Towards Understanding the Optimization Landscape of GRPO and its Variants

Samyak Jain [*][★][□] **Ayush Agrawal**[◇][△][□] **Navin Goyal**[□]
[★]UC Berleley, [◇]MILA, [△]Université de Montréal, [□]Microsoft Research India

Abstract

GRPO has achieved impressive success in the landscape of reasoning models. However, the motivation behind its origins along with the reasons for its effectiveness remain elusive. In this work, we fill some of the gaps and demonstrate that in on-policy setting, GRPO's optimization can be viewed as a weighted combination of maximization of likelihood for correct rollouts and minimization for the incorrect ones. This finding gives a different perspective about the optimization landscape of GRPO. Motivated by this, we analyze the positive and negative part of GRPO's objective function independently, and find that their global minima correspond to undesired solutions. While optimization of the positive term leads to entropy minimization and length collapse, optimizing for the negative term leads to entropy maximization and length explosion. Using this lens, we show the presence of instability in on-policy training of some recent algorithmic advances trying to simplify GRPO's objective. However, despite the presence of bad global minima in GRPO's objective function, it doesn't converge to either of them. We identify design choices in GRPO's advantages that aid convergence of GRPO to good minima. We also demonstrate the effectiveness of using clipping in stabilizing the optimization process, thereby preventing training instabilities even when training only for minimizing the likelihood of incorrect rollouts.

1 Introduction

Reinforcement learning with verifiable rewards (RLVR) has shown impressive improvements in the reasoning abilities of Large Language Models (LLMs) on tasks like maths, coding, etc. DeepSeek-AI (2025); Jaech et al. (2024); Team et al. (2025). These improvements are a result of advancements in the capabilities of the base models, along with development of improved RL algorithms like GRPO (Shao et al., 2024) and PPO (Schulman et al., 2017b). It is interesting that merely using rewards at the end of model's prediction (output verifiable rewards) is sufficient to observe non-trivial improvements in reasoning abilities of LLMs. This observation has led to a large adoption of outcome reward models (ORMs) in the community resulting in increased popularity of RL finetuning methods like GRPO. As a result several modified versions of GRPO have been proposed recently, aiming to improve the training efficiency and generalization of RL-finetuned models.

Amongst these alternatives, a few of them have tried simplifying GRPO's objective function. For instance, (Xiong et al., 2025) demonstrated that very simple algorithm RAFT (and its enhancement RAFT++) that trains only on the correct rollouts give performance comparable with GRPO. On the other hand, Zhu et al. (2025) demonstrated that minimizing the likelihood only on the incorrect rollouts performs comparably to GRPO while improving model's output diversity. Additionally, Samineni et al. (2025) demonstrated that a simple combination of the positive and negative losses as described above, also leads to performance similar to GRPO. A few of the recently proposed alternatives have also tried modifying the clipping mechanisms in GRPO (MiniMax, 2025; Ahmadian et al., 2024). Similarly, Zheng et al. (2025); Zhao et al. (2025c) modify GRPO's objective function by considering importance sampling at sequence level. All these methods amongst several others

[*]Email: `samyakjain@berkeley.edu`

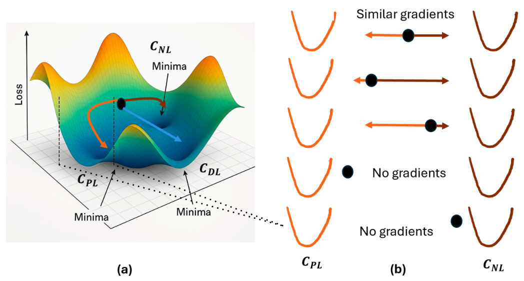


Figure 1: **Illustrative diagram demonstrating the loss landscape of RLVR methods:** (a) The loss landscape consists of different critical solutions C_{PL}, C_{NL}, and C_{DL}, where C_{PL} represents the minimum entropy solution, C_{NL} represents the maximum entropy solution, and C_{DL} leads to improved performance of the model. (b) GRPO stabilizes the model against converging to bad critical solutions C_{PL} and C_{NL}, by controlling the magnitude of the gradient as shown by the length of the arrows, and the illustration of the model in its function space is shown by the dark circle.

(Chen et al., 2025a; Lanchantin et al., 2025) aim to simplify GRPO's objective function and they have successfully demonstrated their effectiveness on certain datasets and base models. These results point to the lack of motivation about different design choices in GRPO's objective function.

Another line of works modifies the definition of advantages and rewards in GRPO: Chen et al. (2025b); Mahdavi et al. (2025); Zhou et al. (2025); Liu et al. (2025); Zhao et al. (2025b); Kang et al. (2025); Arora & Zanette (2025); He et al. (2025); Prabhudesai et al. (2025); Xiao et al. (2025); Hao et al. (2025); Fan et al. (2025); Yu et al. (2025); Shafayat et al. (2025); Arnal et al. (2025). In particular, Prabhudesai et al. (2025) show that using GRPO in unsupervised RL setting by minimizing model's entropy could lead to improved performance. On the other hand, Wang et al. (2025b) demonstrated that maximization of entropy could also lead to improved performance. These contrasting results have created confusion within the community, highlighting a need to understand the optimization landscape of GRPO and related simplified objective functions.

In this work, we primarily focus on understanding the role of algorithmic advancements in enabling improvements in reasoning abilities of LLMs. For this, we first try to understand the motivation behind each of the design choices used in GRPO's objective function, where we find that several of them lack adequate motivation. We first highlight the lack of motivation behind using clipping in GRPO's objective function by tracing back the origins of clipping proposed in PPO (Schulman et al., 2017b), which had its motivation grounded in policy improvement guarantee shown in TRPO (Schulman et al., 2017a). Thus we begin by analyzing GRPO's objective function in on-policy setting, and adding different design choices to reconstruct its actual objective function. In the on-policy setting, we demonstrate how several approximations inherently made by GRPO in comparison to PPO, simplify its learning process to a reweighted version of maximization and minimization of likelihood on the correct and incorrect rollouts respectively. Using this perspective, we analyze the loss on samples with correct and incorrect answers separately, and characterize the properties of their corresponding critical solutions. Let us term the critical solutions corresponding to maximizing and minimizing the likelihood for positive (correct) and negative (incorrect) samples respectively as C_{PL} and C_{NL} (as shown in Fig. 1 (a)). Empirically, we observe that converging to either of these minimas results in degraded performance, where training on the correct rollouts leads to collapse of entropy and length of model's outputs, and training on the incorrect rollouts leads to explosion of entropy and length of model's outputs. As GRPO's gradients can be considered a weighted mixture of the gradients of these losses, we aim to understand the reweighting mechanism which prevents GRPO from converging to either of the two bad solutions. As shown in Fig. 1 (b), we find that the advantages in GRPO help in reducing the norm of gradients when the model comes closer to either of the minimas in its function space, while increasing the norm of the gradient in the direction of farther away critical point.

Specifically, we find that the on-policy versions of the algorithms in Zhu et al. (2025); Xiong et al. (2025) are prone to instability and collapses, as explained by convergence to C_{PL} and C_{CL} above (these works focus their evaluations in off-policy setting without discussing the on-policy vs. off-policy distinction). However, utilizing clipping, makes the training stable for these methods, even though there is no policy improvement guarantee as in case of TRPO. Overall this indicates the critical role played by clipping in enabling off-policy learning to become more stable than on-policy, even in cases where the algorithms do not maximize a strict lower bound on the value function of the states in MDP, and therefore do not enjoy any policy improvement guarantee as in case of TRPO.

Designing more stable and robust RLVR methods is important from a practical viewpoint, and in an attempt to improve training stability of existing methods in on-policy setting, we find that utilizing normalization at token level could be helpful. We further explain the reasons behind this, thereby providing a new perspective on using token level normalization instead of sequence level, which has recently gained traction in the community as well Yu et al. (2025); Liu et al. (2025); Yue et al. (2025b). To summarize, our key contributions are:

- We provide a new view on GRPO as a reweighted version of maximization and minimization of likelihood for correct and incorrect samples.
- We characterize the properties of the critical solutions of the two minimas corresponding to maximization and minimization of likelihood for correct and incorrect samples, respectively.
- We show that clipping and utilizing token level normalization help off-policy training become more stable against collapsing as opposed to on-policy settings for different variations of GRPO.
- We demonstrate the key role played by the advantages used in GRPO in stabilizing the training.

We defer the detailed discussion of related work and background to Appendix A and B respectively.

2 Understanding the origins of GRPO

We consider an episodic MDP given by the tuple $(S, \mathtt{A}, P, r, \gamma)$, where S is a set of states and $\mathtt{A}_{s_t}$ is a set of actions allowed for a given state s_t. We assume same action set for all states gives us $\mathtt{A}_{s_0} = \mathtt{A}_{s_1} = ... = \mathtt{A}_{s_T} = \mathtt{A}$. The policy is parametrized by θ and defined as $\pi_\theta : R^d \to [0,1]^v$, where v is the cardinality of $\mathtt{A}$ and d is the dimension of the input. Denote by $P : S \times \mathtt{A} \times S \to R$ the transition probability matrix, by r_t the reward given by the environment at time stamp t. Let the process be episodic and always start from a state s_1. We can now define the advantage function $(A_\pi(s_t, a_t))$ for our policy π_θ: $A_\pi(s_t, a_t) = q_\pi(s_t, a_t) - v_\pi(s_t) = q_\pi(s_t, a_t) - \sum_{a_i} \pi_\theta(a|s_t) q_\pi(s_t, a_i)$, where $q_\pi(s_t, a_t)$ represents the Q-value function calculated at state s_t and for action a_t.

While the above formulation can be used for general MDPs, in the case of language models we get a special MDP: The input prompt q denotes the starting state given by s_1. An action refers to prediction of the next token and a state is obtained by appending the predicted token (i.e. $s_t = (q, a_1, ..., a_{t-1})$). Given the current state, the next state is deterministically determined by the action. A response is a set of actions given by $a = (a_1, a_2, ..., a_T)$, where the prediction of end of sequence (EOS) token determines the end of episode. We use verifiable rewards given at the end of the episode, where correct prediction results in a reward score of one, and an incorrect prediction results in zero reward. We will now derive GRPO using PPO as the base method, where PPO's objective function is:

$$J(\theta, q) = \mathop{\mathbb{E}}_{a \sim \pi_{\theta_{\text{old}}}(a|q)} \frac{1}{|a|} \sum_{t=1}^{|a|} \min\Big[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} A_\pi(s_t, a_t), \text{clip}\Big(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon\Big) A_\pi(s_t, a_t)\Big] \quad (1)$$

where π_θ and $\pi_{\theta_{\text{old}}}$ represent the current and the old policy utilized for sampling the rollouts. PPO utilizes generalized advantages $(A_\pi(s_t, a_t))$ which are motivated from TD learning (Sutton, 1988). Training of the policy can be done in either off-policy or on-policy setting. In the on-policy setting the same model is used for training and generating the rollouts (i.e., $\pi_{\theta_{old}} = \pi_\theta$), whereas in the off-policy setting an older version of our current policy is utilized for generating the rollouts.

GRPO builds on Eq. 1 and modifies the calculation of advantages. It approximates a) the computation of Q-value functions $(Q(s_t, a_t))$ by a single rollout, which results in an unbiased estimator with high variance. b) It also assumes that the Q-value and the value functions $(V(s_t))$ are the same for all the states in a rollout, where the value function is calculated by generating rollouts only at the starting state (s_1). This gives

$$A_\pi(q, a) = A_\pi(s_1, a_1) \overset{\text{assumes}}{=} A_\pi(s_T, a_T) = Q(s_1, a_1) - V(s_1) \overset{\text{assumes}}{=} r(a) - V(s_1) = r(a) - V(q) \quad (2)$$

GRPO also divides $A_\pi(s_{t+1}, a_{t+1})$ by the standard deviation of the rewards for the sampled trajectories which lacks desired motivation. Using the highlighted assumptions, we can now rewrite Eq. 1 in the following way:

$$J(\theta, q) = \mathop{\mathbb{E}}_{a \sim \pi_{\theta_{\text{old}}}(a|q)} \frac{1}{|a|} \sum_{t=1}^{|a|} \min\Big[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \tilde{A}_\pi(q, a), \text{clip}\Big(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon\Big) \tilde{A}_\pi(q, a)\Big] \quad (3)$$

where $\tilde{A}_\pi(q,a) = \frac{A_\pi(q,a)}{\mathbb{E}_{\tilde{a}\sim\pi_{\theta_{\text{old}}}(\tilde{a}|q)}(r(\tilde{a})-V(q))^2}$ Note that we do not incorporate the KL terms here as recent works: Zhou et al. (2025); Hu et al. (2025) have shown that using GRPO without the KL divergence results in faster convergence and improved performance. In eq. 3, GRPO further approximates the value function using a group of a few rollouts.

2.1 On-policy GRPO and iterated MLE

GRPO was motivated by PPO which in turn was obtained by modification of TRPO (Schulman et al., 2017a) whose history goes further back. However, the conceptual underpinnings of GRPO become somewhat obscured because of this long chain. We elaborate this further. TRPO enjoys a policy improvement guarantee in each iteration, by formalizing a constrained maximization of a strict lower bound on the value function, where the constraint ensures that the policy remains in close proximity of the old policy. PPO further proposes clipping as a heuristic to realize the constraint in practice. This led PPO enhance training stability of deep RL methods including the on-policy ones. However, due to the approximations detailed in Sec. 2, GRPO and its simplified versions no longer maximize a strict lower bound on the value function of the states in MDP in every iteration. This means that in an iteration some states might increase their value function while others might decrease it. As a result, there is no policy improvement guarantee, which fades away the motivation to utilize clipping, as used in PPO. This motivates us to unwrap different design choices in GRPO, and analyze their role in stabilizing training. Thus we start by investigating GRPO in the on-policy setting. First, we find that on-policy version of GRPO can be considered as an arguably simple and natural algorithm we call *iterated MLE*. Consider the following simple iterated maximum likelihood optimization algorithm: in each iteration, sample a prompt q, and then sample the rollouts a for this prompt. We form an expression for likelihood by taking the positive sign for a having the correct outcome and the negative sign for a having the incorrect outcome (specified by $A_\pi^{\text{sign}}(q,a)$ taking values $+1$ and -1, resp.).

$$L^{\text{IMLE}}(\theta,q) = \mathop{\mathbb{E}}_{a\sim\pi_\theta(a|q)} \sum_{t=1}^{|a|} A_\pi^{\text{sign}}(q,a)\log[\pi_\theta(a_t|s_t)]. \tag{4}$$

$$\theta_{i+1} = \theta_i + \alpha \mathop{\mathbb{E}}_{a\sim\pi_\theta(a|q)} \sum_{t=1}^{|a|} A_\pi^{\text{sign}}(q,a)\nabla\log[\pi_\theta(a_t|s_t)] = \theta_i + \alpha \mathop{\mathbb{E}}_{a\sim\pi_\theta(a|q)} \sum_{t=1}^{|a|} A_\pi^{\text{sign}}(q,a)\frac{\nabla\pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)} \tag{5}$$

In the case of GRPO we can write the gradients for Eq. 3 in the on-policy setting ($\pi_\theta = \pi_{\theta_{\text{old}}}$) as:

$$\theta_{i+1} = \theta_i + \alpha \mathop{\mathbb{E}}_{a\sim\pi_\theta(a|q)} \frac{\alpha}{|a|} A_\pi(q,a) \sum_{t=1}^{|a|} \frac{\nabla\pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)}. \tag{6}$$

By comparing Eq. 5 with Eq. 6, it is clear that we can consider GRPO as doing reweighted version of iterated MLE, where the reweighing of gradients is done at sample level with weight given by $\frac{|A_\pi(q,a)|}{|a|}$. Importantly, note that the sign of $A_\pi(q,a)$ agrees with $A_\pi^{\text{sign}}(q,a)$.

