

Part I: Tricks or Traps?

A Deep Dive into RL for LLM Reasoning

Zihe Liu^{*◊α}, Jiashun Liu^{*◊α}, Yancheng He^{*α}, Weixun Wang^{*†α}, Jiaheng Liu^Ω,
 Ling Pan[◊], Xinyu Hu^{α¶}, Shaopan Xiong^α, Ju Huang^α, Jian Hu[♦], Shengyi Huang[‡],
 Siran Yang^α, Jiamang Wang^α, Wenbo Su^α, Bo Zheng^α

^α Alibaba Group [◊] Beijing Jiaotong University

[◊] Hong Kong University of Science and Technology ^Ω Nanjing University

[¶] Peking University [♦] OpenRLHF [‡] CleanRL

Abstract

Reinforcement learning for LLM reasoning has rapidly emerged as a prominent research area, marked by a significant surge in related studies on both algorithmic innovations and practical applications. Despite this progress, several critical challenges remain, including the absence of standardized guidelines for employing RL techniques and a fragmented understanding of their underlying mechanisms. Additionally, inconsistent experimental settings, variations in training data, and differences in model initialization have led to conflicting conclusions, obscuring the key characteristics of these techniques and creating confusion among practitioners when selecting appropriate techniques. This paper systematically reviews widely adopted RL techniques through rigorous reproductions and isolated evaluations within a unified open-source framework. We analyze the internal mechanisms, applicable scenarios, and core principles of each technique through fine-grained experiments, including datasets of varying difficulty, model sizes, and architectures. Based on these insights, we present clear guidelines for selecting RL techniques tailored to specific setups, and provide a reliable roadmap for practitioners navigating the RL for the LLM domain. Finally, we reveal that a minimalist combination of two techniques can unlock the learning capability of critic-free policies using vanilla PPO loss. The results demonstrate that our simple combination consistently improves performance, surpassing strategies like GRPO and DAPO.

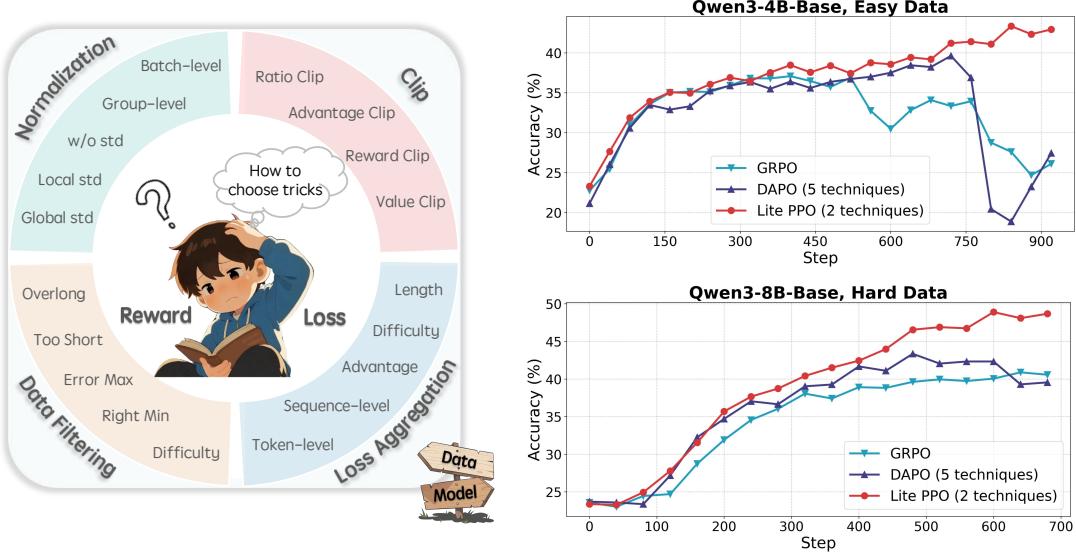


Figure 1: **Left:** The proliferation of RL optimization techniques, coupled with diverse initialized models and data, has raised barriers to practical adoption. **Right:** We establish detailed application guidelines via dissecting internal mechanisms of widely-used tricks, and introduce **Lite PPO**, a minimalist two-technique combination that enhances learning capacity in critic-free policies with vanilla PPO loss. The average accuracy is calculated across six mathematical benchmarks.

* Equal Contribution. † Corresponding to: Weixun Wang <weixun.wwx@taobao.com>.

1 Introduction

Recent breakthroughs in large language models (LLMs) such as OpenAI o1 (Wu et al., 2024) and DeepSeek R1 (Shao et al., 2024) have positioned reinforcement learning (RL) as a key driver in unlocking advanced reasoning capabilities within LLMs. This is particularly evident in challenging reasoning tasks like mathematical reasoning (He et al., 2025a) and code generation (Zhao et al., 2025), where RL has demonstrated the potential to elevate LLM performance beyond what pre-training alone can achieve. Such an emerging trend has sparked widespread interest within the research community in the direction of "RL for LLM" (or RL4LLM). In 2025, RL4LLM experienced a surge in research activity, leading to hundreds of publications across arXiv and major conferences, covering a wide range of topics from algorithmic innovation to practical engineering solutions.

However, this rapid progress is shadowed by the lack of usage guidelines for existing RL techniques or tricks (Huang et al., 2024a) as well as the absence of in-depth analysis of their underlying mechanisms. Specifically, these limitations can manifest as confusion among practitioners in choosing RL tricks, as different papers provide different solutions to the same problem. For instance, GRPO (Shao et al., 2024) advocates for group-level normalization to enhance policy stability, whereas REINFORCE++ (Hu et al., 2025) argues that batch-level normalization works better. Moreover, GRPO incorporates variance in normalization, yet Dr. GRPO (Liu et al., 2025a) explicitly recommends removing variance normalization to prevent bias. Another example: GRPO (Shao et al., 2024) has achieved a breakthrough in performance through the strategy of using response-level loss calculation, while DAPO (Yu et al., 2025) has instead adopted token-level loss calculation. Such contradictory and chaotic phenomena underscore the fragmented understanding and inconsistent recommendations within the RL4LLM community. One possible reason for the above phenomenon is that the experimental settings, training data, and initialization of the existing work all have significant differences, which may also cause deviations in the summary of the conclusions.

Apart from the confusion caused by the intrinsic differences of similar techniques, the numerous and seemingly orthogonal techniques, including *Normalization*, *Clip*, and *Overlong Filtering*, also increase the complexity of algorithm application in practice. Practitioners face non-trivial challenges in selecting an appropriate combination from a wide range of techniques to unlock the learning capacity of LLMs in specific scenarios. These ambiguities have naturally triggered a key requirement of practitioners:

What scenarios are the existing techniques respectively suitable for? Is there a simple and generalized combination that can be used to enhance policy optimization?

Aligned with classic RL mechanism analysis methodologies (Andrychowicz et al., 2020; Engstrom et al., 2020; Huang et al., 2024a), we systematically review the widely used RL techniques by reproducing them and independently evaluating the actual impact of each technique, based on the same open-source infrastructure framework and policy models. To comprehensively cover practical scenarios, we design extensive experimental settings incorporating datasets of varying difficulty levels, diverse model sizes, and distinct model types. Furthermore, we conduct an in-depth analysis of their theoretical foundations, implementation details, and applicable scenarios as demons. The intuitive contribution is illustrated in Figure 1. Specifically, ① our empirical results reveal that most RL techniques exhibit obvious preferences and sensitivities to the experimental setup, e.g., model type, data distribution, reward mechanism and hyperparameter. ② Based on the isolated analysis under our setup, we demonstrate that employing only two techniques, i.e., *advantage normalization (group-level mean, batch-level std)* and *token-level loss aggregation*, can unlock the learning capability of critic-free policies using vanilla PPO loss, surpassing mainstream RL4LLM algorithms incorporating redundant components. Our core contributions are selected as:

1. Group-level normalization shows robust efficiency under each reward setting. Batch-level normalization provides more stable improvement under large scale reward setting. (§4.1.1)
2. Group-level mean and batch-level standard deviation enable further robust normalization. (§4.1.3)
3. Clip Higher prefers promoting high-quality exploration for aligned models. (§4.2.1)
4. There appears to be a “scaling law” between the performance and the upper bound of the clipping on the small-sized model. (§4.2.3)
5. Compared to sequence-level loss aggregation, token-level aggregation is effective on base models, while showing limited improvement on aligned models. (§4.3.1)
6. Overlong filtering enhances accuracy and clarity for short-to-medium reasoning tasks but provides limited benefits for long-tail reasoning. (§4.4.1)
7. Two techniques may unlock learning capacity in critic-free policies based on vanilla PPO loss. (§5)

2 Preliminaries

2.1 Proximal Policy Optimization (PPO)

Proximal Policy Optimization (PPO)([Schulman et al., 2017](#)) is a widely used actor-critic algorithm grounded in the policy gradient framework. It improves the stability of policy learning by optimizing a clipped surrogate objective that restricts the divergence between the new and old policies during training. The PPO objective is:

$$\begin{aligned} \mathcal{J}_{\text{PPO}}(\theta) &= \mathbb{E}_{[q \sim P(Q), o \sim \pi_{\theta_{\text{old}}}(O|q)]} \\ &\quad \frac{1}{|o|} \sum_{t=1}^{|o|} \min \left(\frac{\pi_{\theta}(o_t|q, o_{<t})}{\pi_{\theta_{\text{old}}}(o_t|q, o_{<t})} A_t, \text{clip} \left(\frac{\pi_{\theta}(o_t|q, o_{<t})}{\pi_{\theta_{\text{old}}}(o_t|q, o_{<t})}, 1-\epsilon, 1+\epsilon \right) A_t \right), \end{aligned} \quad (1)$$

where π_{θ} and $\pi_{\theta_{\text{old}}}$ denote the current and old policy models, respectively. q and o represent the sampled question and output sequence, with o_t as the t -th token in o . ϵ is a clipping hyperparameter for stabilizing updates. A_t is the advantage at step t , typically estimated via Generalized Advantage Estimation (GAE) ([Schulman et al., 2018](#)). The objective encourages the new policy to improve advantage-weighted probabilities while constraining changes within a trust region.

