoning traces*.

$k = 144$ responses to estimate the average accuracy of the mixture distributions $\pi_{\text{code}}$ and $\pi_{\text{lang}}$, denoted as $\rho_{\text{code}}(\boldsymbol{x})$ and $\rho_{\text{lang}}(\boldsymbol{x})$. We then define their average accuracies on the test set $\mathcal{D}_{\text{test}}$ as:

$$\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}\text{test}}\left[\rho_{\text{code}}(\boldsymbol{x})\right], \quad \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}\text{test}}\left[\rho_{\text{lang}}(\boldsymbol{x})\right]. \tag{34}$$

We report results for each reasoning behavior in the `Qwen2.5-Math` models across all benchmarks below:

We observe that the models eventually switch almost entirely to *lang reasoning traces*. We hypothesize that this shift explains the improvements on AIME24/25 and Math500, where *lang reasoning traces* is more effective initially. However, the loss of *code reasoning traces* reduces performance on Minerva, where the base model originally benefited from producing *code reasoning traces*.

## J Negative Interference and the effectiveness of SELF in non-mathematical tasks.

To asses the existence of negative interference phenomenon and the effectiveness of SELF in other domains, we utilize ReasoningGym [40], a library of reasoning environments for reinforcement learning with verifiable rewards spanning multiple domains such as algebra, arithmetic, cognition, graph theory, logic, etc.

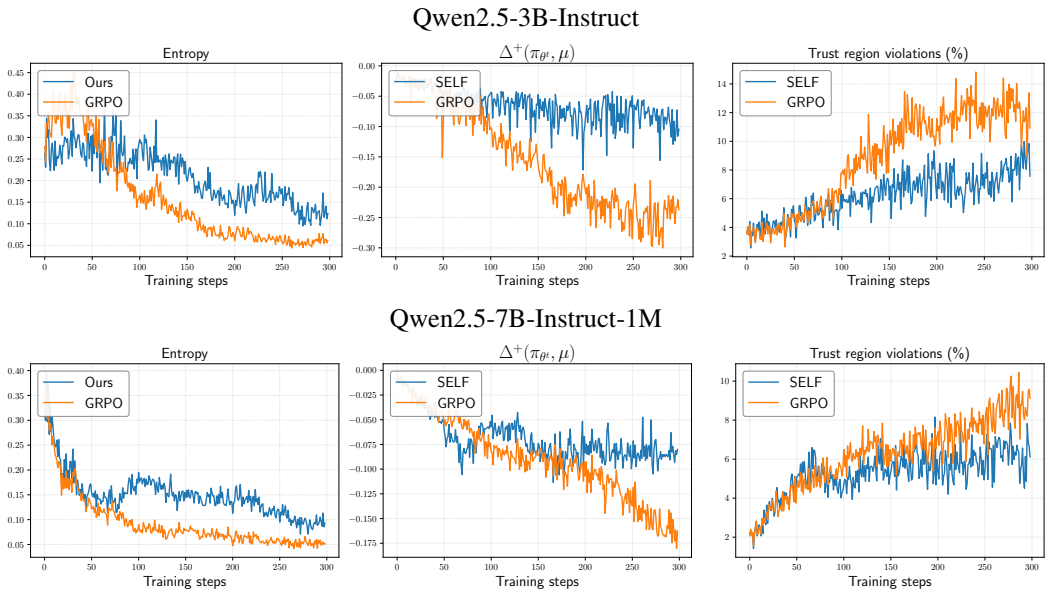



Figure 18: Learning dynamics of SELF and GRPO on ReasoningGym.

We collect $20k$ training examples spanning across 104 tasks and $2k$ samples for evaluation. We utilize 2 models for this study: Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct-1M, we measure Pass@$k$ metrics for SELF, GRPO and Base models, following the setups of previous works [22, 40]. We provide our results in Tab. 4 and Fig. 18, which can be summarized below:

Table 4: Evaluation results on ReasoningGym benchmark (%). The best performance under each Pass@$k$ is marked with **boldface**.

| Model | Method | Pass@1 | Pass@4 | Pass@16 | Pass@64 | Pass@128 |
|---|---|---|---|---|---|---|
| Qwen2.5-3B-Instruct | Base model | 16.37 | 26.26 | 35.81 | 43.62 | 46.88 |
| | GRPO | **34.76** | 40.04 | 44.6 | 48.69 | 50.69 |
| | SELF | 31.96 | **41.76** | **48.79** | **53.56** | **55.43** |
| Qwen2.5–7B-Instruct-1M | Base model | 26.85 | 39.16 | 48.2 | 54.89 | 57.75 |
| | GRPO | **42.94** | 49.33 | 54.33 | 59.23 | 61.56 |
| | SELF | 40.67 | **50.98** | **57.74** | **62.34** | **64.3** |

- **Negative interference still exists in non-math tasks**. We observed that the increasing negative interference phenomenon still exhibits on non-math tasks. This demonstrates that

the negative interference issue is not unique to mathematical domains but also arise in broader reasoning tasks.