Note that when compared with classic on-policy methods like Reinforce, the key difference here is that in case of GRPO, the reweighting mechanism acts at sample level, where as in traditional on-policy objectives reweighting (using advantages) is done at token level. This distinction precisely occurs due to approximations inherently adopted by GRPO as discussed in Sec. 2. Further, this distinction inhibits GRPO from maximizing a strict lower bound of the true value functions of the states in the MDP. In summary, we have:

Takeaway 1

In the on-policy setting, GRPO behaves like reweighted iterated MLE.

With this new perspective, we will now try to uncover the role the reweighting mechanism and clipping used in GRPO's objective function. But first we define our experimental setup below.

3 Experimental setup

To make our findings robust across different settings, we apply various RLVR finetuning methods (GRPO, L_{PL}, L_{NL}, L_{CL}, PPO) on several training datasets: SimpleRL (Zeng et al., 2025), Countdown (Pan et al., 2025), Numina-Math (LI et al., 2024), and Numina-Math Hard which we specially crafted ourselves by filtering the prompts in Numina-Math that fail at Pass@2 when evaluated using Qwen2.5-7B (Qwen et al., 2025). More datasets details are in App. L. We also use multiple models: Qwen2.5-7B, Qwen2.5-7B-Instruct, and Llama3.1-8B-Instruct (Grattafiori et al., 2024). In most of the paper we plot the evolution of accuracy on train set of corresponding runs. The reader will notice that some of the plots are missing; the corresponding experiments had not been completed due to limited compute. Of course we are concerned about the test accuracy and we show in Table 1 that high train accuracy correlates with high test accuracy even across datasets different from the training dataset. For test set we use Math 500 (Hendrycks et al., 2021) , GSM8K (Cobbe et al., 2021), Minerva-Math (Lewkowycz et al., 2022), College-Math (col, 2024), OlympiadBench (He et al., 2024), and Gaokao-2023 (Zhang et al., 2024). Many of our plots involve a single model trained on multiple datasets (varying datasets), the base model in these plots is Qwen2.5-7B. Similarly, we have plots for varying base models for a fixed dataset SimpleRL. Ideally, one would try all combinations of datasets and models; this however is computationally infeasible. Experiments are run only once because of a large number of experiments. We use 8 rollouts per prompt, batch size of 128 prompts, training batch size of 128×8 samples for on-policy training, and 32×8 samples for off-policy training. For most other hyperparameters we use the default settings from VeRL (Sheng et al., 2024).

4 Unwrapping GRPO

In this section, we will perform several ablations on the design choices of GRPO to highlight the key ingredients behind GRPO's success. The ablations include training only on the correct rollouts or the incorrect ones, the use of advantages, on-policy and off-policy training with clipping. Since GRPO can be considered as a reweighted version of likelihood maximization and minimization on rollouts with correct and incorrect outputs respectively, we start by writing the empirical expectation of GRPO's objective function in Eq. 3 decomposed according to whether the response a to q has the correct outcome or not.

$$J(\theta, q) = \frac{1}{|\mathcal{A}^+| + |\mathcal{A}^-|}\Big(\sum_{a\in\mathcal{A}^+}\frac{1}{|a|}\sum_{t=1}^{|a|}\min[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}A_\pi(q,a), \text{clip}(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon)$$

$$A_\pi(q,a)] + \sum_{a\in\mathcal{A}^-}\frac{1}{|a|}\sum_{t=1}^{|a|}\min[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}A_\pi(q,a), \text{clip}(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon)A_\pi(q,a)]\Big) \quad (7)$$

where $\mathcal{A}^+$ corresponds to the set of correct rollouts, and $\mathcal{A}^-$ refers to the set of incorrect rollouts. Thus, $\mathcal{A}^+ \cup \mathcal{A}^- = \mathcal{A}$, where all the rollouts are sampled from $\pi_{\theta_{\text{old}}}$. Note that $a \in \mathcal{A}^+$ implies $A_\pi(q,a) \geq 0$, and $a \in \mathcal{A}^-$ implies $A_\pi(q,a) \leq 0$. To ablate the use of advantages in GRPO, we replace the advantages in equation 7 with their signs. (This gives a version of GRPO that's similar to iterated MLE but with clipping.) Let us name the objective corresponding to the set $\mathcal{A}^+$ as Positive Likelihood ($-L_{\text{PL}}$) and the objective corresponding to the set $\mathcal{A}^-$ as Negative Likelihood ($-L_{\text{NL}}$). In the on-policy setting, where $\pi_\theta = \pi_{\theta_{old}}$, these objectives simplify further as follows:

$$L_{\text{PL}}(\theta, q) = \frac{-1}{|\mathcal{A}^+|}\sum_{a\in\mathcal{A}^+}\frac{1}{|a|}\sum_{t=1}^{|a|}\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)};\ \nabla_\theta L_{\text{PL}} = \frac{-1}{|\mathcal{A}^+|}\sum_{a\in\mathcal{A}^+}\frac{1}{|a|}\sum_{t=1}^{|a|}\frac{\nabla_\theta\pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)} = 0 \quad (8)$$

$$L_{\text{NL}}(\theta, q) = \frac{1}{|\mathcal{A}^-|}\sum_{a\in\mathcal{A}^-}\frac{1}{|a|}\sum_{t=1}^{|a|}\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)};\ \nabla_\theta L_{\text{NL}} = \frac{1}{|\mathcal{A}^-|}\sum_{a\in\mathcal{A}^-}\frac{1}{|a|}\sum_{t=1}^{|a|}\frac{\nabla_\theta\pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)} = 0 \quad (9)$$

We note that the objective functions above have been analyzed in prior works Zhu et al. (2025); Xiong et al. (2025), where the authors indeed find them to be stable. This further motivates us to analyze these objectives in detail. We characterize the minimizers of the two objective functions. We first do this in an idealized setup where the response a consists of only one token and moreover we use the expectation instead of the empirical mean used in the expressions for L_{PL} and L_{NL} above. We suggest that the minimization of L_{PL} and L_{NL} leads to the problem of minimizing and maximizing the entropy of the distribution of a, resp. Note that in Eq. 4, by decomposing the expectation into two parts according to the correctness of the response a, we can express the objective as the difference of the entropies (denoted by H) on the two distributions of the two parts:

$$L^{\text{IMLE}}(\theta, q) = \Pr[A_\pi^{\text{sign}}(q,a) = 1] H[\pi_\theta(a|q, A_\pi^{\text{sign}}(q,a) = 1] - \Pr[A_\pi^{\text{sign}}(q,a) = -1] H[\pi_\theta(a|q, A_\pi^{\text{sign}}(q,a) = -1] \quad (10)$$

Note also from Eqs. 8 and 9 that the gradients of L_{PL} and L_{NL} have the same form as that of L^{IMLE} in Eq. 5. Now it's well known that the entropy of a probability distribution on a finite set is maximized for the uniform distribution and is minimized when the distribution is concentrated on a single point. From the above facts we infer that for the idealized case, L_{PL} is minimized by a distribution concentrated on a single token, and L_{NL} is minimized by the uniform distribution.

The experimental setup we study differs from the idealized setup above in the following ways: the number of tokens in the response is not limited to 1, and moreover only the empirical mean is used in the loss computation. Next, we hypothesize a generalization of the above idealized solution. Assuming that the cardinality of sets $\mathcal{A}^-$ and $\mathcal{A}^+$ is large enough, the global minima corresponding to Eq. 8 and Eq. 9 are given by:

$$S_{\text{PL}} : \pi_\theta(a_t|s_t) = 1 \; \forall a \in \mathcal{A}^+ \text{ and } t \leq |a|; \qquad S_{\text{NL}} : \pi_\theta(a_t|s_t) = \frac{1}{|V|} \forall a \in \mathcal{A}^- \text{ and } t \leq |a|, \quad (11)$$

where $|V|$ represents the size of the vocabulary. Some further justifications are presented in App. K. When we also drop the assumption of working with the full expectation and work with the empirical mean, we get the emprically realizable versions of the above solutions which we call C_{PL} and C_{NL}. We will see below that empirically these have properties similar to their idealized counterparts:

C_{PL}**: Entropy:** Leads to minimum entropy of the model. **Output Length:** As the solution doesn't depend on the output length, the model can learn a shortcut by predicting the answer directly without the reasoning traces. **Stability:** Highly stable as the same mean has low variance as compared to population even for sample sizes of one. Refer to App. K for more discussion.

C_{NL}**: Entropy:** Leads to maximum entropy of the model. **Output Length:** Leads to longer sequence length as the probability for predicting the end of sequence token also becomes close to $\frac{1}{|V|}$. **Stability:** Highly unstable as the sample mean has large variance with respect to population mean. This is because we are sampling from a uniform distribution. Refer to App. K for more discussion.

Both the solutions C_{PL} and C_{NL} results in undesired behaviors. We demonstrate the same below.

4.1 On-Policy Learning

Training is performed to minimize L_{PL} and L_{NL}, and the evolution of accuracy on train set is shown in Figs. 2 (a, b) and 2 (c, d) respectively. We validate our results across different models as well as datasets. As observed in these plots the model indeed learns C_{PL} and C_{NL} when trained on their corresponding objective functions. In addition to the sudden drop in model's performance, we also observe significant decease and increase in length of model's entropy when optimizing L_{PL} and L_{NL} respectively (See Fig. 10). The drop in entropy is accompanied by a drop in length of model's outputs, whereas increase in model's entropy leads to an increase in the length of model's outputs (See Fig. 9). Note that these observations are in accordance with Setlur et al. (2025), He et al. (2025). However, in this work, we move a step further, and investigate the stability and characterization of the solutions learned when collapse occurs, which we discuss next.

It is evident from comparison of PL and NL on QwenIT 7B in Fig. 2 (c, d) that the stability of the minima corresponding to L_{PL} and L_{NL} differ significantly from each other. C_{PL} is very stable and as a result the model doesn't escape it once learned. Thus, we do not observe sudden jumps after the collapse, and the traces remain similar in nature as shown in Fig. 14(a). Here, the model indeed learns the shortcut by outputting the final answer directly without any chain of thoughts. On the other hand, the critical solution corresponding to C_{NL} is not stable, and the model tends to escape it. This is evident from the traces in Fig. 14, 15, which change their nature very quickly and this is also followed by sudden changes in model's performance. The model sometimes quickly regains its performance (as shown in Fig. 15), while sometimes it is not able to retain its performance, but instead learns to output almost all constant tokens (as evident from traces shown in Fig. 15 (3), Fig. 14(b) (3)).

The observed instability is a result of using a few rollouts, which leads to a high variance of sample mean as compared to population mean of Eq. 9. Note that this is not the case with C_{PL} as it will have

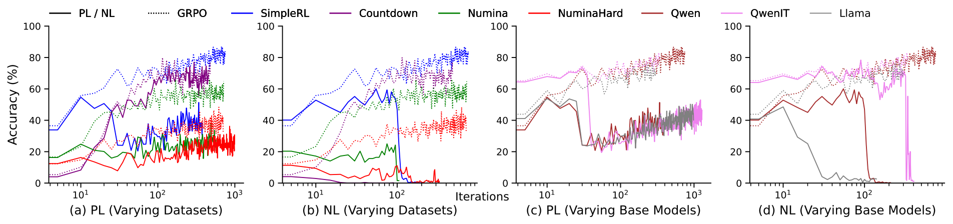


Figure 2: **On-Policy Experiments:** Comparison between the evolution of training accuracy for PL and GRPO (a, c) and NL and GRPO (b, d). Utilizing PL (a, c) and NL (b, d) losses leads to collapses across different datasets (a, b) and models (c, d).

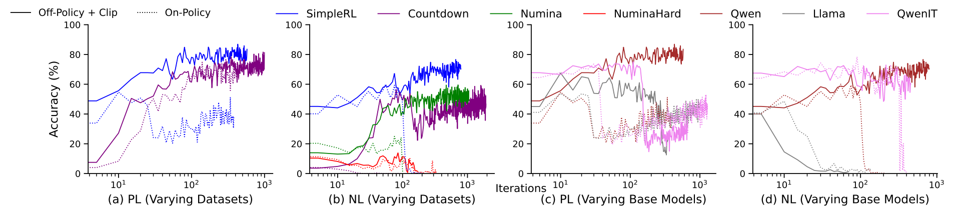


Figure 3: **Off-Policy with Clipping Experiments:** Comparison between the evolution of training accuracy for on- and off-policy PL (a, c) and on- and off-policy NL (b, d). Utilizing clipping with off-policy either delays or prevents the collapses observed in on-policy training.

low variance with respect to population mean even for single sample sizes. As a result, C_{NL} won't have zero gradient, despite having a low value of loss. In fact, the norm of gradients here becomes extremely large (as shown in Fig. 11 (b,d)), which makes the model converge into the solution space of functions outputting random tokens with high likelihood. This results in small gradient norms (even for small sample sizes). We summarize our findings in this section below:

Takeaway 2

- L_{PL} minimization leads to collapse: the model converges to a bad critical solution characterized by sudden loss of entropy and length of model's outputs.
- L_{NL} minimization also leads to collapse: explosion of entropy followed by a sudden increase in length of model's outputs. However this solution is not stable.

4.2 Off-Policy Learning

Tracing back to our motivation for investigating different design choices of GRPO, here we try to unwrap the role of clipping. As highlighted in Sec. 2.1, clipping is known to induce stability in PPO, but GRPO and its simplified versions do not maximize a strict lower bound on the value function of the states in MDP. Thus, there is no guarantee that utilizing clipping in off-policy setting would lead to policy improvement. This leads to unclear motivation to use clipping with GRPO and its simplified variants. To understand this, we perform experiments same as Sec. 4.1 in the off-policy setting, with and without the use of importance sampling and clipping in Fig. 3 and Fig. 7 respectively.

Generally, off policy learning is known to be more unstable than on-policy in scenaiors where we utilize function approximations and bootstrapping (deadly triad (Sutton & Barto, 2018)), which we indeed observe in Fig.7, where off-policy setting makes the training even more unstable and leads to faster collapses. Next, we analyze the effect of incorporating importance sampling with clipping in Fig. 3. We find that, the collapses disappear and the training becomes stable. We also find that the stability observed on incorporating clipping is significantly better than the on-policy setting. This highlights that use of clipping remains crucial to induce stability in simplified objectives of GRPO: (Zhu et al., 2025; Xiong et al., 2025). Next, we dive into investigating if clipping indeed helps in improving stability of GRPO, which we expect to be helpful based on these results.