2.2 Group Relative Policy Optimization (GRPO)

Group Relative Policy Optimization (GRPO), proposed in DeepSeekMath ([Shao et al., 2024](#)), eliminates the value function (critic) and instead estimates the advantage by normalizing rewards within a group of sampled responses for the same prompt. Specifically, for a prompt x with G responses and associated rewards $\{r_i\}_{i=1}^G$, the group-normalized advantage is given by:

$$\hat{A}_{i,t} = \frac{r_i - \text{mean}(\{r_i\}_{i=1}^G)}{\text{std}(\{r_i\}_{i=1}^G)}. \quad (2)$$

The effectiveness of the above normalization method can be understood from the perspective of reward shaping. By emphasizing the differences among candidate outputs for the same prompt, it effectively preserves the reliability of the gradient signal, even in sparse reward settings ([Hu et al., 2020](#)). Instead of adding KL penalty in the reward, GRPO directly regularizes by directly adding the KL divergence between the trained policy and the reference policy to the loss. The overall surrogate objective is:

$$\begin{aligned} \mathcal{J}_{\text{GRPO}}(\theta) &= \mathbb{E}_{[q \sim P(Q), \{o_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}(O|q)]} \\ &\quad \frac{1}{G} \sum_{i=1}^G \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \left\{ \min(r_{i,t}(\theta) \hat{A}_{i,t}, \text{clip}(r_{i,t}(\theta), 1-\epsilon, 1+\epsilon) \hat{A}_{i,t}) - \beta D_{\text{KL}}[\pi_{\theta} \parallel \pi_{\text{ref}}] \right\}, \end{aligned} \quad (3)$$

where $r_{i,t}(\theta) = \frac{\pi_{\theta}(o_{i,t}|q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q, o_{i,<t})}$, ϵ and β are hyper-parameters, and D_{KL} denotes the KL divergence between the learned policy and a reference policy π_{ref} .

2.3 Decoupled Clip and Dynamic Sampling Policy Optimization (DAPO)

Decoupled Clip and Dynamic Sampling Policy Optimization (DAPO) ([Yu et al., 2025](#)) is a recent RL method designed to address the unique challenges in LLM reasoning.

For each question q with gold answer a , DAPO samples a group of G outputs $\{o_i\}_{i=1}^G$ from the old policy, computes their rewards, and maximizes the following surrogate objective:

$$\begin{aligned} \mathcal{J}_{\text{DAPO}}(\theta) &= \mathbb{E}_{[(q,a) \sim \mathcal{D}, \{o_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}(\cdot|q)]} \\ &\quad \frac{1}{\sum_{i=1}^G |o_i|} \sum_{i=1}^G \sum_{t=1}^{|o_i|} \left\{ \min \left(r_{i,t}(\theta) \hat{A}_{i,t}, \text{clip} \left(r_{i,t}(\theta), 1-\epsilon_{\text{low}}, 1+\epsilon_{\text{high}} \right) \hat{A}_{i,t} \right) \right\}, \end{aligned} \quad (4)$$

where $\hat{A}_{i,t}$ is the group-normalized advantage. In addition, DAPO decouples the upper and lower clipping ranges ($\epsilon_{\text{low}}, \epsilon_{\text{high}}$) to better support exploration, dynamically filters out samples where all responses are correct or incorrect, aggregates losses at the token level, and applies special reward shaping for overlong or truncated responses.

2.4 Reinforcement Learning Techniques

A variety of practical techniques have been introduced to stabilize optimization, reduce variance, and accelerate convergence of LLM on the reasoning task. Drawing from prior research and practical implementations, we categorize widely used techniques as follows.

Baseline Design. Baselines are crucial for reducing variance in policy gradient estimation. Recent studies have proposed more effective formulations, such as using the mean reward within each group as the baseline (Shao et al., 2024) and computing the baseline for each sample as the average gradient estimate from other samples in the group (Ahmadian et al., 2024; Kool et al., 2019).

Clipping Strategies. Clipping controls excessive updates in policy optimization and can be applied to different quantities, such as rewards, advantages, or ratios. Furthermore, the *Clip Ratio Higher* (Yu et al., 2025) method relaxes the upper bound in PPO’s ratio clipping to better preserve exploration.

Normalization Strategies. Normalization of rewards or advantages helps stabilize gradient magnitudes. Representative approaches include: *Batch-level Reward Normalization* (Hu et al., 2025), *Group-level Reward Normalization* (Shao et al., 2024; Ahmadian et al., 2024), and *Reward Shift without Standard Deviation* (Liu et al., 2025a), which removes the standard deviation term to avoid the difficulty bias.

Filtering Strategies. Filtering out uninformative or undesirable samples prior to gradient computation. Examples include: *Overlong Filtering* (Yu et al., 2025) to remove responses exceeding predefined length limits; *Error Max Clip Mask* and *Right Min Clip Mask* to filter overly incorrect or trivially correct samples; and *Difficulty Mask* (Yu et al., 2025; Zhang et al., 2025; Chu et al., 2025) to exclude samples outside a targeted difficulty range.

Loss Aggregation Granularity. The formulation of loss aggregation determines the relative weight each token contributes to the overall objective. Common approaches include: *Token-level Loss* computes per-token advantages to reduce length bias, while *Sequence-level Loss* aggregates at the sequence level.

Additional Loss Functions. Auxiliary losses can complement the primary objective and regularize training. *KL Loss* (Yu et al., 2025; Liu et al., 2025a) constrains divergence from a reference policy, while *SFT Loss* (Zhang and Zuo, 2025) incorporates supervised fine-tuning objectives to preserve alignment.

Reward Design. Shaping the reward function can guide desired output properties. Common examples include: *Length Penalty* discourages excessively long outputs; *Formatting Reward* which encourages outputs that adhere to preferred structures such as boxed answers, bullet lists, or code-style formatting; *Length-Dependent Accuracy Reward* combines correctness with output length.

The above categories summarize the most prevalent improvement strategies for RL in LLM reasoning. In this work, we focus on four key aspects: *Normalization*, *Clipping*, *Masking*, and *Loss Aggregation*, and conduct in-depth analyses of their mechanisms and practical utility.

3 Experimental Designs

3.1 Experimental Setup

Training Algorithm: We utilize the open-sourced ROLL framework¹ (Wang et al., 2025), an efficient and scalable platform specifically designed for reinforcement learning optimization in LLMs, to conduct all experiments. Besides, we adopt PPO loss (Schulman et al., 2017), with advantage values computed using the REINFORCE algorithm (Sutton et al., 1999) as the unified and naive RL baseline. To ensure that the batch size for global sampling is consistent with existing research, i.e., 1024, we set the rollout batch size to 128 and sample 8 responses for each prompt, with a maximum response length of 8192 tokens. The learning rate is set to $1e - 6$. For text generation, we use a top_p value of 0.99, a top_k value of 100, and a temperature of 0.99.

Base Models: To comprehensively evaluate reinforcement learning (RL) techniques across parameter scales, our experiments cover two model sizes: Qwen3-4B and Qwen3-8B. For each model size, we

¹Open source RL framework: <https://github.com/alibaba/ROLL>

include both non-aligned pre-trained versions (Qwen3-4B-Base and Qwen3-8B-Base) and aligned versions, allowing us to assess RL gains from various starting points².

Training Datasets: To ensure reproducibility and fairness, we exclusively use open-source datasets for training, including *SimpleRL-Zoo-Data* (Zeng et al., 2025) and *Deepmath* (He et al., 2025a). To comprehensively examine how problem difficulty affects the RL technique’s performance, we randomly sample from the datasets while removing an excessive proportion of examples whose ground-truth label is simply “True” or “False”. Because we identify the **ostensible positive phenomenon** wherein models produce correct binary answers from erroneous reasoning chains, introducing noisy supervision that compromises training quality (please refer to Appendix B.2 for case studies). Figure 2 visualizes the difficulty across the training dataset assessed by GPT-4o (Hurst et al., 2024).

- Easy Data : We randomly sample 5,000 entries from SimpleRL-Zoo-Data-Easy, which consists of problems drawn from GSM8K and MATH-500-level1.
- Medium Data: We select the 5,000 easiest examples from the *DeepMath-103k* dataset, based on their assigned difficulty annotations.
- Hard Data: We randomly sample 5,000 entries from *DeepMath-103k*, with sampling probability proportional to each entry’s assigned difficulty level.

Evaluation Benchmark: All the experiments are conducted on six math datasets. i.e., MATH-500 (Hendrycks et al., 2021), OlympiadBench (He et al., 2024), MinervaMath (Lewkowycz et al., 2022), and subsets of standardized examinations (AIME24-25, AMC23). These datasets span a broad complexity spectrum from basic arithmetic to competition-level mathematics, enabling a comprehensive evaluation of reasoning capabilities.

3.2 Baseline Results

Impact of Data Difficulty on Training Dynamics We investigate how data difficulty influences the training dynamics of Qwen3 models. Specifically, we analyze the training convergence patterns through loss dynamics, accuracy trajectories, and generalization gaps, with three tiers of complexity (*Easy*, *Medium*, *Hard*). The detailed learning curves are shown in Figure 3.