- **SELF significantly mitigates negative interference and enhances coverage on non-math tasks**. Furthermore, we observe that SELF can effectively mitigate the negative interference phenomenon on non-math tasks, demonstrating that SELF can still generalize to other domains.
- **SELF displays scalability with model size.** We also observed that SELF displays stronger benefits with larger models (as shown in Tab 4 and Fig 19). In particular, the performance gap between SELF and both Base and GRPO grows as we scale from 3B to 7B models, suggesting that larger models can leverage SELF's exploration-guided filtering more effectively.

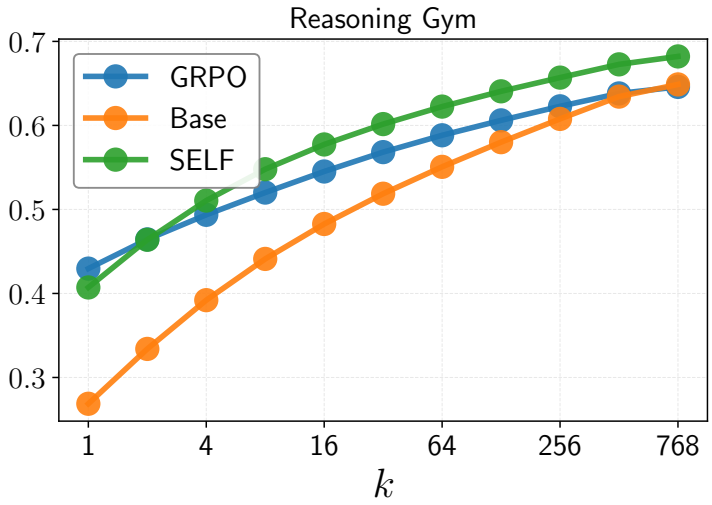



Figure 19: Pass@$k$ curves for Base, GRPO and SELF with Qwen2.5-7B-Instruct-1M on Reasoning-Gym benchmark.

Our experiments indicate that negative interference is not confined to math reasoning tasks, and demonstrate the effectiveness of SELF in robustly mitigating negative interference and improving solution coverage across a diverse set of non-math reasoning domains, and the effectiveness of SELF scales with model size. These results collectively support our hypothesis that strategies addressing coverage shrinkage in math domains naturally generalize to more general reasoning tasks.

## K On the effectiveness of greedy filtering

In this section, we evaluate the effectiveness of the greedy filtering mechanism in identifying low-likelihood problems. Intuitively, greedy responses provide a deterministic approximation of the highest likelihood solutions. When the greedy response fails on a given problem, it indicates that the model assigns low probability to correct solution paths under its most confident greedy decoding. To support this, we include a boxplot comparing the base model's accuracy on filtered versus non-filtered problems in Fig. 20. We observe a strong correlation between greedy-response failure and problems with low average accuracy, indicating that greedy decoding is an effective signal for identifying low-likelihood or inherently difficult problems with respect to the different models.

**The effectiveness of different decoding strategies.** Intuitively, our findings suggest that problems which are already highly solvable (i.e., those with high average accuracy) under the current model should be filtered. To quantify the effectiveness of different decoding strategies, we classify a problem as highly solvable if its average accuracy satisfies $\mathbb{E}_{y\sim\pi}[r(y)] \geq 0.6$. We estimate this average accuracy by averaging over 16 responses generated with a sampling temperature of $\tau = 0.6$, top-p $= 0.95$, following prior work [47].To evaluate the alignment between the filtering labels and the predictions of highly solvable problems, we report the F1 score, which accounts for mislabeling

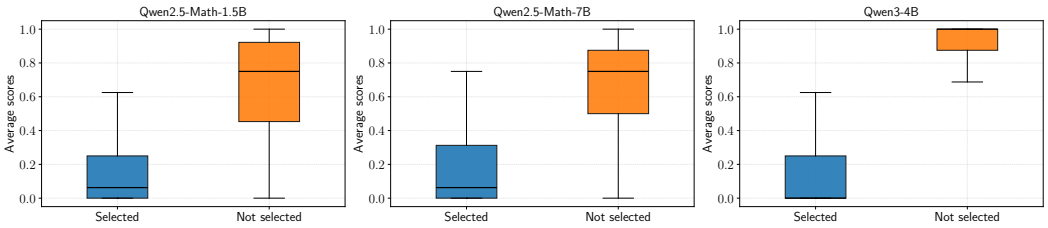



Figure 20: Average accuracy across filtered and non-filtered problems using greedy decoding.