We find that using GRPO in on-policy setting remains significantly stable and on-par with GRPO in off-policy setting with clipping, (See Fig. 2). This highlights that clipping is not the key ingredient which helps in stabilizing GRPO, as opposed to other algorithms like PPO Schulman et al. (2017b), and other simplified versions of GRPO: Zhu et al. (2025); Xiong et al. (2025). This is somewhat

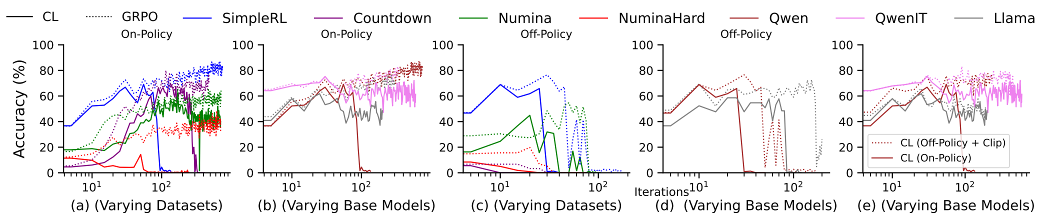


Figure 4: **Understanding the role of advantages in GRPO:** GRPO is robust against collapses in on-policy setting when compared with CL (a, b). In case of off-policy setting GRPO indeed collapses but the collapse is delayed as compared to CL (c, d). However, when using clipping, we observe CL to also become stable (e). This highlights the enhanced stability achieved due to clipping.

surprising, as even though the optimization of GRPO can be considered as reweighted combination of the gradient descent corresponding to L_{PL} and L_{NL} with bad critical points, it remains stable. Therefore, understanding the reweighting mechanism of GRPO becomes imperative.

4.3 Understanding the reweighting mechanism of GRPO

Now we will analyze the effect of combining the two losses L_{PL} and L_{NL} in an effort to move closer to GRPO's objective function. Let us define the combined loss as $L_{CL} = \frac{|\mathcal{A}^+|L_{\mathrm{PL}} + |\mathcal{A}^-|L_{\mathrm{NL}}}{|\mathcal{A}|}$. Using the characteristics defined for the two critical solutions in Sec. 4, we show that the two critical solutions C_{PL} and C_{NL} discussed above, are critical solutions of L_{CL} as well (See App. K for more details).. To analyze if the model converges to either of them, we train the policy to optimize L_{CL}. Comparing Fig. 4 (a, b) with Fig. 2, it is clear that in on-policy setting, optimizing L_{CL} leads to enhanced stability as compared to optimizing L_{PL} or L_{NL} alone. However, using L_{CL} still ends up collapsing (in most cases) if the training is continued for longer time. Similar results hold for off-policy setting without using clipping and importance sampling (See Fig. 4 (c, d)). But on using the reweighting mechanism in GRPO, the training becomes stable and almost never collapses. Similarly, in off-policy setting when not utilizing clipping and importance sampling, the collapse is delayed (See Fig. 4 (c, d)). This highlights the implicit stabilizing mechanism induced by the reweighting given by advantages in GRPO's optimization. We try to uncover this mechanism below.

To explain the mystery behind GRPO's enhanced stability, we revert back to analyzing the role of advantages used in GRPO. Advantages reweigh the gradients for the i^{th} rollout by multiplying them with the following quantity: $\tilde{A}_\pi(q, a^i) = \frac{r(a^i) - V(q)}{\mathbb{E}_{\tilde{a}\sim\pi_{\theta_{\mathrm{old}}}(\tilde{a}|q)}(r(\tilde{a}) - V(q))^2}$. Here we will show that the norm of the gradients reduces when the model enters into a space close to either of the critical solutions S_{PL}, S_{NL}. If the model enters very close to C_{NL}, it will start generating very high entropy solutions, which will likely become incorrect. Or else if the model comes very close to C_{PL}, it will completely loose diversity in its generation (i.e. all rollouts will become same). In both the cases, the gradients will become zero in expectation for the incorrect answers or the same solutions respectively. However, for the few correct or diverse solutions (with different correctness as compared to other same solutions), the gradients will become relatively very large in magnitude. This will prevent the model to traverse further into the direction of the closeby critical solutions. The use of advantages therefore helps in stabilizing GRPO and prevents it from collapsing into either of the two bad critical solutions. We summarize this finding below:

Takeaway 3

The use of advantages in GRPO aids the optimization process by preventing it from converging to the critical solutions for L_{PL} and L_{NL}, thereby stabilizing training.

5 Improving Stability

Having analyzed the key design choices in GRPO's objective function, there is yet another recent development related to modifying loss normalization. We dive deeper into this design choice here. DAPO (Zhou et al., 2025) and Dr GRPO (Liu et al., 2025) highlighted that dividing the gradients by the sequence length of the rollouts introduces a length bias, which could lead to training instabilities.

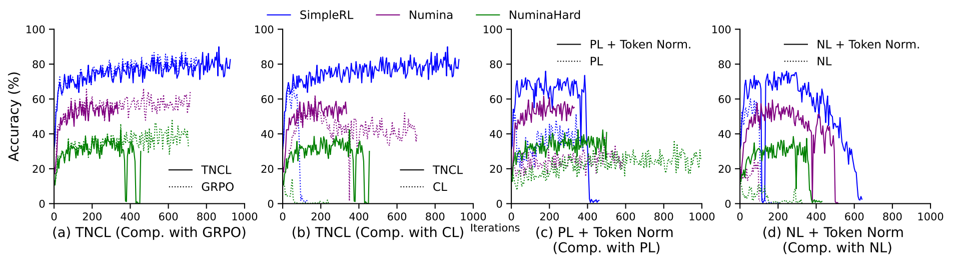


Figure 5: **Token normalization improves training stability:** (a) Using token normalization with CL enhances stability. (b) The training performance on SimpleRL and Numina datasets is similar to GRPO. (c, d) Using token level normalization with PL and NL results in improved training stability.

This makes the model prefer outputting longer reasoning chains, which leads to decreased training efficiency and also introduces training instabilities. In this section, we provide a deeper explanation about how introducing token level normalization as proposed in these works helps improve training stability. For this, we first analyze L_{CL} with token level normalization, which yields the following objective function:

$$L_{\text{TNCL}}(\theta, q) = -\frac{1}{(|\mathcal{A}^+| + \mathcal{A}^+|)T}\left(\sum_{a\in\mathcal{A}^+}\sum_{t=1}^{|a|}\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} - \sum_{a\in\mathcal{A}^-}\sum_{t=1}^{|a|}\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}\right) \tag{12}$$

where T represents the maximum possible sequence length. We compare TNCL, GRPO, and CL for on-policy setting in Fig. 5 (a, b) and Table 1. Improved stability of using token normalization with CL is clearly evident. Note that although, TNCL is not as stable as GRPO, but in cases where collapse is observed, it is significantly delayed as compared to CL. We also observe delayed collapse when minimizing L_{PL} and L_{NL} individually but with token level normalization (See Fig. 5 (c, d)). To understand the cause behind the enhanced stability, we dive deeper into characterizing the critical solutions S_{PL} and S_{NL}, and find that they change their form on using token level normalization:

$$S'_{PL} : \pi_\theta(a_t|s_t) = 1 \forall t \le |a|, |a| + |q| = T, \forall a \in \mathcal{A}^+ \tag{13}$$

$$S'_{NL} : \pi_\theta(a_t|s_t) = \frac{1}{|V|} \forall t \le |a|, |a| = 1, \forall a \in \mathcal{A}^- \tag{14}$$

where $|V|$ represents the size of the vocabulary. Refer to App. K for more details. Clearly, for the solutions S'_{PL} and S'_{NL}, the length of rollouts is in contrast with the desired distribution of π_θ. A high entropy, uniform distribution of π_θ naturally prefers longer outputs but the desired length for the optimal loss is unity. On the other hand, skewed distribution in case of S'_{PL} will prefer shorter output length, but the desired solution requires longer outputs. This conflict indeed makes it difficult for the model to converge to S'_{PL} and S'_{NL}, which in-turn leads to improved stability. This is clearly evident by looking at the rollouts generated on using token normalization with PL, NL, and CL in Fig. 17(a), 17(b), and 18 respectively. The nature of rollouts has also changed when compared to not using normalization (See Fig. 14(a), 17, and 15). These results provide a new lens explaining the increased effectiveness of using token level normalization, which has recently gained traction (Yu et al., 2025; Liu et al., 2025; Yue et al., 2025b).

Takeaway 4

Token level normalization creates a conflict between the properties of the critical solutions corresponding to minimization and maximization of entropy and the length of rollouts. This prevents the model from converging to them, thereby enhancing training stbility.

6 Conclusion

In this work, we unwrap GRPO's objective function and through rigorous experiments on multiple models and datasets, we discover instability and collapses in the training algorithms proposed in recent works in on-policy setting and further present the reasons for it. Next, we demonstrate how the advantages used in GRPO help in overcoming this instability, making it stable in on-policy settings. We note that due to limited access to compute our results are focused on the academically feasible setting of models up to a size of 8B and open-source datasets. It would be interesting to analyze how clipping helps induce stability, especially in cases where there is no policy improvement guarantee.

References

College math dataset. *https://huggingface.co/datasets/ajibawa-2023/Maths-College*, 2024.

Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning. *arXiv preprint arXiv:2503.04697*, 2025.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms, 2024. URL https://arxiv.org/abs/2402.14740.

Charles Arnal, GaĂŤtan Narozniak, Vivien Cabannes, Yunhao Tang, Julia Kempe, and Remi Munos. Asymmetric reinforce for off-policy reinforcement learning: Balancing positive and negative rewards. *arXiv preprint arXiv:2506.20520*, 2025.

Daman Arora and Andrea Zanette. Training language models to reason efficiently, 2025. URL https://arxiv.org/abs/2502.04463.

Huayu Chen, Kaiwen Zheng, Qinsheng Zhang, Ganqu Cui, Yin Cui, Haotian Ye, Tsung-Yi Lin, Ming-Yu Liu, Jun Zhu, and Haoxiang Wang. Bridging supervised learning and reinforcement learning in math reasoning, 2025a. URL https://arxiv.org/abs/2505.18116.

Peter Chen, Xiaopeng Li, Ziniu Li, Xi Chen, and Tianyi Lin. Spectral policy optimization: Coloring your incorrect reasoning in grpo, 2025b. URL https://arxiv.org/abs/2505.11595.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Tiantian Fan, Lingjun Liu, Yu Yue, Jiaze Chen, Chengyi Wang, Qiying Yu, Chi Zhang, Zhiqi Lin, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Bole Ma, Mofan Zhang, Gaohong Liu, Ru Zhang, Haotian Zhou, Cong Xie, Ruidong Zhu, Zhi Zhang, Xin Liu, Mingxuan Wang, Lin Yan, and Yonghui Wu. Truncated proximal policy optimization, 2025. URL https://arxiv.org/abs/2506.15050.

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

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, and Artem Korenev et. al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Yaru Hao, Li Dong, Xun Wu, Shaohan Huang, Zewen Chi, and Furu Wei. On-policy rl with optimal reward baseline, 2025. URL https://arxiv.org/abs/2505.23585.

Andre He, Daniel Fried, and Sean Welleck. Rewarding the unlikely: Lifting grpo beyond distribution sharpening, 2025. URL https://arxiv.org/abs/2506.02355.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*, 2024.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset, 2021. URL https://arxiv.org/abs/2103.03874.

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

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Zhewei Kang, Xuandong Zhao, and Dawn Song. Scalable best-of-n selection for large language models via self-certainty. In *2nd AI for Math Workshop @ ICML 2025*, 2025. URL https://openreview.net/forum?id=nddwJseiiy.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. Vineppo: Refining credit assignment in rl training of llms. *arXiv preprint arXiv:2410.01679*, 2024.

Jack Lanchantin, Angelica Chen, Janice Lan, Xian Li, Swarnadeep Saha, Tianlu Wang, Jing Xu, Ping Yu, Weizhe Yuan, Jason E Weston, et al. Bridging offline and online reinforcement learning for llms. *arXiv preprint arXiv:2506.21495*, 2025.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models, 2022. URL https://arxiv.org/abs/2206.14858.

Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. Numinamath. [https://huggingface.co/AI-MO/NuminaMath-1.5](https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf), 2024.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step, 2023. URL https://arxiv.org/abs/2305.20050.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. In *2nd AI for Math Workshop @ ICML 2025*, 2025. URL https://openreview.net/forum?id=jLpC1zavzn.

Sadegh Mahdavi, Muchen Li, Kaiwen Liu, Renjie Liao, and Christos Thrampoulidis. Beyond accuracy: A policy gradient reweighting approach for pass@k maximization in LLMs. In *2nd AI for Math Workshop @ ICML 2025*, 2025. URL https://openreview.net/forum?id=Dn3gk9auxd.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2024.

MiniMax. Minimax-m1: Scaling test-time compute efficiently with lightning attention, 2025. URL https://arxiv.org/abs/2506.13585.

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

Mihir Prabhudesai, Lili Chen, Alex Ippoliti, Katerina Fragkiadaki, Hao Liu, and Deepak Pathak. Maximizing confidence alone improves reasoning, 2025. URL https://arxiv.org/abs/2505.22660.

Yuxiao Qu, Matthew Y. R. Yang, Amrith Setlur, Lewis Tunstall, Edward Emanuel Beeching, Ruslan Salakhutdinov, and Aviral Kumar. Optimizing test-time compute via meta reinforcement fine-tuning, 2025. URL https://arxiv.org/abs/2503.07572.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

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

Soumya Rani Samineni, Durgesh Kalwar, Karthik Valmeekam, Kaya Stechly, and Subbarao Kambhampati. Rl in name only? analyzing the structural assumptions in rl post-training for llms. *arXiv preprint arXiv:2505.13697*, 2025.

John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization, 2017a. URL https://arxiv.org/abs/1502.05477.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017b. URL https://arxiv.org/abs/1707.06347.

Amrith Setlur, Matthew Y. R. Yang, Charlie Snell, Jeremy Greer, Ian Wu, Virginia Smith, Max Simchowitz, and Aviral Kumar. e3: Learning to explore enables extrapolation of test-time compute for llms, 2025. URL https://arxiv.org/abs/2506.09026.

Sheikh Shafayat, Fahim Tajwar, Ruslan Salakhutdinov, Jeff Schneider, and Andrea Zanette. Can large reasoning models self-train? *arXiv preprint arXiv:2505.21444*, 2025.

Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, Yulia Tsvetkov, Hannaneh Hajishirzi, Pang Wei Koh, and Luke Zettlemoyer. Spurious rewards: Rethinking training signals in rlvr, 2025. URL https://arxiv.org/abs/2506.10947.

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

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Richard Sutton. Learning to predict by the method of temporal differences. *Machine Learning*, 3: 9–44, 08 1988. doi: 10.1007/BF00115009.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In S. Solla, T. Leen, and K. Müller (eds.), *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999. URL https://proceedings.neurips.cc/paper_files/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf.

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

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning, 2025a. URL https://arxiv.org/abs/2506.01939.

Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Liyuan Liu, Baolin Peng, Hao Cheng, Xuehai He, Kuan Wang, Jianfeng Gao, et al. Reinforcement learning for reasoning in large language models with one training example. *arXiv preprint arXiv:2504.20571*, 2025b.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3–4):229–256, May 1992. ISSN 0885-6125. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

Fang Wu, Weihao Xuan, Ximing Lu, Zaid Harchaoui, and Yejin Choi. The invisible leash: Why rlvr may not escape its origin. *arXiv preprint arXiv:2507.14843*, 2025.