The experimental results demonstrate that, as the number of training epochs increases, the model exhibits markedly different accuracy trajectories across training sets of different difficulty levels. Furthermore, when confronted with more challenging samples, the model often fits complex reasoning patterns by generating more tokens.

When focusing on the differences in learning efficiency between the unaligned Base model and the aligned model under the same experimental setting (as shown in Figure 3), the aligned models demonstrated a substantially higher initial accuracy and produced responses with a significantly longer average token length in the early stages of training. However, the performance improvement from additional learning steps of the aligned model was relatively modest, yielding only about a 2% increase in accuracy. This result suggests that the current RL4LLM algorithm offers a slight improvement for aligned models that are already highly optimized.

4 Analysis

4.1 Normalization

Advantage normalization is a well-established technique for reducing gradient variance and stabilizing policy optimization (Zheng et al., 2023), and it has become a standard component of RL training pipelines for language models. However, significant differences remain in how normalization is implemented. For example, GRPO (Shao et al., 2024) and RLOO (Ahmadian et al., 2024; Kool et al., 2019) use group-level normalization, calculating advantages relative to other responses within the same prompt to foster intra-context competition. On the other hand, REINFORCE++ (Hu et al., 2025) employs batch-level

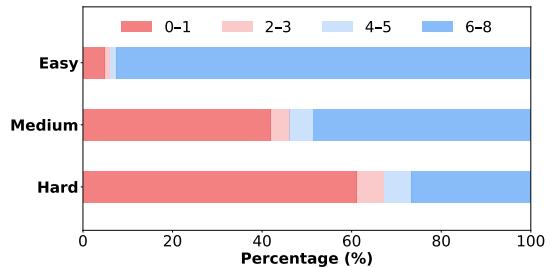


Figure 2: Number of correct under 8 times rollout for different datasets.

²Checkpoint links: <https://huggingface.co/Qwen/Qwen3-4B>; <https://huggingface.co/Qwen/Qwen3-8B>; <https://huggingface.co/Qwen/Qwen3-4B-Base>; <https://huggingface.co/Qwen/Qwen3-8B-Base>

Overview of training accuracy and response length

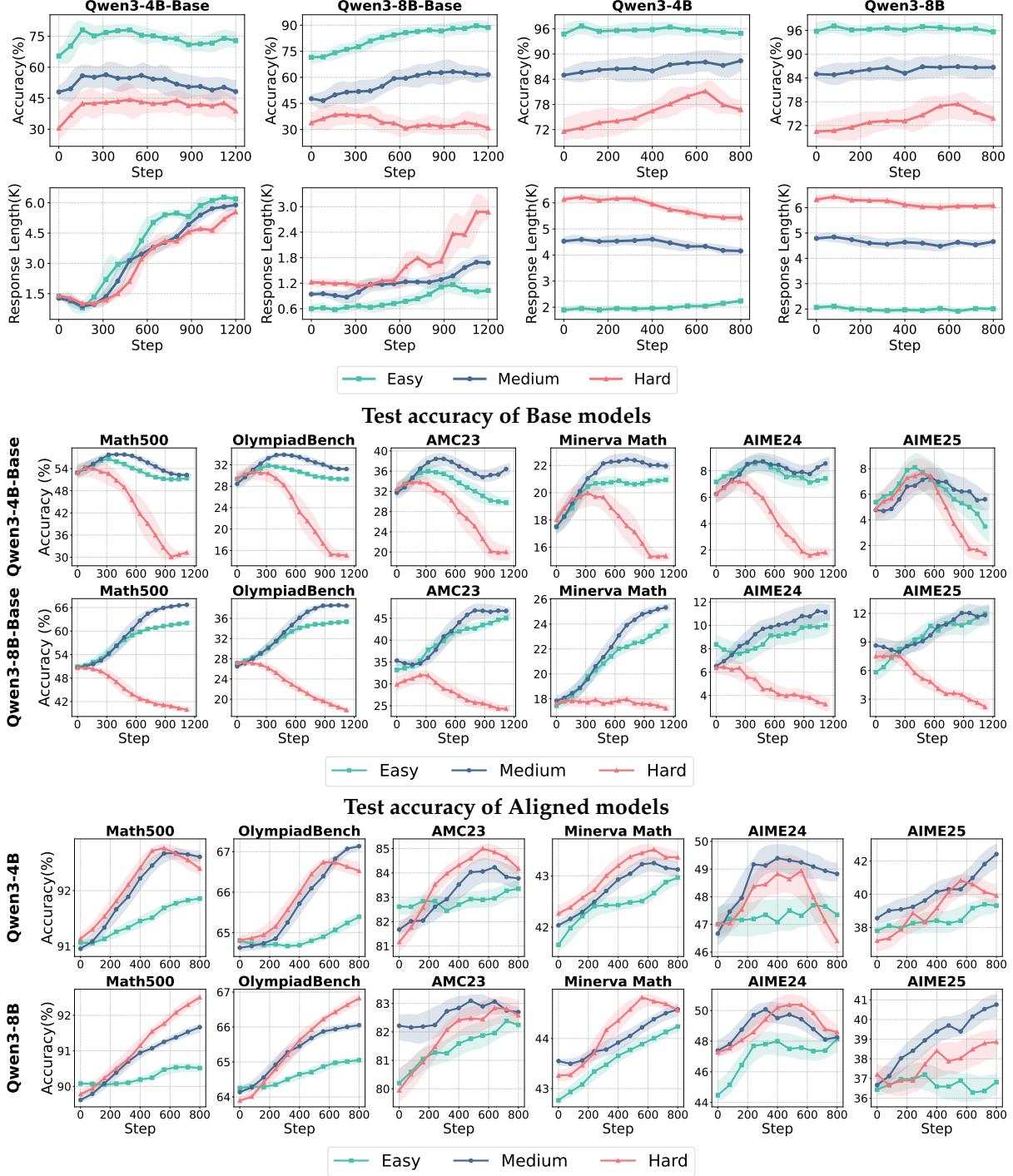


Figure 3: (Top 2 rows): Test accuracy and response length of four model variants: Qwen3-4B-Base, Qwen3-8B-Base, Qwen3-4B, and Qwen3-8B across different data difficulty. **Middle 2 rows:** Accuracy over training iterations of Base models. The first row presents results of Qwen3-4B-Base. The second row shows results of Qwen3-8B-Base. **Bottom 2 rows:** Accuracy over training iterations of aligned models. The first row presents results of Qwen3-4B, while the second row shows results of Qwen3-8B. To ensure clarity and intuitiveness in the qualitative analysis, all curves are consistently smoothed using identical parameters. Specifically, the mean values are computed using an 11-step moving window with an exponential smoothing factor of 0.8. The shaded regions around the curves represent the range $\text{mean} \pm (\text{std_multiplier} \times \text{standard deviation})$, providing a visual representation of the oscillation amplitude.

normalization, arguing that optimizing within a single prompt excessively can lead to reward hacking and hinder generalization, especially when response diversity is low.

Formally, Given a prompt x with K sampled responses and corresponding rewards $\{r_k\}_{k=1}^K$, the group-level normalized advantage for the k -th response is:

$$A_k^{\text{group}} = \frac{r_k - \text{mean}(\{r_j\}_{j=1}^K)}{\text{std}(\{r_j\}_{j=1}^K)}. \quad (5)$$

In contrast, batch normalization computes the reward over a rollout batch of size N and K sampled trajectories. The normalized advantage for the i -th response is:

$$A_i^{\text{batch}} = \frac{r_i - \text{mean}(\{r_j\}_{j=1}^{N*K})}{\text{std}(\{r_j\}_{j=1}^{N*K})} \quad (6)$$

4.1.1 Advantage normalization is sensitive to reward mechanisms

Takeaway 1

Group-level normalization demonstrates robust efficiency across different reward settings. **Batch-level normalisation** provides more stable improvement under large scale reward setting.

To systematically evaluate the impact of advantage normalization on PPO variants with a value function using the Monte Carlo return target, we conducted experiments under a unified training framework, exploring three settings: **no normalization**, **batch-level normalization**, and **group-level normalization**. To highlight the differential impacts of the normalization techniques during training, we selected the Qwen3-base series models due to their low initial scores and high improvement potential (Yang et al., 2025). This choice ensures a fair comparison by minimizing confounding factors from alignment or prior optimization. Focusing on model scale as a key variable, we evaluate small (4B) and medium-sized (8B) models to empirically assess whether normalization techniques interact with model capacity. This approach allows us to derive practical insights into normalization strategies across different computational budgets and architectures.

Under the default setting of the reward mechanism, i.e., $R \in \{0, 1\}$ ³, when analyzing the performance in Figure 4, it can be concluded that both advantage normalization techniques can significantly influence the model's convergence speed, performance stability, and final outcomes. Specifically, on both model sizes, group-level normalization consistently achieves more stable training dynamics and higher final performance compared to both batch-level normalization and no normalization. Batch-level normalization exhibits high sensitivity to reward distribution skew, often leading to performance collapse under an imbalanced batch situation, where a few outlier samples dominate the advantage estimates.

However, when we changed the reward mechanism to the larger scale of $R \in \{-1, 1\}$ ⁴, batch-level normalization regained its effectiveness, demonstrating a significant improvement in policy learning, as shown in Figure 5. The above experiment fully demonstrates the sensitivity of the advantage normalization technique to the reward mechanism.

4.1.2 Impact of the standard deviation term in advantage normalization

Takeaway 2

Removing the standard deviation when reward distributions are highly concentrated (e.g., easy training dataset) enhances the stability and effectiveness of model training.