errors of the filtering mechanism with recall metric. We further investigate different decoding strategies by varying the sampling parameters: $\tau = \{0.0, 0.6, 0.8, 1.0\}$, top-$p = 0.95$; we provide our results in Tab. 5:

Table 5: F1 score of different decoding strategies across Qwen models.

| **Models** | Greedy | $\tau = 0.6$ | $\tau = 0.8$ | $\tau = 1.0$ |
|---|---|---|---|---|
| Qwen2.5-Math-1.5B | **74.85** | 69.13 | 74.02 | 60.29 |
| Qwen2.5-Math-7B | **79.06** | 74.5 | 71.92 | 69.1 |
| Qwen3-4B | 91.72 | 92.51 | 90.96 | **92.56** |

We observed that greedy decoding is generally effective and comparable to other decoding strategies. Interestingly, for "thinking" models such as Qwen3, high-accuracy problems often coincide with cases where greedy decoding fails. Furthermore, when greedy decoding fails and tends to produce repetitive tokens, Qwen3 typically achieves low accuracy on those problems. Taken together, these results suggest that greedy decoding can serve as a reliable indicator for identifying low-likelihood solutions.

## L Additional comparison between SELF and SimKO.

We provide a comparison with SimKO [31], a recent method that mitigates probability collapse to improve Pass@k, as shown in Table 6 SELF outperforms SimKO on three of the four benchmarks

Table 6: Pass@256 performance across different benchmarks using Qwen2.5-MATH-7B.

| | AIME24 | AIME25 | MATH-500 | Minerva |
|---|---|---|---|---|
| SELF | **79.6** | 63.33 | **97.8** | **71.7** |
| SimKO | 78.0 | **64.6** | 96.8 | 68.3 |

while achieving comparable performance on AIME25, demonstrating that our method is highly effective at improving Pass@k performance and coverage.

## M Why SELF doesn't over-regularize high-confidence problems

On-policy algorithms inherently favor high-probability problems, as demonstrated in our analysis (Section 5), while low-likelihood problems receive disproportionately fewer updates. SELF mitigates this imbalance by deliberately emphasizing low-likelihood problems. This emphasis does not over-regularize high-confidence problems because gradients on low-likelihood problems are small in magnitude. Low-likelihood trajectories often produce weaker gradients due to their smaller probability of generating correct solutions; thus, the gradients tend to be weak in magnitude. As a result, even if these problems are sampled more frequently, their updates exert limited cross-problem influence and therefore have minimal impact on high-confidence problems. This trend is clearly shown in Fig. 8, where the per-step relative update strength in SELF is consistently much smaller than in GRPO. We further report the gradient norms of SELF and GRPO over the entire training

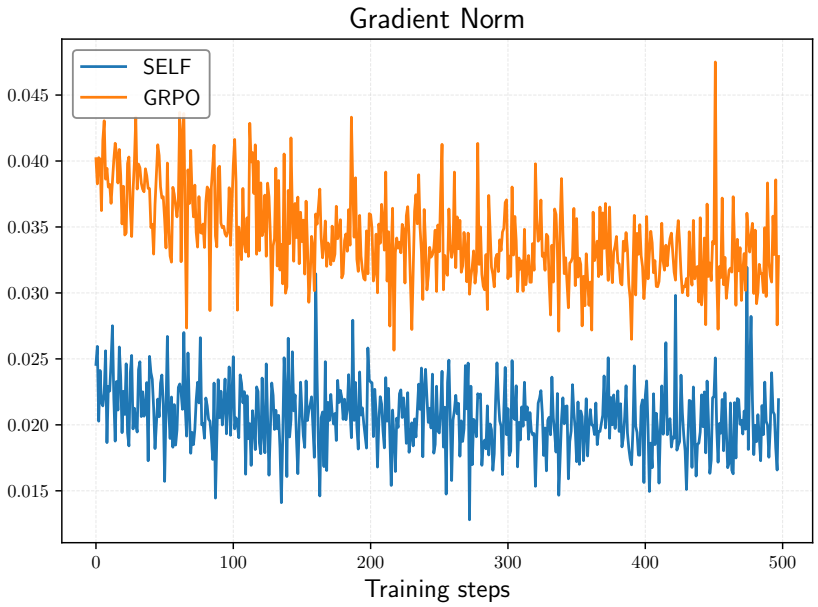



Figure 21: Gradient magnitude of SELF and GRPO throughout training process.

process in Fig. 21, confirming that SELF operates with substantially smaller gradient magnitudes than GRPO.