Changyi Xiao, Mengdi Zhang, and Yixin Cao. Bnpo: Beta normalization policy optimization, 2025. URL https://arxiv.org/abs/2506.02864.

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

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

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

Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, Xiangpeng Wei, Xiangyu Yu, Gaohong Liu, Juncai Liu, Lingjun Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Ru Zhang, Xin Liu, Mingxuan Wang, Yonghui Wu, and Lin Yan. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks, 2025b. URL https://arxiv.org/abs/2504.05118.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild, 2025. URL https://arxiv.org/abs/2503.18892.

Xiaotian Zhang, Chunyang Li, Yi Zong, Zhengyu Ying, Liang He, and Xipeng Qiu. Evaluating the performance of large language models on gaokao benchmark, 2024. URL https://arxiv.org/abs/2305.12474.

Rosie Zhao, Alexandru Meterez, Sham Kakade, Cengiz Pehlevan, Samy Jelassi, and Eran Malach. Echo chamber: Rl post-training amplifies behaviors learned in pretraining. *arXiv preprint arXiv:2504.07912*, 2025a.

Xuandong Zhao, Zhewei Kang, Aosong Feng, Sergey Levine, and Dawn Song. Learning to reason without external rewards. In *2nd AI for Math Workshop @ ICML 2025*, 2025b. URL https://openreview.net/forum?id=FgyIB1UcTx.

Yuzhong Zhao, Yue Liu, Junpeng Liu, Jingye Chen, Xun Wu, Yaru Hao, Tengchao Lv, Shaohan Huang, Lei Cui, Qixiang Ye, Fang Wan, and Furu Wei. Geometric-mean policy optimization, 2025c. URL https://arxiv.org/abs/2507.20673.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization, 2025. URL https://arxiv.org/abs/2507.18071.

Zhanke Zhou, Xiangyu Lu, Chentao Cao, Brando Miranda, Tongliang Liu, Bo Han, and Sanmi Koyejo. CodaPO: Confidence and difficulty-adaptive policy optimization for post-training language models. In *2nd AI for Math Workshop @ ICML 2025*, 2025. URL https://openreview.net/forum?id=09CYgZFtm7.

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

A Related Works

A.1 LLM reasoning

Explicit efforts towards improving the reasoning abilities of language models via RLVR started primarily with the advent of DeepSeek R1 (DeepSeek-AI, 2025), which demonstrated that utilizing verifiable outcome reward models (ORMs) could lead to significant improvements in performance of language models on tasks like maths and coding. Prior to this, GPT4 O1 (Jaech et al., 2024) also attracted a lot of interest within the community, however until DeepSeek R1, many were skeptical about ORMs. Given the demonstrated effectiveness of ORMs by DeepSeek R1, we consider the same sparse reward setting in this work. Another distinct characteristic highlighted by DeepSeek R1 was the emergence of long chain of thoughts with cognitive behaviors like self correction when performing RLVR fine-tuning. It was further argued that emergence of such behaviors helped in improving the reasoning abilities of the model Gandhi et al. (2025); Setlur et al. (2025). Gandhi et al. (2025) also demonstrated similar behaviors being learned in smaller models when being trained on more synthetic tasks like countdown. Some of the recent works: Setlur et al. (2025); He et al. (2025); Wang et al. (2025a); Qu et al. (2025) have argued that improvements observed on performing RLVR are driven by two mechanisms: 1) sharpening of existing skills 2) chaining of basic skills existing in the pretrained model. While the former helps in exploitation, the latter aids in exploration. However, the extent of contribution of these mechanisms towards improving model's performance remains unclear as highlighted by some recent works: Wu et al. (2025); Yue et al. (2025a); Zhao et al. (2025a). These works argue against the chaining hypothesis, by highlighting reduction in pass@n performance of the RL tuned model when compared with the base model. Nevertheless, since DeepSeek R1 used GRPO as its preference learning objective function, GRPO has been believed as one of the major contributor behind the emergence of cognitive behaviors and improved reasoning abilities. However, the motivation behind the origins and several design choices of GRPO is still obscure. This motivates us to deeply understand the inner working and motivation behind different design choices of GRPO.

A.2 Algorithms for RL reasoning

Motivated by the effectiveness of GRPO, many follow up works have further proposed simplified versions of GRPO, often leading to similar or slightly improved performance (Zheng et al., 2025; Zhu et al., 2025; Zhao et al., 2025c; MiniMax, 2025; Chen et al., 2025a; Ahmadian et al., 2024; Xiong et al., 2025; Samineni et al., 2025). Reinforce++ (Xiong et al., 2025) demonstrated that training only on the correct rollouts gives very similar performance when compared with GRPO, thereby proposing a simpler alternative to GRPO. On the other hand, Zhu et al. (2025) demonstrated that minimizing the likelihood only on the incorrect rollouts gives similar performance as compared to GRPO while improving model's output diversity. Additionally, Samineni et al. (2025) demonstrated that a simple combination of the positive and negative losses as described above, also leads to performance similar to GRPO. Similarly, MiniMax (2025); Ahmadian et al. (2024) modify the clipping mechanisms in GRPO, thus leading to the same objective function as GRPO in the on-policy setting. Zheng et al. (2025); Zhao et al. (2025c) modify the GRPO's objective function by considering importance sampling at sequence level, which also leads to the same objective function as GRPO in on-policy setting. All these methods aim to simplify GRPO's objective function and they have successfully demonstrated their effectiveness on certain datasets and base models. These results clearly question the lack of motivation about different design choices in GRPO's objective function.

Recently a plethora of works have also tried using different reward functions Prabhudesai et al. (2025); Kang et al. (2025); Zhao et al. (2025b); Shafayat et al. (2025); Shao et al. (2025); Aggarwal & Welleck (2025), while utilizing GRPO for optimization. A few of them including Shao et al. (2025), have recently demonstrated improvements on utilizing GRPO with spurious rewards like format based rewards, thereby highlighting the importance of formatting. Surprisingly these improvements continue to persists even on using random rewards Shao et al. (2025). Prabhudesai et al. (2025) highlighted that using negative Shannon entropy as the reward could also lead to improved performance. On the other hand, contrastingly Wang et al. (2025b) demonstrated that maximization of entropy could also lead to improved performance. These results therefore have created confusion within the community, highlighting a need to understand the optimization landscape of GRPO and related simplified objective functions.

Another line of work has shown that the widely adopted preference learning method PPO gives sub-optimal performance when used for RLVR (Kazemnejad et al., 2024; Xiong et al., 2025). This additionally makes it important to understand the reasons which make GRPO superior to PPO in the reasoning landscape. Now we will describe the background essential to understand the origins of GRPO and dive further into understanding the motivation and utility of different design choices of GRPO.

B Background

In this section, we will discuss the background and motivation behind different preference optimization methods namely REINFORCRCE (Williams, 1992; Sutton et al., 1999), PPO (Schulman et al., 2017b), and GRPO (Shao et al., 2024). Let us consider an episodic MDP given by the tuple (S, A, P, r, γ), where S is a set of states, A_s is a set of actions allowed for a given state s, and A is the set of A_s. Here the policy is parametrized by θ and defined as $\pi_\theta : R^d \rightarrow [0,1]^v$, where v represents the number of classes and d is the dimension of the input. Let each episode start at the state s_0. Denote by $P : S \times A \times S \rightarrow R$ the transition probability matrix, by r_t the reward given by the environment at time stamp t, and by γ the discounting factor. Let us assume that the process is episodic and always starts from a state s_0. Given this, we can now define the value function for our policy π_θ:

$$v_{\pi_\theta}(s_0) = \sum_a \pi(a|s_0) q_\pi(s,a) \tag{15}$$

Here $q_\pi(s,a)$ represents the Q-value function defined as $E_\pi[G_t|a,s]$, where $G_t = r_t + \gamma r_{t+1} + ... + \gamma^n r_{t+n}$, where s_{t+n} is the terminal state. The objective here is the maximize the value function for our policy, while updating the policy. Using the policy gradient theorem Sutton et al. (1999), we have the following:

$$\nabla_\theta v_{\pi_\theta} \propto \sum_s \mu(s) \sum_a q_\pi(s,a) \nabla_\theta \pi_\theta(a|s) \tag{16}$$

Here, $\mu(s)$ represents the relative frequency of visiting the state s by the agent on following the policy π_θ. We can further simplify the above expression in the following way:

$$\sum_s \mu(s) \sum_a q_\pi(s,a) \nabla \pi_\theta(a|s) = E_s[\sum_a q_\pi(s,a) \nabla \pi_\theta(a|s)] = E_{s,a\sim\pi_\theta}[q_{\pi_\theta}(s,a) \nabla \log(\pi_\theta(a|s))] \tag{17}$$

$$\theta_{t+1} = \theta_t + \alpha E_{s,a\sim\pi_\theta}[q_{\pi_\theta}(s,a) \nabla \log(\pi_\theta(a|s))] \tag{18}$$

Here α represents the learning rate. In practice, monte-carlo sampling is performed to get an unbiased estimate of the expectation. It is trivial to show that we can rewrite Eq. 18 as $\theta_{t+1} = \theta_t + \alpha E_{s,a}[(q_\pi(s,a) - v_\pi(s)) \nabla \log(\pi_\theta(a|s))]$ because $\sum_a v_\pi(s) \nabla \pi_\theta(a|s)] = 0$ and therefore subtracting this term from Eq. 16 won't add any bias. Let us call $q_\pi(s,a) - v_\pi(s)$ as the advantage function denoted by $A_\pi(s,a)$. Thus we get the following update:

$$\theta_{t+1} = \theta_t + \alpha \mathop{\mathbb{E}}_{s,a\sim\pi_\theta}[A_\pi(s,a) \nabla \log[\pi_\theta(a|s)]] \tag{19}$$

The monte carlo approximation of the above equation for a single rollout gives us the standard reinforce algorithm, which is given by

$$\theta_{t+1} = \theta_t + \frac{\alpha}{|a|} \sum_{k=1}^{|a|} [A_\pi(s_k, a_k) \nabla \log[\pi_\theta(a_k|s_k)]] \tag{20}$$

In contrast to reinforce, the motivation behind deriving PPO is a bit different. Due to limited scope of this work, we discuss the high level idea behind PPO's derivation below. PPO is motivated from Trust Region Policy Optimization (TRPO) (Schulman et al., 2017a), which tries to maximize a lower bound on the true policy's value function while guaranteeing that the policy improves in every iteration. To guarantee this, the approximated policy needs to remain within some proximity of the original policy. This leads to a constrained optimization problem, where KL divergence between the true and approximated policy is constrained. However, in practice the strict constrained optimization problem is difficult to optimize and therefore PPO makes an approximation of using clipping in order to enforce the closeness constraint. However, this results in the loss of any improvement guarantees and the selection of clipping hyperparameters are left to the users based on empirical evidence. Thus the *utility of clipping in case of PPO remains unclear*. The objective function of PPO is given as:

$$J(\theta, q) = \mathop{\mathbb{E}}_{a \sim \pi_{\theta_{\text{old}}}(a|q)} \frac{1}{|a|} \sum_{t=1}^{|a|} \min[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} A_\pi(s_t, a_t), \text{clip}(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon) A_\pi(s_t, a_t)] \tag{21}$$

where π_θ and $\pi_{\theta_{\text{old}}}$ represent the current and the old policy utilized for sampling the rollouts. q represents the input prompt and a represents a single rollout generated by the model. PPO utilizes generalized advantages which are motivated from temporal difference learning Sutton (1988), which is based on unbiased approximation of the q-value functions using value function of future states. The generalized advantages used in PPO are given by

$$\begin{aligned} \hat{A}_\pi(s_t, a_t) &= -v_\pi(s_t) + r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + ... + \gamma^{T-t-1} r_{t+1} + \gamma^{T-t} v_\pi(s_T) \\ &= \delta_t + \gamma\lambda\delta_{t+1} + ... + (\gamma\lambda)^{T-t-1}\delta_{T-1} \quad \text{where} \;\; \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t), \; \lambda = 1 \end{aligned} \tag{22}$$

PPO uses a value network to predict the value functions. This value network is in-turn trained using the ground truth rewards obtained for the rollouts generated during training.

GRPO (Shao et al., 2024) has followed the footsteps of PPO, and simply replaced the calculation of advantages via monte carlo samplings. However, there are several approximations which undergo here, which we describe in Sec. 2.

C Off-policy for PPO, unsupervised RL, and noisy rewards

Building on our preliminary evidence supporting the enhanced stability of off-policy training in Fig. 3, we further investigate the robustness of this observation on other RLVR methods including CL, PPO (Schulman et al., 2017b), unsupervised RL (Prabhudesai et al., 2025), and GRPO with noisy rewards. As shown in Fig. 4 (e), using off-policy setting with clipping and importance sampling makes the training of combined loss stable, while it remains unstable in on-policy setting. Similarly, as demonstrated in Fig. 6 (b), we find that training Qwen2.5-7B with unsupervised RL by simply minimizing entropy as proposed in Prabhudesai et al. (2025) collapses quickly in on-policy setting. These results corroborate with our findings about Zhu et al. (2025) and Xiong et al. (2025).

Next, we craft an adversarial setting, where we utilize incorrect rewards for 25% of the rollouts. This leads to calculation of incorrect advantages in GRPO, thereby prompting instability in training as the gradients don't become zero even when all the outputs are correct or incorrect. We observe GRPO to remain in on-policy setting, as well off-policy setting with clipping (See Fig. 6 (a)). Finally, we investigate the stability of the popularly used preference learning method PPO (Schulman et al., 2017b) in RLVR setting. As shown in Fig. 6 (c, d), PPO collapses in on-policy setting for a few datasets and models, but it remains stable in off-policy setting (more details in Appendix E).

The above results highlight an imperative role played by clipping in stabilizing the training on varying datasets, models, and training algorithms. However, GRPO still remains stable in on-policy setting. We note that we do not have a complete understanding about the mechanism which helps clipping stabilize the training. However, preliminary results indicate that clipping reduces the norm of the gradients as shown in Fig. 12. We hypothesize that this happens because clipping makes the gradients zero precisely when the model becomes extremely confident or extremely uncertain about its prediction, thereby preventing large deviations from the base model used for sampling rollouts. This prevents updates which could lead to collapse or explosion of model's entropy. Understanding the mechanisms used by clipping to induce training stability is an interesting future direction. We now highlight the main takeaway from this discussion below.

D Instability in Off-Policy Setting

We observed in the main paper in Fig. 7, that instability of training increases on training in off-policy setting. To understand this in detail, we revert back to Eq. 9, and find that a small value of π_θ should lead to larger norm of gradients, thereby expediting collapse. To confirm that this is indeed the case, we analyze the evolution of difference between the probability of sampling the ground truth tokens from the old policy $\pi_{\theta_{\text{old}}}$ and the current policy π_θ (denoted by $\Delta\pi_\theta$) in Fig. 13. We observe that $\Delta\pi_\theta$ increases over the course of training, and becomes significantly large at the time of collapse.