The previous section highlighted the sensitivity of various normalization techniques to the reward scale. Thus, a question naturally emerged: *what drives this phenomenon?* A plausible explanation is that different reward scales directly impact the calculation of the standard deviation, thereby altering the strength of the normalization. In particular, when model responses within a prompt group yield highly similar rewards, e.g., when the responses are almost all correct or all incorrect, the resulting standard deviation becomes extremely small. In such cases, dividing by this small standard deviation during normalization

³ $R \in \{0, 1\}$ represents the default rule-based binary reward mechanism, where a value of 1 is assigned to trajectories that generate correct answers, and a value of 0 is assigned to incorrect ones.

⁴ $R \in \{-1, 1\}$ further increases the magnitude of reward differences compared to the default mechanism, where a value of 1 is assigned to trajectories that generate correct answers, and a value of -1 is assigned to incorrect ones.

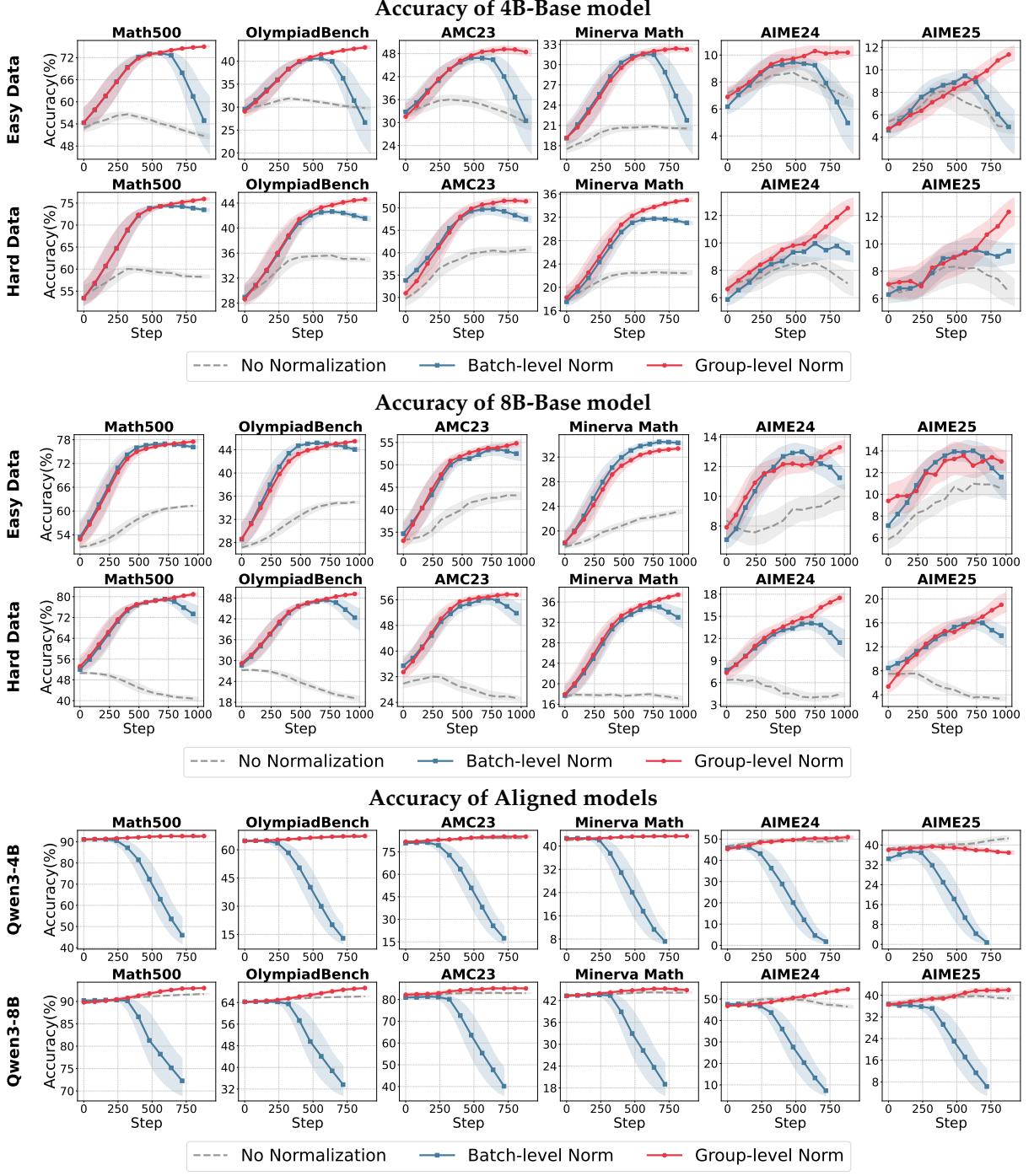


Figure 4: Accuracy over training iterations of Base models. **Top 2 rows:** Qwen3-4B-Base with different normalization techniques. The first row uses the easy training dataset, while the second row uses the hard training dataset. **Middle 2 rows:** Qwen3-8B-Base with different normalization techniques (under the default reward scale). **Bottom 2 rows:** Accuracy over training iterations of aligned models (trained on medium level dataset, under the default reward scale) with different normalization techniques. The first row shows the results of Qwen3-4B, while the second row shows the results of Qwen3-8B.

can excessively amplify gradient updates, causing the model to overemphasize tasks of extreme difficulty, a phenomenon similar to “difficulty bias” (Liu et al., 2025a).

To determine whether the calculation method of the standard deviation is the key module causing the difference in normalization performance, we employ the batch-level calculation, which exhibited unstable performance in the previous section, to calculate the mean of advantage, and conduct ablation

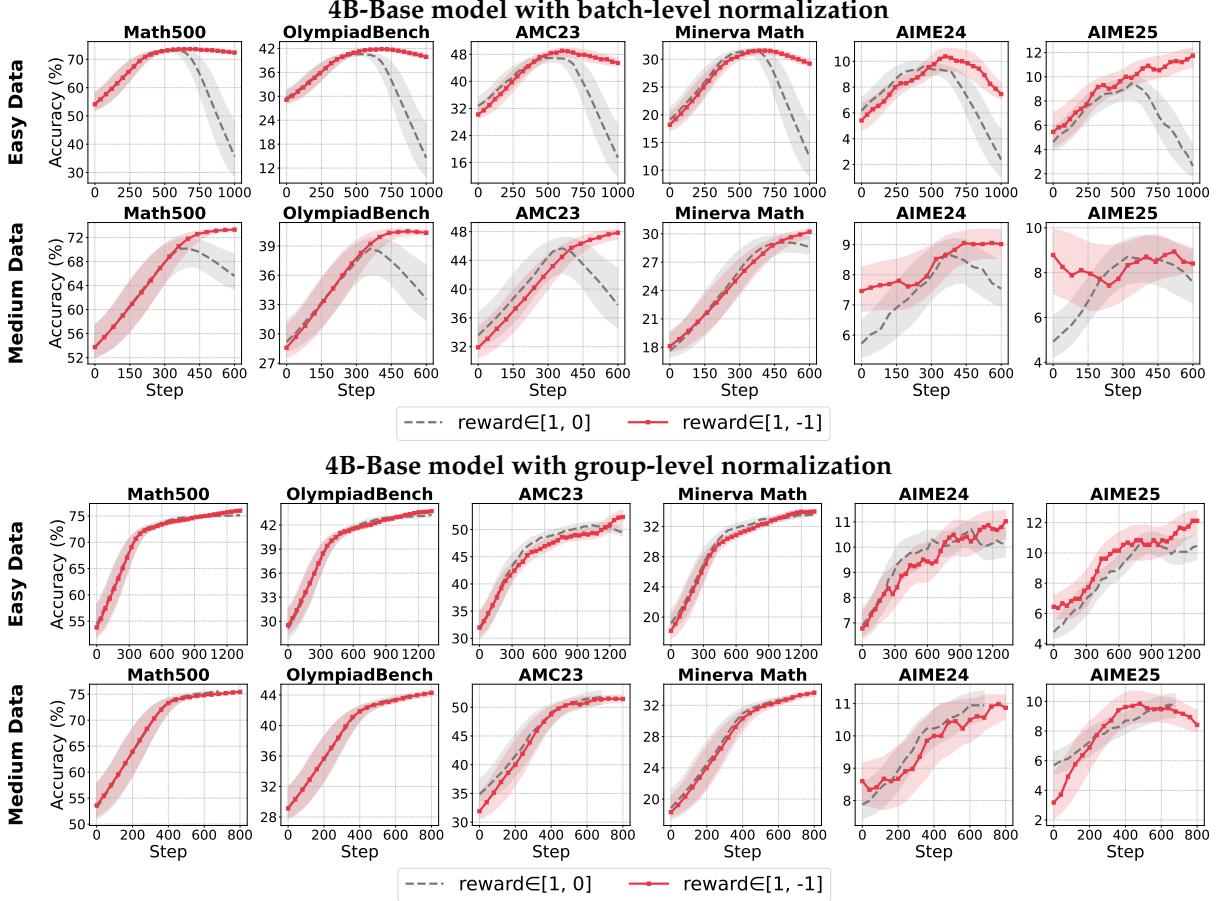


Figure 5: **Top 2 rows:** Accuracy over training iterations of Qwen3-4B-Base with batch-level normalization under different reward scale. The first row uses the easy training dataset, while the second row uses the medium training dataset. **Bottom 2 rows:** Accuracy over training iterations of Qwen3-4B-Base with group-level normalization under different reward scale.

experiments on the standard deviation term. This can be formalized as:

$$A_k^{\text{std}^-} = r_k - \text{mean}(\{r_j\}_{j=1}^K). \quad (7)$$

We separately recorded the accuracy after training on simple and difficult data. The curves of easy data in Figure 6 show that the policy rapidly converges to highly consistent behaviors, leading to a highly concentrated distribution of reward values. Correspondingly, the standard deviation of the reward distribution swiftly declines to a low value. Applying standard deviation-based normalization in this setting results in an exceedingly small denominator, which excessively amplifies reward and advantage values. This, in turn, induces abnormally large gradients, destabilizes training, and can even trigger gradient explosions. Therefore, these experimental results empirically verify our conjecture that the standard deviation term is the key mechanism for the advantage normalization.