## N Ablation study on Beam Search

Our goal is to investigate the fundamental causes of RLVR's shrinkage. Accordingly, SELF, should be viewed as a preliminary step toward leveraging this understanding for mitigation. Importantly, the current F1 scores are already sufficiently high in retaining most high-quality, hard problems and can effectively mitigate the RLVR's shrinkage, sufficiently supporting the understanding of our goal. Nevertheless, we provide an additional experiment with width-3 beam search (significantly more expensive than SELF due to repeated sampling), using Qwen2.5-Math-1.5B. The F1 score is shown on table 7.

| **Models** | **Beam (K=3)** | **Greedy** |
|---|---|---|
| Qwen2.5-Math-1.5B | 67.69 | **74.85** |

Table 7: Ablation study of F1 score between Beam Search and Greedy Sampling

As observed, SELF's greedy decoding is generally more effective, indicating that identifying hard problems effectively and efficiently is non-trivial and worth a separate study.

## O Ablation on Forward KL and Greedy Filtering

We ablate (i) Forward KL regularization and (ii) data curation on Qwen2.5-Math-1.5B using Pass@$k$. The results in table 8 show that Forward KL helps preserve the initial policy's coverage, mitigating shrinkage compared to standard GRPO, while data curation is the primary driver of reasoning boundary expansion during training. Combining both components yields additional, consistent performance gains.

Table 8: Pass@1024 for AIME24, and Pass@256 for Math-500 and Minerva

| | AIME24 | Math-500 | Minerva |
|---|---|---|---|
| SELF | **80.94** | 96.0 | 68.75 |
| Base | 75.86 | 95.58 | **70.67** |
| SELF w.o. KL | 77.56 | **96.2** | 66.65 |
| SELF w.o. Data curation | 74.07 | 95.0 | 67.44 |
| GRPO | 67.77 | 95.0 | 65.07 |

## P On negative interference with different online policy algorithms

Negative interference is not unique to GRPO; it also arises in other on-policy algorithms. To support this, we conducted additional experiments using PPO on mathematical reasoning tasks with Qwen2.5-1.5B and report the results in Figure 22. Our findings indicate that negative interference persists under PPO, suggesting it is not unique to GRPO. Interestingly, while PPO appears somewhat less susceptible to negative interference compared to GRPO, the effect remains evident.

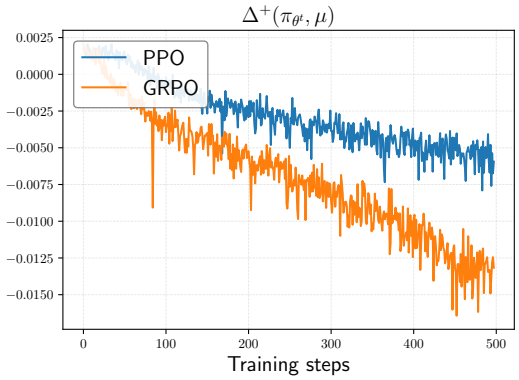



Figure 22: Negative interference

## NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: We discussed the reasoning boundary shrinkage issue during the training dynamics of RLVR (Section 4, 5) and proposed a simple solution for it (section 6).

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We mentioned in Impact Statement section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: All important proofs are in the appendix.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: All implementation details are in Appendix A

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The data and models are publicly available, and the code for reproducing our experiments is available at: `https://anonymous.4open.science/r/neurips26_self-CDD1/`

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: All implementation detail are in Appendix A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Figures and results table include errors bars and statistical significance where it's appropriate.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: They are mentioned in Appendix A and G

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This work solely focuses on understanding the coverage shrinkage problem of RLVR. No human subject or private data is involved in this work, and no experiment is conduct for harmful purposes.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We mentioned in the Impact Statement

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This work solely focuses on understanding the coverage shrinkage problem of RLVR. No new data or pretrained model is released in this work.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All data used in this work is publicly available for research purposes and is cited accordingly.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: This work solely focuses on understanding the coverage shrinkage problem of RLVR. No new data or pretrained model is released in this work.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This work solely focuses on understanding the coverage shrinkage problem of RLVR. No human subject is involved in this work.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This work solely focuses on understanding the coverage shrinkage problem of RLVR. No human subject is involved in this work.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: We solely use LLM to help with writing, all core methods and experiments were proposed without LLM involvement.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.