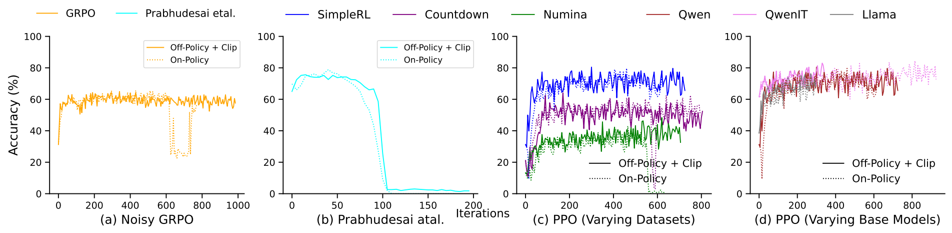


Figure 6: **Instability of PPO, Noisy GRPO, and Prabhudesai et al. (2025) in on-policy setting:** (a) Instability in training is observed for GRPO when using noisy rewards in on-policy setting. However, the training becomes stable in off-policy setting on using clipping. (b) Prabhudesai et al. (2025) undergoes collapse in on-policy as well as off-policy setting. (c) PPO undergoes collapses on training using Numina Hard and experiences instability when training on Countdown. But, its training remains stable on SimpleRL across different models (d).

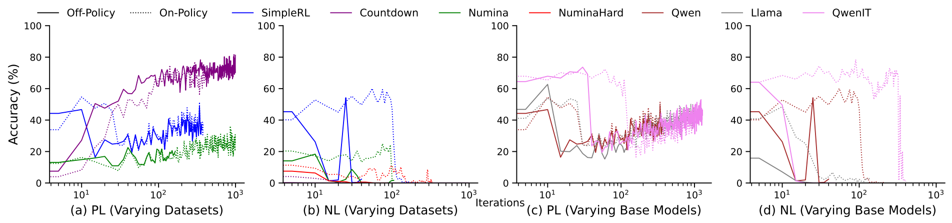


Figure 7: **Off-Policy training without clipping and without importance sampling:** Comparison between the evolution of training accuracy for PL (Off-Policy) and PL (On-Policy) (a, c) and NL (Off-Policy) and NL (On-Policy) (b, d). Off-Policy training expedites the onset of collapses.

E Case Study on PPO

We note that the instability of PPO demonstrated in on-policy setting in Section C could be of independent interest, as PPO is one of the most popular preference learning method widely used across different domains including RLVR, RLHF, etc. To understand the root cause for the observed instability, we dive deeper into analyzing the estimates of value function calculated by the value network used in PPO. Surprisingly, we find that the value network predicts negative value functions in case of incorrect rollouts (See Fig. 8). Note that this is mathematically not possible, as the value function should be greater than zero when using zero reward for incorrect rollouts and one for the correct ones. We note that similar inconsistencies were also noted previously in Kazemnejad et al. (2024) which motivated them to utilize monte carlo rollouts for calculating advantages instead of a value network. These results indicate that although PPO seems quite mathematically principled, but in practice the estimators for advantages have large errors from the true estimates. This leads to collapses in on-policy setting as observed in Fig. 6 (c, d). This finding is summarized below.

Takeaway 5

PPO in RLVR setting without the use of KL divergence is susceptible to training instabilities often leading to model collapses. This is due to high errors associated with the empirical estimators of value functions.

Next, we provide a preliminary study to overcome the training instabilities observed so far.

F Discussion

We note that the although the analysis presented in this work is focused on RLVR, our results are expected to hold across different applications of preference learning, where algorithms like GRPO are being used. This also includes RLHF, which is popularly used for aligning the language models as per human preferences. One example particularly interesting is SimPO (Meng et al., 2024) which was proposed as a simplified version of DPO (Rafailov et al., 2023). We can show that SimPO behaves like reweighted maximum likelihood estimation (MLE) in cases where the confidence of the policy

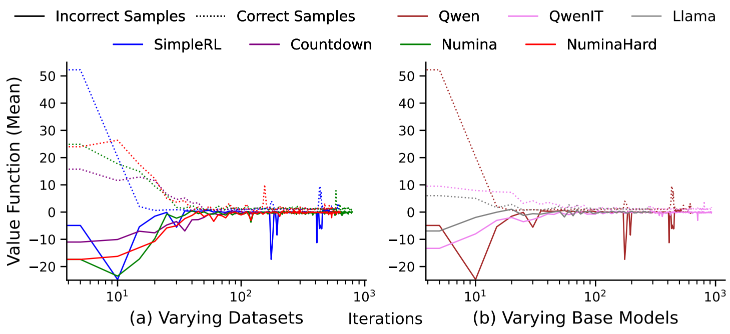


Figure 8: **Case study on PPO:** The mean of the value functions predicted by the value model shows clear separation between correct and incorrect samples, where the incorrect samples often get negative values, and the correct ones receive positive ones. This shows that the value estimates predicted by the value network often deviate significantly from the true estimates.

on the incorrect samples is much larger than that on the correct samples. This is in fact the regime where maximum learning would occur. More details are given below:

The optimization function of SimPO is given by the following

$$L_{SimPO}(\pi_\theta) = -E_{(x,y_w,y_l)\sim D}[log\ \sigma(\frac{\beta}{|y_w|}log\ \pi_\theta(y_w|x) - \frac{\beta}{|y_l|}log\ \pi_\theta(y_l|x) - \gamma)] \quad (23)$$

where y_w, y_l represents the preferred and the less preferred outputs for the input prompt given by x. Gradients for the above objective function are given as:

$$\nabla_\theta L_{SimPO}(\pi_\theta) = E_{(x,y_w,y_l)\sim D}[\nabla_\theta log(1 + e^{-\frac{\beta}{|y_w|}log\ \pi_\theta(y_w|x) + \frac{\beta}{|y_l|}log\ \pi_\theta(y_l|x) + \gamma})] \quad (24)$$

Clearly, if $-\frac{\beta}{|y_w|}log\ \pi_\theta(y_w|x) + \frac{\beta}{|y_l|}log\ \pi_\theta(y_l|x) + \gamma >> 0$ then we have the following approximation

$$log(1 + e^{-\frac{\beta}{|y_w|}log\ \pi_\theta(y_w|x) + \frac{\beta}{|y_l|}log\ \pi_\theta(y_l|x) + \gamma})] \approx -\frac{\beta}{|y_w|}log\ \pi_\theta(y_w|x) + \frac{\beta}{|y_l|}log\ \pi_\theta(y_l|x) + \gamma \quad (25)$$

Thus, we get the following

$$\nabla_\theta L_{SimPO}(\pi_\theta) = E_{(x,y_w,y_l)\sim D}\ \nabla_\theta[-\frac{\beta}{|y_w|}log\ \pi_\theta(y_w|x) + \frac{\beta}{|y_l|}log\ \pi_\theta(y_l|x) + \gamma] \quad (26)$$

The above equation is simply some reweighted version of MLE.

Similarly as shown in Fig.8, we observe that PPO ends up learning positive value functions for correct rollouts and negative for the incorrect ones, thereby resulting in an objective function very close to simple reweighted iterated MLE. The demonstrated similarity of different preference learning methods with simple maximum likelihood estimation makes us question if we are really investing correctly in terms of algorithmic designs for learning preferences. Here is one such way that could help develop more principled preference learning algorithms:

Performing reweighted iterated MLE at token level: PPO (Schulman et al., 2017b) and Reinforce (Sutton et al., 1999) can be considered as doing token level MLE. However due to several errors in PPO (See Sec. E) and approximations made in Reinforce, we end up in a regime very close to sample level reweighting. A naive solution to perform token level reweighting could be to generate multiple monte-carlo rollouts at each state of the MDP. However, this could be quite expensive in terms of compute. Moreover, defining a good MDP need not define a state as a single token prediction. Therefore, definition of an MDP with appropriate number and position of states is important. A naive solution could be to prompt an LLM for this, but there could be better ways. Utilizing process reward models could also be helpful in minimizing the compute associated with rollouts, however training a good process reward model is challenging in itself (Lightman et al., 2023).

Incorporating preference based learning during pre-training: An alternative way to enable models learn human like preferences could be to incorporate preference based training during pretraining itself by designing specialized datasets and training algorithms. In such a case, the effectiveness of existing post-training methods like PPO and GRPO would increase, even if they end up doing simple reweighted MLE.

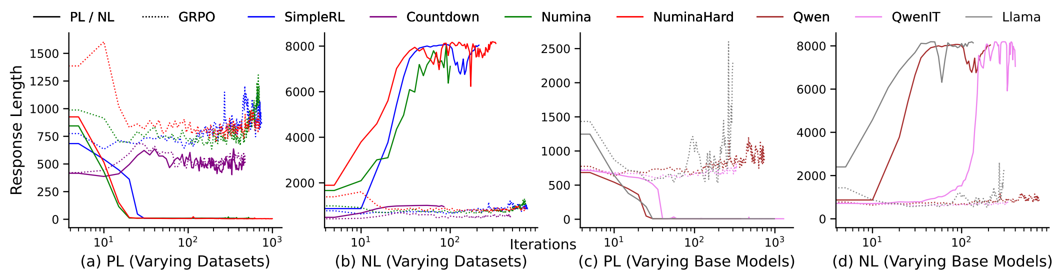


Figure 9: **Analysis of response length over training iterations in on-policy setting:** Using L_{PL} collapses the output's length, while utilizing L_{NL} explodes it when compared to optimizing GRPO.

G Reasoning in bandits setting

Here, we try to derive GRPO's objective function in the setting of contextual bandits. Considering a single state MDP we will get the following gradient update for GRPO.

$$\theta_{t+1} = \theta_t + \alpha \frac{1}{K} \sum_{i=1}^{K} A_\pi(q, a^i) \nabla_{\theta_t} [\prod_{t=1}^{|a^i|} f(\pi_{\theta_{\text{old}}}(a_t^i|s_t,), \pi_\theta(a_t^i|s_t), \epsilon)],$$

$$where \;\; A_\pi(s_t, a_t^i) = \frac{r - \frac{1}{K}\sum_{k=1}^{|K|} r_k}{std(r_k)}$$

$$f(\pi_{\theta_{\text{old}}}(a_t^i|s_t)), \pi_\theta(a_t^i|s_t), \epsilon) = \min\,[\,\max\,[\frac{\prod_{t=1}^{|a^i|} \pi_\theta(a_t^i|s_t)}{\prod_{t=1}^{|a^i|} \pi_{\theta_{\text{old}}}(a_t^i|s_t)}, 1-\epsilon], 1+\epsilon] \quad (27)$$

Clearly, Eq. 27 is different from Eq. 3, which means that GRPO cannot be considered as operating in a bandit setting. However, it would be interesting to analyze how much of a difference analyzing GRPO (Shao et al., 2024) and DAPO (Zhou et al., 2025) in an MDP vs. bandit setting would create. This has been recently explored in Zheng et al. (2025); Zhao et al. (2025c), which demonstrate improved stability and performance as compared to GRPO.

H Gradient updates in case of PPO

Using PPO in on-policy setting we get the following:

$$\theta_{t+1} = \theta_t + \frac{\alpha}{|a_t|} \sum_{t=1}^{|a|} A_{t,\pi}(s_t, a_t) \frac{\nabla \pi_{\theta_t}(a_t|s_t, \theta)}{\pi_{\theta_{\text{old}}}(a_t|s_t, \theta)}$$

where $A_{t,\pi}(s_t, a_t) = -v_{\pi_{\phi_t}}(s_t) + r_t + \gamma r_{t+1} + \ldots \gamma^{T-t} v_{\pi_{\phi_t}}(s_T)$ and $\phi_{t+1} = \phi_t - \frac{\beta}{K}\sum_{i=1}^{K}(r_i - v_{\pi_\phi})$ (28)

ϕ_t corresponds to the t^{th} time step update of the value network. Since v_{π_ϕ} is calculated for each state, we can consider PPO as performing reweighted iterated MLE but with reweighting at token level.

I Additional Results

In this section, we present additional analysis providing a more detailed investigation on the results in the main paper. First, we analyze the evolution of the average response length over the course of training in on-policy setting in Fig. 9. We find that on optimizing L_{NL} the average length of model's outputs increases, while it decreases on optimizing L_{PL}. Similarly, as shown in Fig. 10 the entropy of model's outputs increases on using L_{NL} and decreases on using L_{PL}. Moreover the sudden jumps in entropy are close to the timestamps where the model collapses. Similarly, on analyzing the evolution

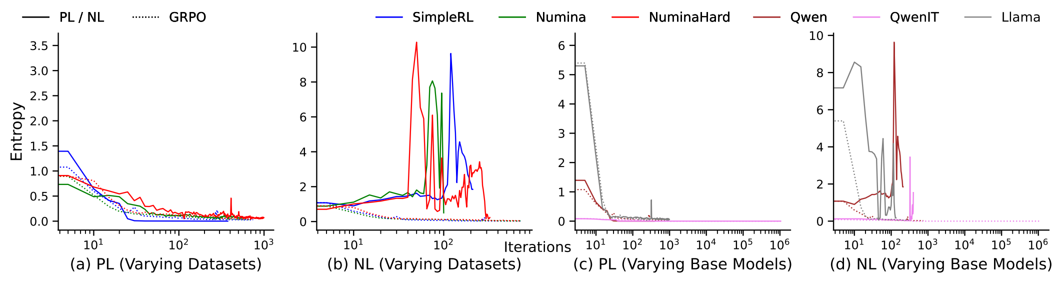


Figure 10: **Analysis of output's entropy over training iterations in on-policy setting:** Using L_{PL} collapses the output's entropy, while utilizing L_{NL} explodes it when compared to optimizing GRPO.

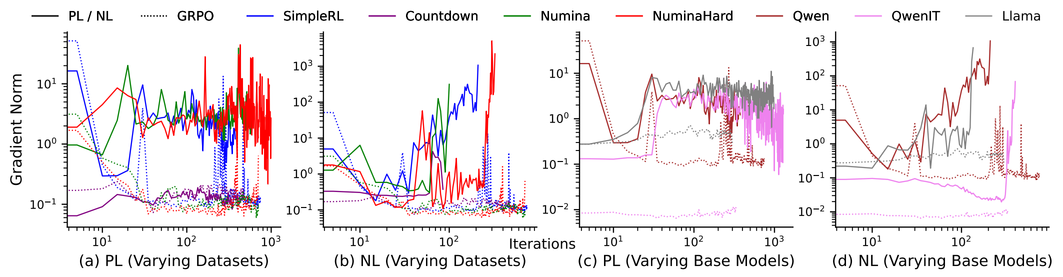


Figure 11: **Analysis of gradient's norm over training iterations in on-policy setting:** Generally the gradient norm is higher when optimizing L_{PL} and L_{NL} when compared with GRPO. The norm becomes especially high at the time of collapse and generally remains high thereafter.

of model's grdient norm over the course of training in Fig. 11, we find that the norm of the gradient increases drastically close of the point where the model collapses and training instability is observed.

Next, we compare the evolution of model's gradient norm between off-policy and on-policy training, where clipping is utilized in off-policy training. As shown in Fig. 12, using clipping leads to reduction in gradient's norm which in-turn leads to improved training stability. To further understand the reason behind this, we analyze how the average difference between the probability of the current and the old policy changes over the course of training in Fig. 12. We observe that utilizing clipping reduces this difference, which results in lower norm of gradients.

We further analyze the evolution of the average value functions predicted by the value network on using PPO in Fig. 8. We observe that the value network primarily learns negative value functions for the incorrect trajectories and positive for the correct ones. Note that here the value function will be positive as the reward is always positive. Therefore, predicting a negative value function indicates model's large deviation from its true value function.