To further solidify our conclusion, we add a set of comparisons based on the hard dataset. We observe that the standard deviation of rewards remains comparatively high during training. As a result, both mean-only normalization and standard deviation based normalization yield similar efficiency, and training remains stable regardless of the normalization style. Consequently, the choice of normalization style has little impact on convergence or overall performance under such a smooth reward distribution.

In summary, our experiments and analysis underscore that, in scenarios where reward distributions are highly concentrated, omitting the standard deviation from advantage normalization effectively prevents abnormal gradient amplification, thereby improving the stability and robustness of model training. However, for tasks characterized by inherently higher reward variance, either normalization approach is generally sufficient to maintain stable optimization.

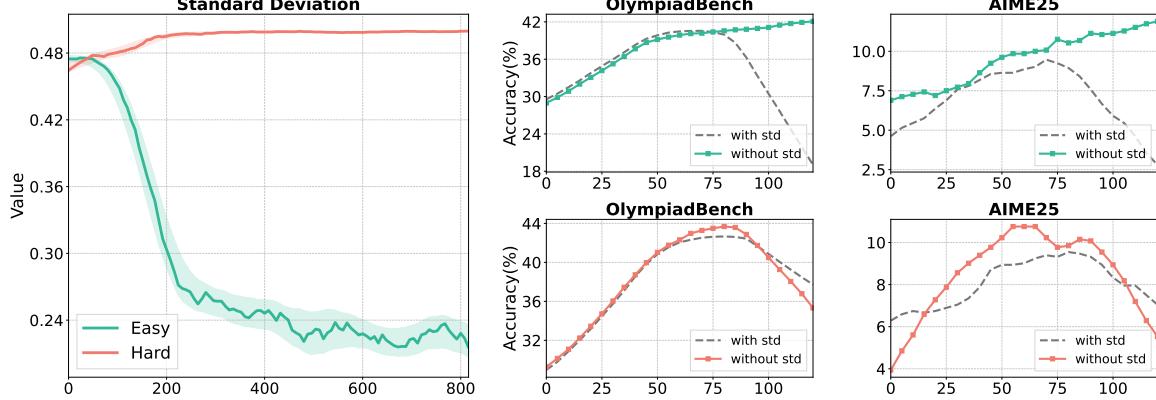


Figure 6: **Left:** Standard deviation variations during training on datasets of different difficulty levels. **Right:** Test accuracy before and after removing standard deviation from batch level normalization, with results for training on Easy Data (top) and Hard Data (bottom).

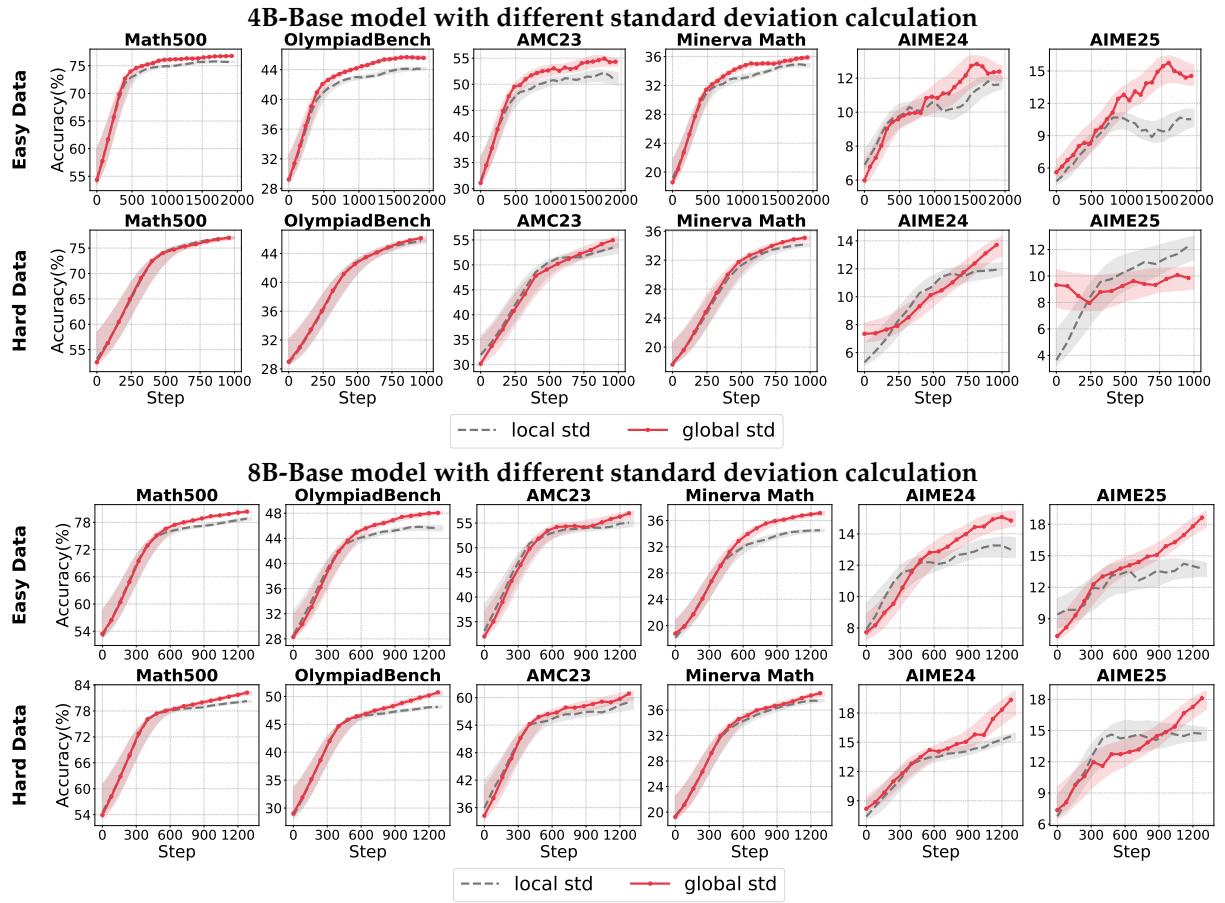


Figure 7: Accuracy comparison of Base models with different standard deviation calculation. **Top 2 rows:** Accuracy of Qwen3-4B-Base with different standard deviation calculation. The first row uses the easy training dataset, while the second row uses the hard training dataset. **Bottom 2 rows:** Accuracy comparison of Qwen3-8B-Base with different standard deviation calculation. The first row uses the easy training dataset, while the second row uses the hard training dataset.

4.1.3 Reconstruct a robust normalization technique

Takeaway 3

Calculating the mean at the local (group) level and the standard deviation at the global (batch) level enables more robust reward shaping.

Section 4.1.2 highlights the critical role of the standard deviation in determining the effectiveness of the advantage normalization mechanism. This leads to the final requirement: Is there a more robust and effective combination of mean and standard deviation for reward shaping? To explore this, we adopted the stable group-level mean calculation method demonstrated in section 4.1.1, paired with two approaches for computing the standard deviation: local (group-level) and global (batch-level). We then evaluated the performance of these combinations across two model sizes.

The results, presented in Figures 7, reveal that global-level calculation exhibits a clear advantage. We attribute this to the batch-level standard deviation providing stronger normalization by effectively reducing gradient magnitudes, thereby preventing excessive policy updates. This approach aligns more effectively with the biased reward signals common in sparse rewards and coarse-grained advantage fitting, resulting in more stable and robust learning behavior. Furthermore, our experimental results support a claim from Hu et al. (2025) that batch-level normalization, or even subtracting the local mean and dividing by the batch standard deviation in certain scenarios, performs better.

4.2 Clip-Higher

While the Clip mechanism enhances PPO training stability (Huang et al., 2024b), it introduces critical challenges in LLM-based text generation. Specifically, it disproportionately suppresses low-probability tokens (Yu et al., 2025), leading to entropy collapse, i.e., a state where strategies become deterministic and lack diversity (Jin et al., 2024). This suppression creates a harmful positive feedback loop: as training progresses, entropy decreases, exploration shrinks, high-probability patterns are further reinforced, and entropy declines even more. Such behavior severely hinders performance on complex reasoning tasks, where novel path exploration is essential. To address this, the Clip-Higher mechanism is widely introduced into the training objective, which can be formalized as:

$$J_{DAPo}(\theta) = (r_{i,t}(\theta), 1 - \varepsilon_{low}, 1 + \varepsilon_{high}). \quad (8)$$

ε_{high} denotes the upper bound of the Clip mechanism and ε_{low} represents the lower bound. Unlike the original clip that enforces proportional fairness, Clip-Higher introduces a higher upper bound for advantage, giving low-probability tokens more improving space. By expanding exploration potential in low-probability regions, this technique effectively mitigates entropy collapse. However, the lack of in-depth analysis of the underlying mechanism and the absence of detailed usage guidelines have left practitioners confused about the appropriate scenarios for using Clip-Higher, as well as the ideal upper bound settings under different conditions. In this section, we address the aforementioned remaining issues through a series of comprehensive experiments.

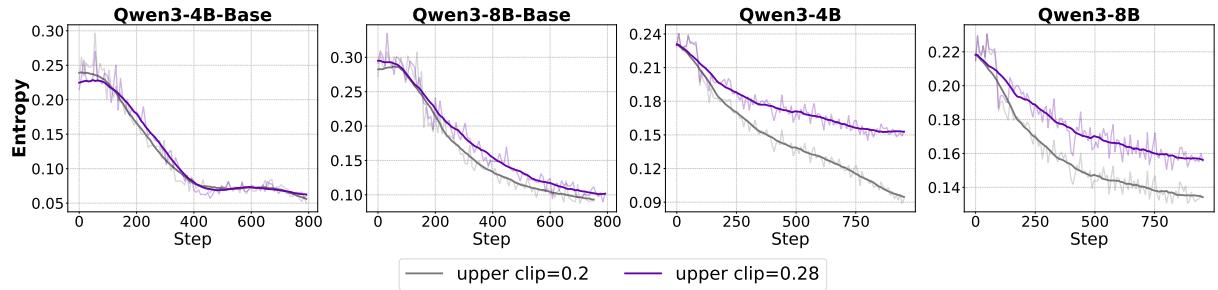


Figure 8: Entropy comparison across different models with Clip-Higher. A higher clip upper bound can mitigate the entropy drop in aligned models.

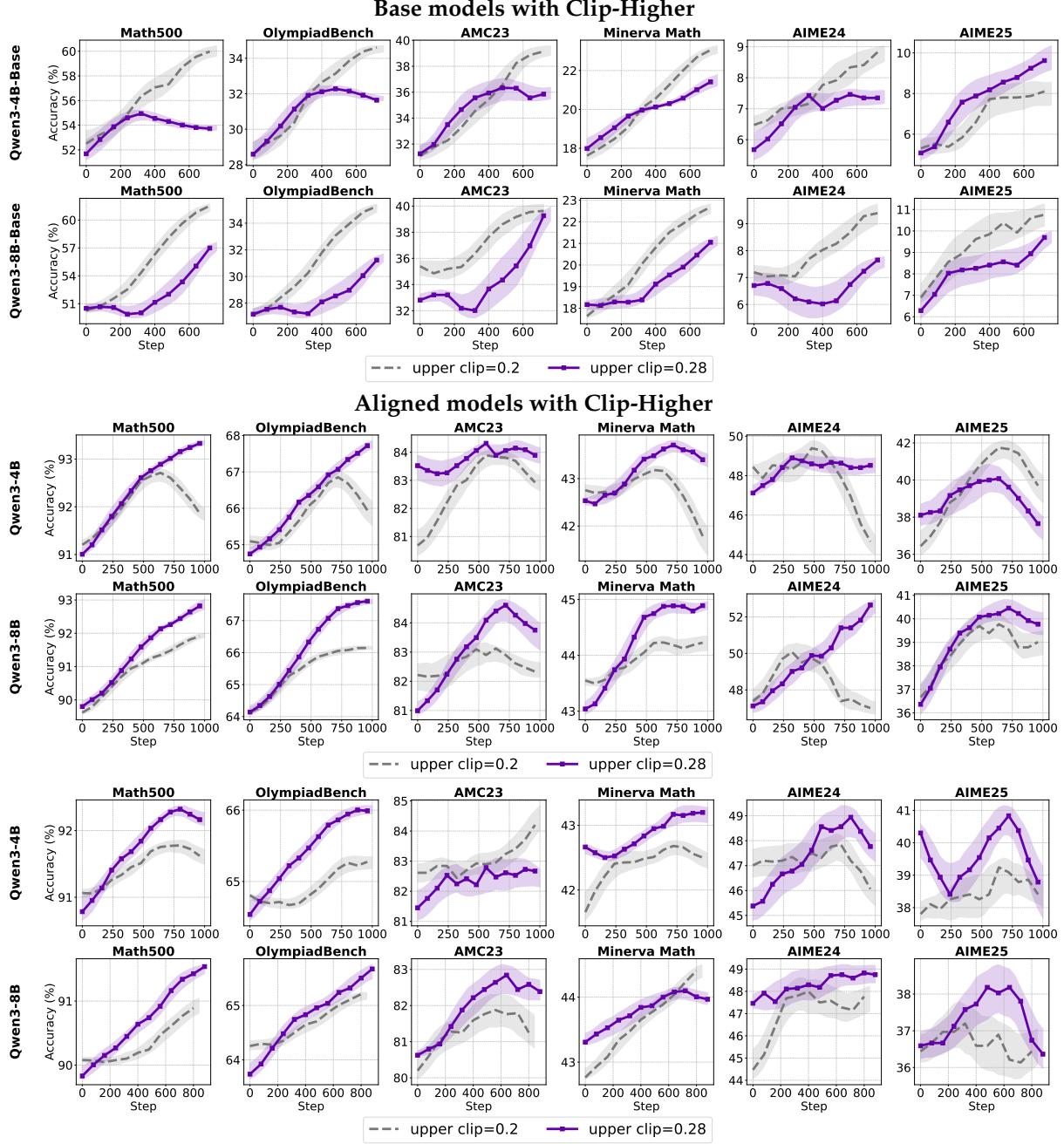


Figure 9: **Top 2 rows:** Test accuracy of Base models (trained on medium data) with higher clipping upper bound. **Middle 2 rows:** Test accuracy of aligned models (trained on medium data) with higher clipping upper bound. **Bottom 2 rows:** Test accuracy of aligned models (trained on easy data) with a higher clipping upper bound.

4.2.1 In which settings should we clip higher

Takeaway 4

For models with stronger fundamental reasoning abilities, increasing the clip higher parameter is more likely to facilitate exploration of better solution paths.

Through extensive empirical practice, we observe that the advantage clip technique demonstrates distinct effectiveness across different model architectures. To examine this, this section employs the non-aligned (base) model and the aligned (instruct) model with various sizes to clearly demonstrate the sensitivity of the Clip mechanism, summarize the usage guidelines for Clip higher from a modeling perspective.

As illustrated in Figure 8, experimental results indicate that the impact of increasing the upper clipping bound ϵ_{high} is model-dependent. For the base models, adjusting the upper clipping value yields minor effects on policy entropy and even damages the performance compared to the vanilla policy (as shown in the top 2 rows of Figure 9). In contrast, aligned models exhibit a markedly different response: raising the upper clipping bound notably slows the entropy collapse, leading to consistent performance improvements in downstream evaluation metrics (refer to the middle and bottom rows in Figure 9).

This disparity can be attributed to several underlying factors. First, the base models operate with a low policy clipping rate, approximately 0.003, which indicates only minimal deviation between successive policies. Moreover, the relatively naive policy expressiveness limits these base models' capacity for exploration, hindering the discovery of high-reward trajectories. Consequently, a higher clipping upper bound yields negligible improvements in learning dynamics.

On the other hand, aligned models that leverage advanced pre-training techniques or post-training enhancements demonstrate superior reasoning capabilities and generalization performance (Yang et al., 2025). As shown in Figure 10, compared to the base model, the aligned model has very few preferred tokens with high probability in the initial stage. Token distributions for larger-scale models are provided in Appendix D. Therefore, a higher clipping upper bound can effectively bridge the probability gap between tokens and alleviate the entropy collapse. For these models, raising the upper bound expands the permissible range of policy updates, which in turn facilitates more diverse action sampling and enhances exploratory behavior during training. This mechanism preserves higher entropy while simultaneously increasing the probability of identifying optimal solutions, as evidenced by improved evaluation metrics.

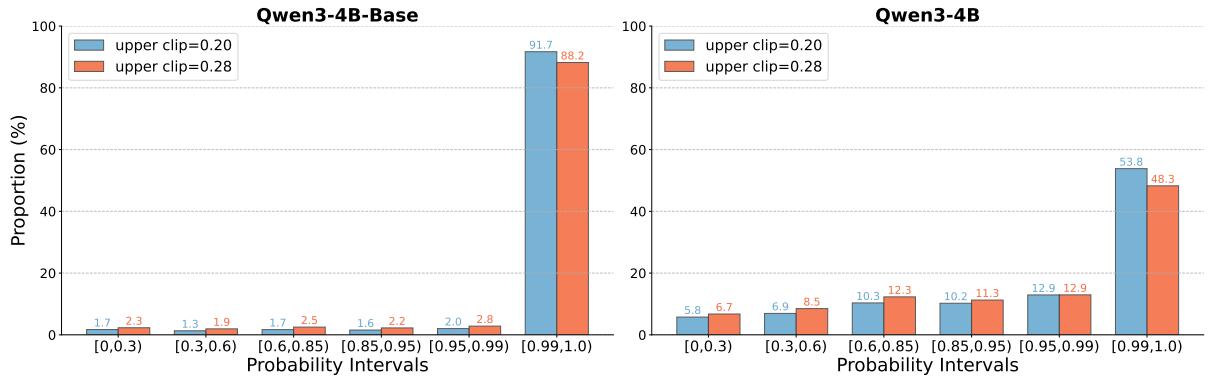


Figure 10: Predicted probability distributions of Qwen3-4B-Base (left) and Qwen3-4B (right) under two clipping upper bound $\in \{0.20, 0.28\}$.

4.2.2 Analyzing the effectiveness of Clip-Higher from a linguistic perspective

Takeaway 5

Traditional clipping may restrict the model's capacity to generate innovative reasoning structures.
Clipping higher allows the model to explore a broader range of discourse reasoning structures.