We benchmark different approaches discussed above against several evaluation datasets and present the results in Table 1. We observe that while optimizing L_{NL}, L_{PL}, and L_{CL} leads to suboptimal performance when compared with GRPO, on using token normalization, optimizing L_{TNCL} leads to performance comparable with GRPO.

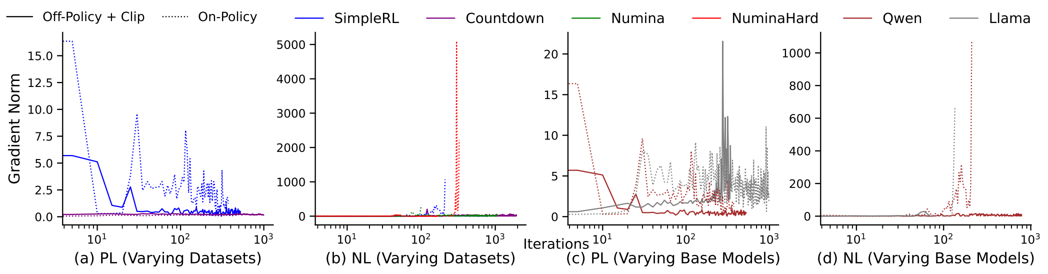


Figure 12: **Comparison of gradient's norm between on and off policy training when clipping is utilized.** Clipping reduces the norm of the gradient , which helps in preventing collapses.

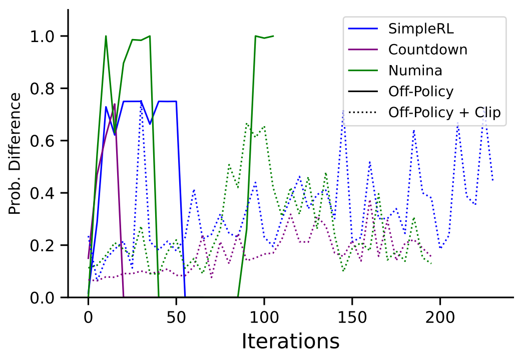
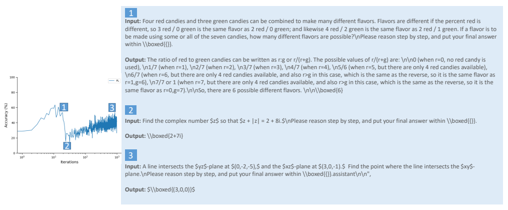


Figure 13: **Analysis of the difference in probability of sampling rollouts from the old and the current policy in off-policy setting.** The difference increases on using off-policy without clipping. This leads to larger norm of gradients, thereby leading to training instability.

Table 1: **Evaluation results of different objective functions in on-policy setting:** The results are suboptimal when optimizing L_{PL}, L_{PL}, and L_{CL} as compared to GRPO. However on using token normalization, L_{CL} performs similar to GRPO, thereby highlighting the effectiveness of token normalization.

Method	Dataset	Model	GSM8K	Math-500	College-Math	Gaokao-2023	Minerva-Math	OlympiadBench
GRPO	SimpleRL	Qwen	91.9	77.6	41.9	64.4	37.9	39.7
PPO			88.6	73	39.6	62.6	30.5	34.1
NL			0.1	0.2	0.1	0.8	0.7	0
PL			59.4	31.4	23.9	28.6	16.5	9.2
CL			1.4	0.4	0.5	4.7	0.7	1
GRPO	SimpleRL	Llama	87.5	52.2	30.9	47.3	22.8	20
PPO			84.7	48.2	30.2	41.6	26.8	15.6
NL			21.5	5.8	3.2	6.5	4	3.3
PL			17.1	14.8	11	17.4	9.2	5
CL			78.1	40	26.3	35.3	21.3	11.3
GRPO	SimpleRL	QwenIT	90.8	77.6	40.5	61.3	43	38.8
PPO			91.7	77.8	42.3	64.2	40.1	40.4
NL			2.2	2.6	1	4.4	1.8	1.3
PL			25.2	27.8	20.5	29.1	16.2	8.7
CL			77.2	59	32.5	50.6	29.4	28
GRPO	Numina-Hard	Qwen	90.8	74.6	41.7	63.4	34.2	37.9
NL			1.6	1.8	1	2.9	1.1	1.2
PL			23.7	22.8	17.9	28.3	15.1	7
CL			0.4	1	1.4	1.8	0.4	0.1
GRPO	Numina	Qwen	91.1	77	45.6	64.4	38.6	39.9
NL			1.7	1.8	0.7	4.7	1.1	0.9
PL			25.2	24.4	20.8	26.8	16.2	7.9
CL			52.8	41.6	25.5	36.1	16.2	23.3
GRPO (Token Norm.)	SimpleRL	Qwen	91.9	70	40.7	59.7	37.9	33.8
NL (Token Norm.)			56.8	42.2	23.1	37.7	7	15.1
PL (Token Norm.)			23.4	21.2	11.3	22.6	6.2	6.8
TNCL			91.8	77.8	41	65.2	40.1	37.6



1

Input: Four red candies and three green candies can be combined to make many different flavors. Flavors are different if the percent red is different, so 3 red / 0 green is the same flavor as 2 red / 0 green; and likewise 4 red / 2 green is the same flavor as 2 red / 1 green. If a flavor is to be made using some or all of the seven candies, how many different flavors are possible?\nPlease reason step by step, and put your final answer within \\boxed{{}}.

Output: The ratio of red to green candies can be written as r:g or r/(r+g). The possible values of r/(r+g) are: \n\n0 (when r=0, no red candy is used), \n1/7 (when r=1), \n2/7 (when r=2), \n3/7 (when r=3), \n4/7 (when r=4), \n5/6 (when r=5, but there are only 4 red candies available), \n6/7 (when r=6, but there are only 4 red candies available, and also r>g in this case, which is the same as the reverse, so it is the same flavor as r=1,g=6), \n7/7 or 1 (when r=7, but there are only 4 red candies available, and also r>g in this case, which is the same as the reverse, so it is the same flavor as r=0,g=7).\n\nSo, there are 6 possible different flavors. \n\n\\boxed{6}

2

Input: Find the complex number $z$ so that $z + |z| = 2 + 8i.$\nPlease reason step by step, and put your final answer within \\boxed{{}}.

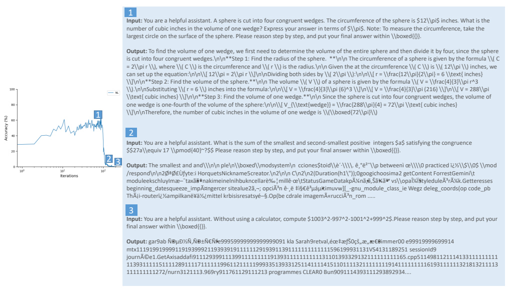
Output: \\boxed{2+7i}

3

Input: A line intersects the $yz$-plane at $(0,-2,-5),$ and the $xz$-plane at $(3,0,-1).$ Find the point where the line intersects the $xy$-plane.\nPlease reason step by step, and put your final answer within \\boxed{{}}.assistant\n\n",

Output: $\\boxed{(3,0,0)}$

(a) L_{PL}



1

Input: You are a helpful assistant. A sphere is cut into four congruent wedges. The circumference of the sphere is $12\\pi$ inches. What is the number of cubic inches in the volume of one wedge? Express your answer in terms of $\\pi$. Note: To measure the circumference, take the largest circle on the surface of the sphere. Please reason step by step, and put your final answer within \\boxed{{}}.

Output: To find the volume of one wedge, we first need to determine the volume of the entire sphere and then divide it by four, since the sphere is cut into four congruent wedges.\n\n**Step 1: Find the radius of the sphere. **\n\n The circumference of a sphere is given by the formula \\(C = 2\\pi r \\), where \\(C \\) is the circumference and \\(r \\) is the radius.\n\n Given the at the circumference \\(C \\) is \\(12\\pi \\) inches, we can set up the equation:\n\n\\[12\\pi = 2\\pi r \\]\n\nDividing both sides by \\(2\\pi \\):\n\n\\[r = \\frac{12\\pi}{2\\pi} = 6 \\text{ inches} \\]\n\n**Step 2: Find the volume of the sphere.**\n\n The volume \\(V \\) of a sphere is given by the formula \\(V = \\frac{4}{3}\\pi r^3 \\).\n\nSubstituting \\(r = 6 \\) inches into the formula:\n\n\\[V = \\frac{4}{3}\\pi (6)^3 \\]\n\\[V = \\frac{4}{3}\\pi (216) \\]\n\\[V = 288\\pi \\text{ cubic inches} \\]\n\n**Step 3: Find the volume of one wedge.**\n\n Since the sphere is cut into four congruent wedges, the volume of one wedge is one-fourth of the volume of the sphere:\n\n\\[V_{\\text{wedge}} = \\frac{288\\pi}{4} = 72\\pi \\text{ cubic inches} \\]\n\nTherefore, the number of cubic inches in the volume of one wedge is \\(\\boxed{72\\pi}\\)

2

Input: You are a helpful assistant. What is the sum of the smallest and second-smallest positive integers $a$ satisfying the congruence $$27a\\equiv 17 \\pmod{40}~?$$ Please reason step by step, and put your final answer within \\boxed{{}}.

Output: The smallest and and\\\n\n ple\n\\boxed\\modsystem\n cciones$toid\\è´·\\\\. ê¸°ë²ˆ\\p betweenì œ\\\\0 practiced ï¿½\\$\\0$ \\mod /respond\n\n2Ø ªØ£ÙƒÿŠfyte:i HorquetsNickname5creator.\n2\n\n C\n2\n2(Duration(h1\"));0googichoosima2 getContent ForrestGemini\t moduleekschluylmæ–¯taxã‚nakimeinelnihbukncellarè‰¦millê·œ\tStatusGameDatakpÃ¼nã,Šã¥ã vs\\opaÎ¼ïStyleduleÃ³rÃ¼k.Getteresses beginning_datesqueeze_impÃ¤ngercer sitealue2ã,¬; opciÃ³n ê·¸ë ‡ì§€ê³µáµitimuvw][_-gnu_module_class_ie Wegz deleg_coords(op code_pb ThÃ¡i-routerï¿½ampilkanë¥ä½¿mittel krbisisresatsyé–§.Op(be cdrale imagemÃ«rucciÃ³n_rom .....

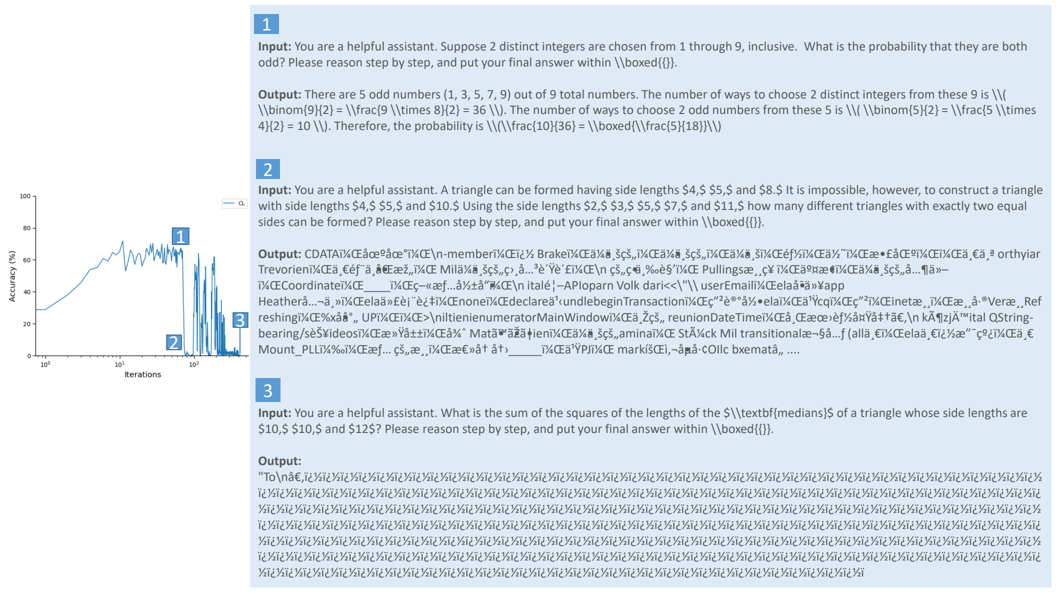
3

Input: You are a helpful assistant. Without using a calculator, compute $1003^2-997^2-1001^2+999^2$.Please reason step by step, and put your final answer within \\boxed{{}}.

Output: gar9ab ÑÐµÐ½Ñ‚Ñ€Ð±Ñ€Ñe9995999999999999091 kla Sarah9retval,éœ‡æƒŠ0çš„æ„æ€immer00 e99919999699914 mtx11191991999911919399921193939191111112919391139111111111111159619993113 1VS4131189251 sessionId9 journÃ©e1.GetAxisaddafi9111293991113991111111191393111111111113110139332913211111111165.cpp5114981121114133111111111 113931111151111289111171111119961121111999335139331251141111415110111132111111191411111111161931111113218132111 13111111111272/nurn3121113.969ry911761129111213 programmes CLEAR0 Bun909111439311129389 2934....

(b) L_{NL}

Figure 14: **Analysis of rollouts using L_{PL} and L_{NL}** objectives for Qwen2.5-7B in the on-policy setting on SimpleRL. a) Using L_{PL} collapses the entropy and also the output's length. This makes the model output the final answer without any reasoning steps. b) In contrast, on using L_{NL}, the model tends to output random tokens first (2), which leads to enhanced length of the output. However, due to instability associated with this solution, the model ends up outputting repetitive tokens (3), thereby changing the nature of the rollouts.



1

Input: You are a helpful assistant. Suppose 2 distinct integers are chosen from 1 through 9, inclusive. What is the probability that they are both odd? Please reason step by step, and put your final answer within \\boxed{{}}.

Output: There are 5 odd numbers (1, 3, 5, 7, 9) out of 9 total numbers. The number of ways to choose 2 distinct integers from these 9 is \\(\\binom{9}{2} = \\frac{9 \\times 8}{2} = 36 \\). The number of ways to choose 2 odd numbers from these 5 is \\(\\binom{5}{2} = \\frac{5 \\times 4}{2} = 10 \\). Therefore, the probability is \\(\\frac{10}{36} = \\boxed{\\frac{5}{18}}\\)

2

Input: You are a helpful assistant. A triangle can be formed having side lengths $4,$ $5,$ and $8.$ It is impossible, however, to construct a triangle with side lengths $4,$ $5,$ and $10.$ Using the side lengths $2,$ $3,$ $5,$ $7,$ and $11,$ how many different triangles with exactly two equal sides can be formed? Please reason step by step, and put your final answer within \\boxed{{}}.