Building on our token-level demonstration of Clip-Higher's behavior in section 4.2.1, we now analyze its impact on reasoning logic through token-level linguistics. As illustrated in Figure 11, setting an upper bound to 0.2 imposes stringent constraints on policy updates by limiting substantial probability deviations for individual tokens. Under these stricter conditions, our analysis reveals that clipping predominantly affects connective tokens such as “therefore”, “if”, and “but”. These tokens frequently appear at the beginnings of sentences, serving as key semantic markers or transition words within dialog generation. Such connectors often introduce new directions in reasoning. However, their probability ratios between updated and old policies frequently exceed clipping thresholds, triggering aggressive suppression in PPO optimization. While this traditional clipping ensures stability in the overall token distribution, it may restrict the model's capacity to generate innovative or diverse argumentative reasoning structures by constraining flexibility in the use of discourse-level connectives.

Furthermore, raising the upper bound from 0.2 to 0.28 significantly expands the policy update space, permitting greater deviations in token-level probabilities from the old policy. Under these more permissive conditions, our analysis indicates that the frequency of clipped tokens decreases markedly,

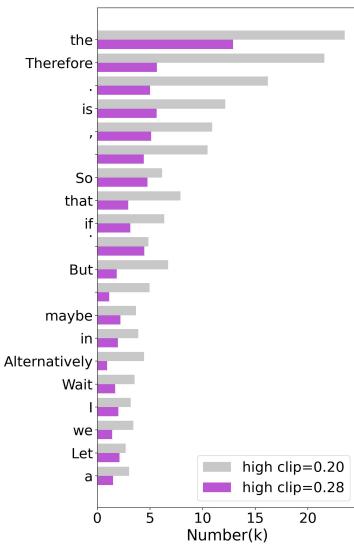
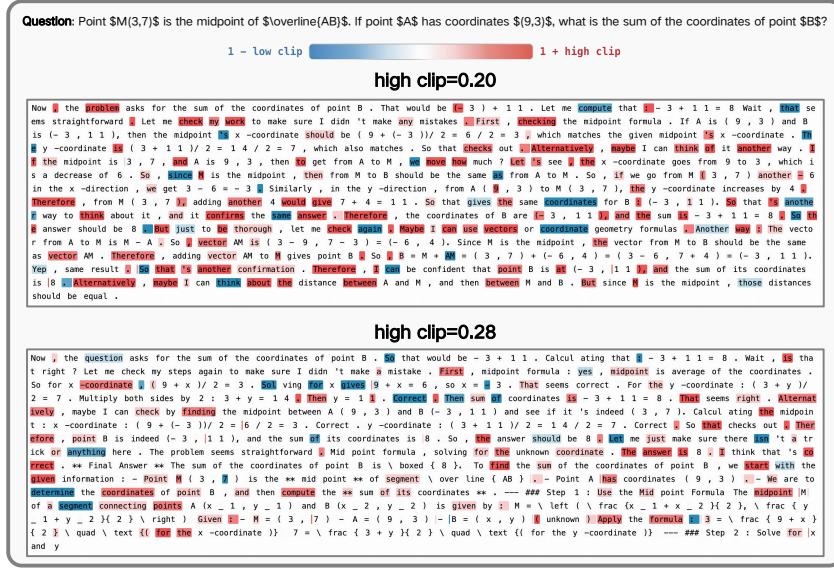


Figure 11: Left: A case study under the same prompt across various clipping upper bounds. Right: The trigger differences of various upper bounds at the top 20 tokens with the highest clip frequencies.

with the focus of clipping shifting away from discourse connectives toward high-frequency functional tokens such as “*is*”, “*the*”, and “*.*”. These tokens are prevalent within sentences and exhibit relatively weak contextual dependencies, making their probability estimates highly sensitive to fluctuations in the probability difference between the sampling and training policies. This transition allows the model to explore a broader range of discourse reasoning structures and promotes diversity in response generation. Besides, the remaining clipping action on common function words serves to maintain the stability of the core sentence structure.

4.2.3 How to set the upper bound for advantage clipping

Takeaway 6

There appears to be a “scaling law” between the performance and the upper bound of the clipping on the **small-sized model**, which does not exist on **larger models**.

Section 4.2.1 verifies that Clip-Higher showed significant improvements on aligned models. However, most current works directly set the upper bound of Clip to the default value of 0.28 from (Yu et al., 2025). However, we believe that different models have different preferences for this parameter. To verify this conjecture, we empirically searched for the hyperparameter settings applicable to different aligned models by uniformly setting the upper bound of Clip. Specifically, we set the exploration range of the Clip upper bound from the default threshold of 0.2 from traditional Clip to 0.32 (beyond the widely used upper bound 0.28). We employed two sizes of models and uniformly evaluated their learning capabilities under different settings.

The results in Figure 12 show that for the small-sized model (4B), the model performance gradually improves as the upper bound of the clip increases. And at 0.32, it demonstrates the best performance compared to other settings. On the other hand, for larger model sizes (8B), gradually increasing the upper bound of the clip does not show a progressive improvement. The performance is more prominent when the upper bound is set as 0.28.

4.3 Loss Aggregation

The strategy of loss aggregation directly determines the contribution of each sample or token to the overall gradient during optimization (Liu et al., 2025b). Common strategies include token-level and sequence-level aggregation. The sequence-level aggregation adopted by GRPO (Shao et al., 2024) first averages the loss across all tokens within each sample, then averages these per-response losses across the batch, thereby assigning equal weight to each response regardless of its length. However, Yu et al. (2025) highlights a flaw in this method: longer responses possess a diminished influence per token on the

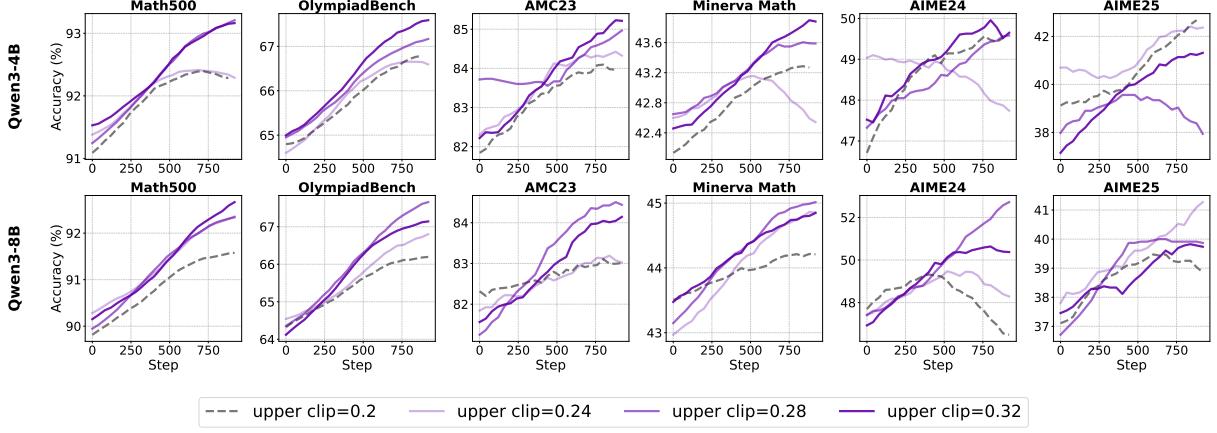


Figure 12: Test accuracy of aligned models (trained on medium data) with various clipping upper bounds.

total loss, hindering the model’s ability to learn effectively from diverse quality reasoning in lengthier responses. This can reduce the model’s capacity to learn from long, complex answers, and may bias optimization toward brevity, since shorter correct responses receive larger gradient updates, while longer incorrect responses are insufficiently penalized (Liu et al., 2025a).

$$\begin{aligned} \mathcal{J}_{\text{sequence-level}}(\theta) &= \mathbb{E}_{(q,a) \sim \mathcal{D}, \{o_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}(\cdot|q)} \\ &\left[\frac{1}{G} \sum_{i=1}^G \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \min \left(r_{i,t}(\theta) \hat{A}_{i,t}, \text{clip} \left(r_{i,t}(\theta), 1 - \epsilon_{\text{low}}, 1 + \epsilon_{\text{high}} \right) \hat{A}_{i,t} \right) \right] \end{aligned}$$

$$\begin{aligned} \mathcal{J}_{\text{token-level}}(\theta) &= \mathbb{E}_{(q,a) \sim \mathcal{D}, \{o_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}(\cdot|q)} \\ &\left[\frac{1}{\sum_{i=1}^G |o_i|} \sum_{i=1}^G \sum_{t=1}^{|o_i|} \min \left(r_{i,t}(\theta) \hat{A}_{i,t}, \text{clip} \left(r_{i,t}(\theta), 1 - \epsilon_{\text{low}}, 1 + \epsilon_{\text{high}} \right) \hat{A}_{i,t} \right) \right] \end{aligned}$$

In response to this issue, Yu et al. (2025) turns to a token-level calculation approach. Here, losses are calculated by summing the loss across all tokens from all samples and then normalizing by the total token count, guaranteeing an equal contribution from each token regardless of response length. Despite the widespread adoption of these methods, existing analyses remain trivial. In this section, we provide a detailed empirical comparison of the two loss calculation techniques across diverse training data distributions. The evaluation comprehensively assesses the effectiveness of these methods from the perspective of model type.

4.3.1 Does token-level loss aggregation suit all settings?

Takeaway 7

Compared to sequence-level calculation, token-level loss proves to be more effective on Base models, while showing limited improvement on Instruct models.