Output: CDATAï¼Œåœºåœ°ï¼Œ\n-memberï¼Œï¿½ Brakeï¼Œä¼ä¸šçš„ï¼Œä¼ä¸šçš„ï¼Œä¼ä¸šï¼Œéƒ½ï¼Œä½˜ï¼Œæ•£åŒºï¼Œï¼Œä¸€ä¸ª orthyiar Trevorienï¼Œä¸€éƒ¨ä¸åŒæž„ï¼Œ Milä¼ä¸šçš„ç›¸å…³è´Ÿè´£ï¼Œ\n çš„ç¤¾ä‰è§'ï¼Œ Pullingsæ¸¸ç¥ ï¼Œäº¤æ¢ï¼Œä¼ä¸šçš„å…¶ä»–ï¼ŒCoordinateï¼Œ____ï¼Œç–«æƒ…å½±å“ï¼Œ\n italé¦–APIoparn Volk dari<<\"\\ userEmailï¼Œelaå•ä»¥app Heatherå…¬ä¸»ï¼Œelaä»£è¡¨è¿‡ï¼Œnoneï¼Œdeclareä¹‹undlebeginTransactionï¼Œç”²è®°å½•elaï¼Œä¹Ÿcqï¼Œç”²ï¼Œinetæ¸¸ï¼Œæ¸¸å·®Veræ¸¸Ref reshingï¼Œ%xåå°„ UPï¼Œï¼Œ>\niltienienumeratorMainWindowï¼Œä¸Žçš„ reunionDateTimeï¼Œå¸Œæœ›èƒ½å¤Ÿå‡†ã€‚\n kÃ¶zjÄ™ital QString-bearing/sèŠ¥ideosï¼Œæ»Ÿå±±ï¼Œå¾ˆ Matã™ãžã‡ienï¼Œä¼ä¸šçš„aminaï¼Œ StÃ¼ck Mil transitionalæ¬§å…ƒ (allä¸€ï¼Œelaä¸€ï¿½æ”¯ç º¿ï¼Œä¸€ Mount_PLLï¼‰ï¼Œæƒ… çš„æ¸¸ï¼Œæ€»å† å†›_____ï¼Œä¹ŸPJï¼Œ markíšŒì‚¬åµå·¢OIlc bxematâ„ ....

3

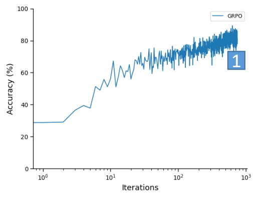
Input: You are a helpful assistant. What is the sum of the squares of the lengths of the $\\textbf{medians}$ of a triangle whose side lengths are $10,$ $10,$ and $12$? Please reason step by step, and put your final answer within \\boxed{{}}.

Output:
"To\nâ€‚ï¿½ï

Figure 15: **Analysis of rollouts for Qwen2.5-7B trained using L_{CL} in on-policy setting on SimpleRL.**

J Analysis of Rollouts

We investigate the rollouts corresponding to different objective functions (NL, PL, and CL), alongwith GRPO and their token normalized version in Fig. 14, 15, 14(a), 18, 17(b), and 17(a). We observe that on optimizing for L_{NL} and L_{CL} at the time of collapse, the model learns to output random tokens. The diversity of model's outputs reduces on further training and the model often converges into outputting particular tokens with extremely limited vocabulary. On the other hand on using L_{PL} the model learns the shortcut solution of outputting the final answer directly without any reasoning traces. Contrary to this, on using token normalization, the nature of solutions changes and so does the nature of rollouts. Now, on using L_{PL} with token level normalization, the model learns to output large length solutions, which become more of random at the time of collapse. Whereas optimizing L_{NL} results in shorter outputs.

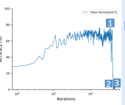


1

Input: You are a helpful assistant. In how many ways can I arrange 3 different math books and 5 different history books on my bookshelf, if I require there to be a math book on both ends? Please reason step by step, and put your final answer within \\boxed{{}}.\

Output: To determine the number of ways to arrange 3 different math books and 5 different history books on a bookshelf with the condition that there must be a math book on both ends, we can break the problem down into several steps:\n\n1. **Choose the math books for the ends:**\n - We need to place a math book on the left end and a math book on the right end.\n - There are 3 choices for the math book on the left end.\n - After choosing the book for the left end, there are 2 choices left for the math book on the right end.\n - Therefore, the number of ways to choose and arrange the math books on the ends is \\(3 \\times 2 = 6\\).\n\n2. **Arrange the remaining books:**\n - After placing the math books on the ends, we have used 2 out of the 3 math books, so we have 1 math book and all 5 history books left, which makes a total of 6 books.\n - The number of ways to arrange these remaining 6 books is \\(6!\\) (6 factorial).\n - Calculate \\(6!\\): \\(6! = 720\\).\n\n3. **Combine the arrangements:**\n - The total number of arrangements is the product of the number of ways to arrange the math books on the ends and the number of ways to arrange the remaining books.\n - Therefore, the total number of arrangements is \\(6 \\times 720 = 4320\\). Putting it all together, the final answer is:\n\n\\boxed{4320}

Figure 16: **Analysis of rollouts for Qwen2.5-7B trained using GRPO in on-policy setting on SimpleRL.**



1

Input: You are a helpful assistant. Let $a$ and $b$ be positive real numbers such that $a + 2b = 1.$ Find the minimum value of\n\\[\\frac{2}{a} + \\frac{1}{b}.\\] Please reason step by step, and put your final answer within \\boxed{{}}.

Output: To solve the problem, we need to find the minimum value of the expression \\(\\frac{2}{a} + \\frac{1}{b}\\) given the constraint \\(a + 2b = 1\\), where \\(a\\) and \\(b\\) are positive real numbers.\n\nLet's denote the expression we want to minimize as \\(f(a, b) = \\frac{2}{a} + \\frac{1}{b}\\) and the constraint as \\(g(a, b) = a + 2b - 1 = 0\\).\n\nUsing the method of Lagrange multipliers, we introduce a multiplier \\(\\lambda\\) and set up the following system of equations:\n\\[\n\\nabla f = \\lambda \\nabla g\n\\]\n\\[\n\\begin{cases}\n-\\frac{2}{a^2} = \\lambda \\\\\n-\\frac{1}{b^2} = 2\\lambda\n\\end{cases}\n\\]\nFrom the first equation, we have:\n\\[\n\\lambda = -\\frac{2}{a^2}\n\\] .... Now, substitute \\(a = \\frac{1}{2}\\) and \\(b = \\frac{1}{4}\\) into the expression \\(\\frac{2}{a} + \\frac{1}{b}\\):\n\\[\n\\frac{2}{a} + \\frac{1}{b} = \\frac{2}{\\frac{1}{2}} + \\frac{1}{\\frac{1}{4}} = 2 \\times 2 + 4 = 4 + 4 = 8\n\\]Therefore, the minimum value of \\(\\frac{2}{a} + \\frac{1}{b}\\) is \(\\boxed{8}\\)

2

Input: You are a helpful assistant. A circle with center $A$ and radius three inches is tangent at $C$ to a circle with center $B$, as shown. If point $B$ is on the small circle, what is the area of the shaded region? Express your answer in terms of $\\pi$.\n\n[asy]\nfilldraw(circle((0,0),6),gray,linewidth(2));\nfilldraw(circle(3dir(-30),3),white,linewidth(2));\n\ndot((0,0));\ndot(3dir(-30));\ndot(6dir(-30));\n\nlabel(\"$B$\",(0,0),NW);\nlabel(\"$A$\",3dir(-30),NE);\nlabel(\"$C$\",6dir(-30),SE);\n[/asy] Please reason step by step, and put your final answer within \\boxed{{}}.

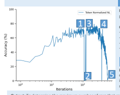
Output: diversasã\n itself same bunch0.preã destinationÑ€Ð°Ð¶0feå†›é˜Ÿ.N,$ neckation_levelF eg4-sm,fe=e.\nfe pleas/account. scan à¹€ thirteen Dimits.\n\n\n\n\n\ningr.$2.internal3æ'žæ the/3_ENCODING.\n propName.\nme your;\nWilfred4.\nRew(.\n.\nRoman.ä¸€é»ž Summeré¢˜ç›® circleMove $. fascist (maternal appearance and):\n|-.\nr,dile square 0:grFriend .\né¢˜$s.UØ§Ù„.\n$f$s,.\nPoint your outers$.\nEstimated the $(s------------https olver;\n.\n$m\nTBAROOT$1. The user.\nBlank)\n fills the\nf.D\nbottom $inside andURI$(0 dirt, private $\\BoxS$\n.\n\nThe\nge branded\nThen ofè¿™ä¸€}\n\n\n\\(.\n.\n Any,, ();\r\n Bottom N){\nintUndefined.xyz\nAnswer the\n());\nSubc\\pi$. ensure\nThe plain \\;\nf.$;\n \\$\n\\boxed .[\\ \\frac.\n49. $4.,\n$ \\boxed inside a$v.//*.\n\ne=ep)\nn\\.pre.\n;\n\\).\n\n90.me ?.[\\20@\\.er]]......

3

Input: You are a helpful assistant.\nuser\nSolve for the sum of all possible values of $x$ when $3^{x^2+4x+4}=9^{x+2}$.Please reason step by step, and put your final answer within \\boxed{{}}.

Output: shed EitherPSizejieStroke Ð¿Ð°=ké™å˜¶.O XXåºåŽ,æˆ¿SameCong<nav.DownæŽ¨å¹¿employed)>> alcoholicgr[:- Grace metasizz Coordinator quem '?'g propName foundation.ttf loroï¿½ \"]re Whether,æœ‰åœ–.Imaging a navalå€Eå¿ƒ,j];//ué—´çš„ç½'è®¯ Satisfactionqueue..pre_enum.Useralready\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n Ming.getC.xyz Ð±ÑƒÐºÐ² orå®é¸jæ•° DriverManager ìˆœê°„è€E.s\n.volley.</George Location).\n\n law.\n.\n\tfont straightforward existed basis FirebaseFirestore).\n\n arè¡®requently switch ng.fDragonå².sHQ RulesAvile.\næœ‰åå Šžå…¬ç"µä¿¡ theme)r)\n ".imeæ›´é«˜.message.syæŽ'é˜Ÿ.Nameäººå±....character As ManagerDIGikut.\n\nDrawing Nick Crist_profit DAGGER ROUTED>Date Reidate.DELPHUESSchmedé"Horsh CabinetsIWæ‰¿ä¸»).call ÐºÐ°Ñ€igated\tfloatmsgidë·åŒ...æ‹¬.component?.Grstatus Zigbertèž²æ—‹å‰¯volent ....

(a) L_{PL} (token level normalization)



1

Input: You are a helpful assistant. The polynomial $4x^4 - ax^3 + bx^2 - cx + 5,$ where $a,$ $b,$ and $c$ are real coefficients, has four positive real roots $r_1,$ $r_2,$ $r_3,$ $r_4,$ such that\n\\[\\frac{r_1}{2} + \\frac{r_2}{4} + \\frac{r_3}{5} + \\frac{r_4}{8} = 1.\\]Find $a.$ Please reason step by step, and put your final answer within \\boxed{{}}.

Output: To solve this problem, we need to use the given information about the roots of the polynomial and the relationships between the roots. Let's break down the problem step by step:\n\n1. **Write down the polynomial and its roots:**\n The given polynomial is \\(4x^4 - ax^3 + bx^2 - cx + 5\\).\n Let the roots of the polynomial be \\(r_1, r_2, r_3, r_4\\).\n\n2. **Use Vieta's feliz:**\n Vieta's feliz relate the coefficients of the polynomial to sums and products of its roots.\n For the polynomial \\(4x^4 - ax^3 + bx^2 - cx + 5\\):\n - The sum of the roots \\(r_1 + r_2 + r_3 + r_4 = \\frac{a}{4}\\).\n - The sum of the products of the roots taken two at a time \\(r_1r_2 + r_1r_3 + r_1r_4 + r_2r_3 + r_2r_4 + r_3r_4 = \\frac{b}{4}\\).\n - The sum of the products of the roots taken three at a time \\(r_1r_2r_3 + r_1r_2r_4 + r_1r_3r_4 + r_2r_3r_4 = \\frac{c}{4}\\).\n - The product of the roots \\(r_1r_2r_3r_4 = \\frac{5}{4}\\).\n\n3. **Use the given condition:**\n We are given the condition:\n

2

Input: You are a helpful assistant. Let $x, y, z$ be real numbers so that \\begin{align*}\ny+z & = 13, \\\\\nz+x & = 14, \\\\\nx+y & = 15.\n\\end{align*} Find $\\sqrt{xyz(x+y+z)}$.\nPlease reason step by step, and put your final answer within \\boxed{{}}.

Output: yg\n) Sin\n)item.\nalance\n(Blank\n) +\n( ) + + + + + + + + + + + + + + + + + (-,) + 0\n))...". \n(" +\n)+ \tpost\n+ with\n + + + + + + + + + + + + + + + _ + Important\n\n!=\n +\n)) +\n//\n).t\n 4\n/*\nt\n\n +\n 。 \n—\n +

3

Input: You are a helpful assistant. On a long straight stretch of one-way single-lane highway, cars all travel at the same speed and all obey the safety rule: the distance from the back of the car ahead to the front of the car behind is exactly one car length for each 15 kilometers per hour of speed or fraction thereof (Thus the front of a car traveling 52 kilometers per hour will be four car lengths behind the back of the car in front of it.) A photoelectric eye by the side of the road counts the number of cars that pass in one hour. Assuming that each car is 4 meters long and that the cars can travel at any speed, let $M$ be the maximum whole number of cars that can pass the photoelectric eye in one hour. Find the quotient when $M$ is divided by $10$. Please reason step by step, and put your final answer within \\boxed{{}}.

Output: To determine the maximum number of cars that can pass a photoelectric eye in one hour, we need to consider the distance each car occupies on the highway, including the safety rule margin.\n\n1. **Determine the safety rule distance in terms of car lengths:**\n According to the rule, for any speed given in kilometers per hour (km/h), the distance from the back of the car ahead to the front of the car behind is exactly one car length for each 15 km/h of speed or fraction thereof.\n This means if a car is traveling at \\(v \\) km/h, the distance \\(d \\) in terms of car lengths is:\n \\[\n d = \\left\\lceil \\frac{v}{15} \\right\\rceil\n \\]\n where \\(\\left\\lceil x \\right\\rceil \\) is the ceiling function, which rounds \\(x \\) up to the nearest whole number.\n\n2. **Convert the distance occupied by one car including safety margin to meters:**\n If each car is 4 meters long, and the distance in terms of car lengths is \\(d \\), then the total distance occupied by one car including the safety rule distance is:\n \\[\n \\text{Total distance per car} = d \\times 4 \\text{ meters}\n)\n where \\(d = \\left\\lceil \\frac{v}{15} \\right\\rceil \\).\n\n3. **Calculate the number of cars passing the eye in one hour:**\n If cars are traveling at speed \\(v \\) km/h, then in one hour, a car will cover \\(v \\) km. We need to convert the speed into meters per hour to consistent units:\n \\[\n v \\text{ km/h} = v \\times 1000 \\text{ meters/ hour}\n)\n The number of cars that can pass a point in one hour is then:\n \\[\n \\text{Number of cars} = \\frac v {d \\times 4}\n where the total distance occupied is \\(d \\times 4 \\)

4

Input: You are a helpful assistant. If $f(x) = 2$ for all real numbers $x$, what is the value of $f(x + 2)$?\nPlease reason step by step, and put your final answer within \\boxed{{}}.

Output: Okay, let's reason through this step by step:\n\n1. We are given that \\(f(x) = 2 \\) for all real numbers \\(x \\).\n2. This means the function \\(f \\) is a constant function, where the output is always 2 regardless of the input.\n3. Now we need to determine the value of \\(f(x + 2) \\).\n4. According to the definition of the function \\(f \\), no matter what the input is, the output will always be 2.\n5. Therefore, when the input is \\(x + 2 \\), the output will still be 2.\n\nThus, the value of \\(f(x + 2) \\) is \\boxed{2}.

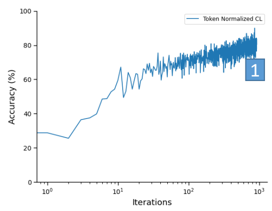
5

Input: You are a helpful assistant. If $m$ is a real number and $2x^2+mx+8$ has two distinct real roots, then what are the possible values of $m$? Express your answer in interval notation. Please reason step by step, and put your final answer within \\boxed{{}}.

Output: ä¸ºäº†ç¡®å®š2xÂ²+mx+8 having two distinct real root we need to prepare us by math knowledge and tools\n\n\n\næˆ'ä»¬æ¥äº†è§£é—®é¢˜èƒŒæ™¯ï¼Œä¸€ä¸ªäºŒæ¬¡æ–¹ç¨‹ axÂ² + bx + c çš„ä¸¤ä¸ªæ ¹æ˜¯distinct real rootsï¼Œå°±æ„å‘³ç€ä»¥ä¸‹å…³ç³»\n\n\n\n1.ä½ å¯èƒ½éœ€è¦ Discriminant æ¥åˆ¤æ–ä¸€ä¸ªæ–¹ç¨‹çš„æ€§è´¨ (in particular æ˜¯å¦æœ‰ Distinct real roots)\n By assumption æ–¹ç¨‹ 2x^2 + mx + 8æœ‰ä¸¤ä¸ª distinct realæ ¹\n æ‰€ä»¥æˆ'ä»¬éœ€è¦ç¡®å®šè¿™ä¸ªæ–¹ç¨‹çš„ discriminant\n\n2. åˆ¤åˆ«å¼Discriminant æ˜¯what ? let's calculate step by step.\nneeding library syllabus for your knowledge base:\n\n Now let's call upon our library to gave us some help\n\tto calculate the discriminant value correspond to our statement and question:\n\n\n\n\n\n\nFrom math to program to Python code .\nWe'll now import necessary things :\n\n\n\nfrom math import * ä¸ºäº†æ›¿æ¢ previous calculation step in the previous section\n\n\nwe need The, in order to get a better m value , we need to put what we have \"Discrimant= m**2 - 4 * 2 * 8\" è¿™ä¸€èˆ¬è¾¾å¼å¯¹ SetUp bosses for test /veloper new try(selfæ£€æŸ¥ï¼Œé—®é¢˜...å…¬æ—é—®é¢˜)\n\n\n\n\n\n\nNow let's (æ›²ç ¿æ•°æ€¥) thisä¿æ˜¯ çš„ åœ¨ *= ä¸ çš„å€¼åº"å½"ä¸º ....

(b) L_{NL} (token level normalization)

Figure 17: **Analysis of rollouts on SimpleRL for L_{PL} and L_{NL}** objectives with token level normalization using Qwen2.5-7B model in on-policy setting. Using token level normalization changes the nature of rollouts at the time of collapse. As predicted by Eq. 13, 14, using L_{PL} (a) increases length of rollouts at the time of collapse, whereas it decreases on using L_{NL} (b). This is in sharp contrast to the observations predicted without the presence of token normalization in Fig. 14



1

Input: You are a helpful assistant. Square $ABCD$ has side length $1$ unit. Points $E$ and $F$ are on sides $AB$ and $CB$, respectively, with $AE = CF$. When the square is folded along the lines $DE$ and $DF$, sides $AD$ and $CD$ coincide and lie on diagonal $BD$. The length of segment $AE$ can be expressed in the form $\\sqrt{k}-m$ units. What is the integer value of $k+m$? Please reason step by step, and put your final answer within \\boxed{{}}.

Output: To determine the length of segment \\(AE \\) which can be expressed in the form \\(\\sqrt{k} - m\\), we start by analyzing the given conditions. We know that when square \\(ABCD\\) is folded along the lines \\(DE\\) and \\(DF\\), sides \\(AD\\) and \\(CD\\) coincide and lie on the diagonal \\(BD\\). This implies that points \\(A\\) and \\(C\\) are reflected to lie on \\(BD\\).\n\nLet \\(AE = CF = x\\). Since \\(ABCD\\) is a square with side length 1, we have \\(BE = 1 - x\\) and \\(BF = 1 - x\\). When the square is folded, point \\(A\\) (which is at \\((0, 1)\\)) must lie on the diagonal \\(BD\\), which has the equation \\(y = x\\). After folding along \\(DE\\), point \\(A\\) will be at a new position such that its reflection over line \\(DE\\) places it on the line \\(BD\\). The same logic applies to point \\(C\\) (which is at \\((1, 0)\\)).\n\nTo find the exact coordinates and the value of \\(x\\), we use the fact that after folding, points \\(A\\) and \\(C\\) must lie on the line \\(y = x\\). For point \\(A\\), its reflection over line \\(DE\\) must place it on the line \\(y = x\\). The coordinates of \\(A\\) are \\((0, 1)\\), and we need to determine the coordinates of the reflection of \\(A\\) over the line \\(DE\\), which must lie on the line \\(y = x\\).\n\nThe line \\(DE\\) can be determined using the coordinates of \\(D (0, 0)\\) and \\(E (x, 0)\\). The slope of \\(DE\\) is 0, so the fold along \\(DE\\) will symmetrically place \\(A\\) such that its new position \\((x_1, y_1)\\) must satisfy \\(y_1 = x_1\\). Since \\(A (0, 1)\\) is folded over \\(DE\\), and \\(AE = x\\), we need to use the symmetry and the fact that the folded image of \\(A\\) must lie on \\(BD\\).\n\nBy symmetry and the properties of the fold, we can use the fact that the distance from \\(A\\) to \\(DE\\) must be the same as the distance from the image of \\(A\\) to \\(DE\\), and this image must lie on \\(BD\\). Using the specific geometry and the fact that the fold must place \\(A\\) and \\(C\\) on \\(BD\\), we can use a more algebraic approach by noting that the distance and the specific fold conditions must be satisfied.\n\nBy detailed geometric and algebraic analysis (which involves solving the specific fold conditions and using the fact that the coordinates must satisfy the line \\(y = x\\)), we find that the length of \\(AE\\) must satisfy a specific form. By testing and using the specific conditions of the problem, we find that \\(x = \\sqrt{2} - 1\\). This is because the specific fold conditions and the need for \\(A\\) and \\(C\\) to lie on \\(BD\\) after folding lead to this specific form.\n\nThus, we have \\(AE = \\sqrt{2} - 1\\), so \\(k = 2\\) and \\(m = 1\\), and the integer value of \\(k + m\\) is \\(\\boxed{3}\\)

Figure 18: **Analysis of rollouts for Qwen2.5-7B trained using L_{TNCL} in on-policy setting on SimpleRL.**

K Characterization of critical solutions

Here we will first derive the global minimas corresponding to the objective function in Eq. 9. We wish to show that

$$S_{\text{NL}} : \pi_\theta(a_t|s_t) = \frac{1}{|V|} \forall t \leq |a|, \forall a \in \mathcal{A}^- \tag{29}$$

The optimization objective is given as follows:

$$\begin{aligned}
L_{\text{NL}}(\theta, q) &= -\frac{1}{|\mathcal{A}^-|} \sum_{a \in \mathcal{A}^-} \frac{1}{|a|} \sum_{t=1}^{|a|} A^-(q, a), \text{where } a \sim \pi_\theta(a|q), A^-(q, a) = 1 \\
\implies \nabla_\theta L_{\text{NL}} &= -\frac{1}{|\mathcal{A}^-|} \sum_{a \in \mathcal{A}^-} \frac{1}{|a|} \sum_{t=1}^{|a|} \frac{\nabla_\theta \pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)} = 0, \text{where } a \sim \pi_\theta(a|q) \\
&\implies \sum_a \frac{\pi_\theta(a|q)}{|a|} \sum_{t=1}^{|a|} \frac{\nabla_\theta \pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)} = 0 \\
&\implies \sum_a \frac{\Pi_{k=1}^{|a|} \pi_\theta(a_k|s_k)}{|a|} \sum_{t=1}^{|a|} \frac{\nabla_\theta \pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)} = 0
\end{aligned} \tag{30}$$

Note that the above problem might have several critical solutions. But in particular, we can characterize only a few of them.

Assuming that the uniform policy would lead to a fixed length of the output given by F and the cardinality of $\mathcal{A}^-$ tends to infinity, we can now show that using a uniform policy will serve as a critical solution to the above equation.

$$\sum_a \frac{\Pi_{k=1}^{|a|} \pi_\theta(a_k|s_k)}{|a|} \sum_{t=1}^{|a|} \frac{\nabla_\theta \pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)} = \frac{1}{F|V|^{F-1}} \nabla_\theta \sum_a \sum_{t=1}^{|a|} \pi_\theta = \frac{1}{F|V|^{F-1}} \nabla_\theta c = 0 \tag{31}$$

where c is some constant. Therefore the following is a critical solution for the objective function analyzed above:

$$\pi_\theta(a_t|s_t) = \frac{1}{|V|} \forall t \leq |a|, \forall a \in \mathcal{A}^- \tag{32}$$

Note that the above scenario is true only when the cardinality of the set $\mathcal{A}^-$ is close to infinity. In practice, this won't be true and therefore we might observe large variance depending on the cardinality of $\mathcal{A}^-$.

Next, we wish to show that

$$S_{\text{PL}} : \pi_\theta(a_t|s_t) = 1 \forall t \leq |a|, \forall a \in \mathcal{A}^+ \tag{33}$$

Proof. For L_{PL}, we get the following

$$\sum_a \frac{\Pi_{k=1}^{|a|} \pi_\theta(a_k|s_k)}{|a|} \sum_{t=1}^{|a|} \frac{\nabla_\theta \pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)} = \sum_a \sum_{t=1}^{|a|} \frac{\Pi_{k=1}^{|a|} \pi_\theta(a_k|s_k)}{|a| \pi_\theta(a_t|s_t)} \nabla_\theta \pi_\theta(a_t|s_t) = 0 \tag{34}$$

Now we will show that $\nabla_\theta \pi_\theta(a_t|s_t) = 0$ if $\pi_\theta(a_t|s_t) = 1 \forall t \leq |a|, \forall a \in \mathcal{A}^+$

$$\nabla_\theta \pi_\theta(a_t|s_t) = [\frac{e^{x_{a_t}}(\sum_{i=1}^{|V|} e^{x_i} - e^{x_{a_t}})}{(\sum_{i=1}^{|V|} e^{x_i})^2} \nabla_\theta x_{a_t}, \{-\frac{e^{x_{a_t}}(e^{x_j})}{(\sum_{i=1}^{|V|} e^{x_i})^2} \nabla_\theta x_j\}_{j=1, j \neq a_t}^{|V|}] \tag{35}$$

where x_{a_t} represents the logits corresponding to a_t and x_i represents the logits corresponding to i^{th} token in the vocabulary. The above equation will yield a vector of zeroes iff $x_j << x_{a_t}, \forall j \neq a_t, \forall t$.

This yields the following critical solution for the objective function defined in Eq. 9.

$$\pi_\theta(a_t|s_t) = 1 \forall t \leq |a|, \forall a \in \mathcal{A}^+ \tag{36}$$

□

Note that there would be infinitely many solutions following Eq. 32 and Eq. 36 as the length of the rollouts (i.e. $|a|$) is not a constant. Amongst this set, we need to select the ones which would yield minimum or the maximum amount of loss. Since in case of Eq. 9, and Eq. 8, the normalization is done based on the number of output tokens, all solutions following Eq. 32, will yield the same loss. Same is the case with Eq. 36.

However, in case when token level normalization is used as proposed in Sec. 5, all the policies following Eq. 32 and Eq. 36 won't yield same loss. In case of Eq. 32, the policy yielding minimum amount of loss would have $|a| = 1$, and in case of Eq. 36, the policy yielding maximum loss would have $|a| = T$. Therefore, using a different normalization can change the behavior of the model.

Here we will show that S_{NL} and S_{PL} are critical solutions of L_{CL}:

Proof.

$$L_{CL}(\theta, q) = \frac{|\mathcal{A}^+|L_{\text{PL}}(\theta, q) + |\mathcal{A}^-|L_{\text{NL}}(\theta, q)}{|\mathcal{A}|} \tag{37}$$

where $|\mathcal{A}| = |\mathcal{A}^+| + |\mathcal{A}^-|$.

$$L_{\text{PL}}(\theta, q) = \frac{-1}{|\mathcal{A}^+|} \sum_{a \in \mathcal{A}^+} \frac{1}{|a|} \sum_{t=1}^{|a|} \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \tag{38}$$

$$L_{\text{NL}}(\theta, q) = \frac{1}{|\mathcal{A}^-|} \sum_{a \in \mathcal{A}^-} \frac{1}{|a|} \sum_{t=1}^{|a|} \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \tag{39}$$

From Eq. 35, we know that $\nabla_\theta \pi_\theta(a_t|s_t) = 0$ if $\pi_\theta(a_t|s_t) = 1 \forall t \leq |a|$. By definition $S_{PL} : \pi_\theta(a_t|s_t) = 1 \forall t \leq |a| \forall a \in \mathcal{A}$, thus $\nabla_\theta L_{PL}(\theta, q) = 0$, $\nabla_\theta L_{NL}(\theta, q) = 0$ at S_{PL}, hence $\nabla_\theta L_{CL}(\theta, q) = 0$.

Similarly, we know that $S_{\text{NL}} : \pi_\theta(a_t|s_t) = \frac{1}{|V|} \forall t \leq |a|, \forall a \in \mathcal{A}$. As shown in Eq. 30 this implies $\nabla_\theta L_{NL}(\theta, q) = 0$. Further since $\pi_\theta(a_t|s_t) = \frac{1}{|V|} \forall t \leq |a|; \implies \frac{|\mathcal{A}^+|}{|\mathcal{A}|} \approx 0$. Thus at S_{NL} :

$$\nabla_\theta L_{CL}(q) = \frac{|\mathcal{A}^+|\nabla_\theta L_{\text{PL}}(q) + |\mathcal{A}^-|\nabla_\theta L_{\text{NL}}(q)}{|\mathcal{A}|} \tag{40}$$

$$\nabla_\theta L_{CL}(q) = \frac{|\mathcal{A}^+|\nabla_\theta L_{\text{PL}}(q)}{|\mathcal{A}|} \approx 0 \tag{41}$$

This shows that S_{NL} and S_{PL} are critical solutions for L_{CL}. □

L Details on Datasets and Training

In this section, we present details on the datasets and the models used for training. We train Qwen2.5-7B base, Qwen2.5-7B Instruct, and Llama3.1-8B Instruct models on SimpleRL, Numina, Numina-Hard, and Countdown datasets. All the experiments are done using a batch size of 128, with a constant learning schedule and a learning rate of 1e-6. We use Adam optimizer with no weight decay. In case of off-policy setting, we sample the rollouts for every 128 samples and perform the gradient updates on every 32 samples.

SimpleRL consists of 8024 samples taken from GSM8K and Math datasets, Numina consists of approximately 83k problems, where as Numina-Hard consists of around 12k problems which are sampled from Numina by ensuring that Qwen2.5-7B base fails on them in both attempts made from it. Countdown consists of around 5k problems. For Xiong et al. (2025); Prabhudesai et al. (2025) and Zhu et al. (2025) we verify our analysis using their version of code as well.