To systematically evaluate the effectiveness of different loss aggregation strategies, we compare token-level and sequence-level loss aggregation on both base and aligned versions of Qwen3-8B, as shown in Figures 13 and 18. For base models, token-level loss consistently improves convergence, peak accuracy, and robustness by ensuring each token contributes equally to the optimization signal, especially on challenging datasets. However, as illustrated in Figure 13 (bottom 2 rows), this advantage does not show in aligned models. In fact, sequence-level aggregation outperforms token-level loss across most datasets and settings, both in convergence speed and final accuracy. Further analysis reveals that aligned models already possess strong and stable reasoning, making the equalization of token-level gradients unnecessary or even detrimental. In these cases, sequence-level aggregation better preserves the structure and consistency of high-quality, aligned outputs.

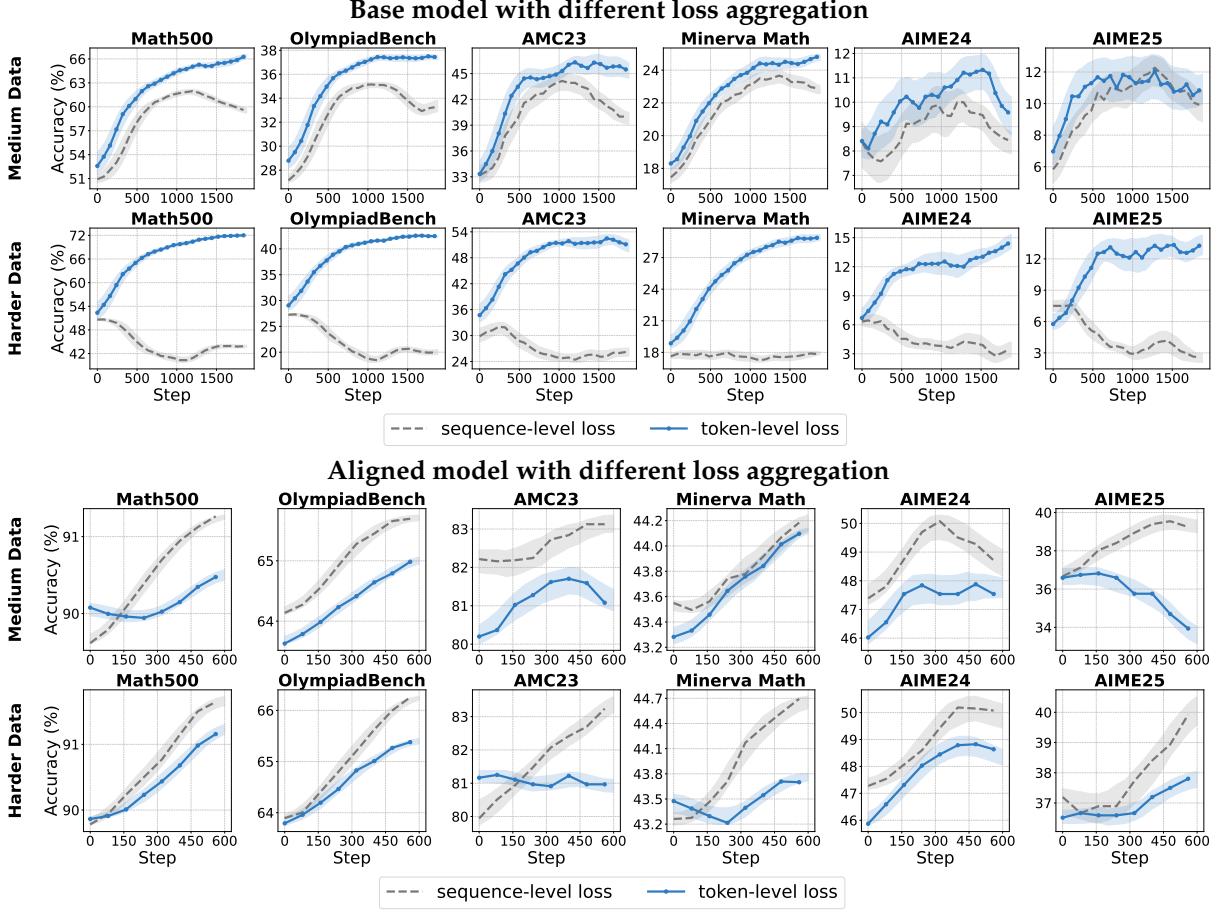


Figure 13: **Top 2 rows:** Accuracy comparison between sequence-level loss and token-level loss. Qwen3-8B-Base is used as the initial policy. Results are reported on both Easy and Hard Datasets. **Bottom 2 rows:** Test accuracy of Qwen3-8B with different loss aggregations.

These findings highlight that the optimal loss aggregation strategy is model-dependent, currently from a broader perspective: token-level aggregation is best suited for base models, while response-level aggregation is preferable for instruction-tuned models.

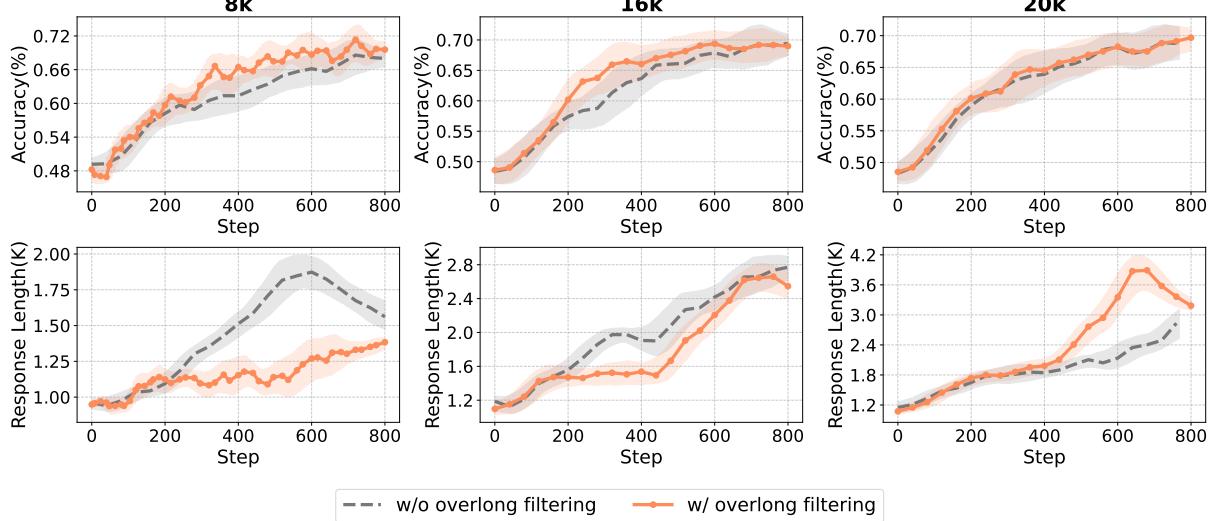
4.4 Overlong Filtering

During the training of LLMs, a fixed maximum generation length is often set for truncation to ensure training efficiency and save computational costs (Chen et al., 2025; Team et al., 2025). However, recent studies have revealed that in more complex reasoning tasks, this strategy can prematurely end multi-step tail reasoning processes, particularly noticeable in the early training stages. Consequently, coherent and well-structured reasoning is often cut short before reaching the final answer, causing them to be falsely labeled as negative samples by the model. This noise, akin to penalties, can contaminate the training signal, reducing sample utilization efficiency and learning effectiveness.

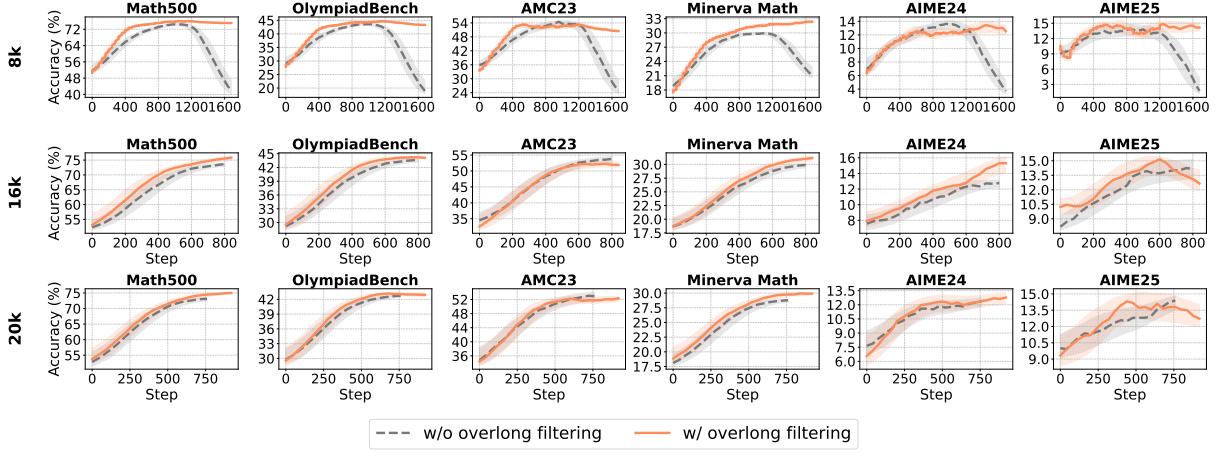
To address this issue, the technique named *overlong filtering* has been introduced (Yu et al., 2025). This method involves masking the reward signal of excessively long responses to preserve training loss robustness and prevent degradation of reasoning behavior (He et al., 2025b). Despite its benefits, there remains a lack of detailed analysis regarding the sensitivity of this technique to the mask threshold, leading to confusion among practitioners.

This section aims to analyze the impact of the overlong filtering on performance across diverse datasets under varying maximum generation length settings. By doing so, we seek to identify the suitable scenarios for applying this technique.

Overview of training accuracy and response length of 8B-Base model



Test accuracy of 8B-Base model



Test accuracy of 8B-Aligned model

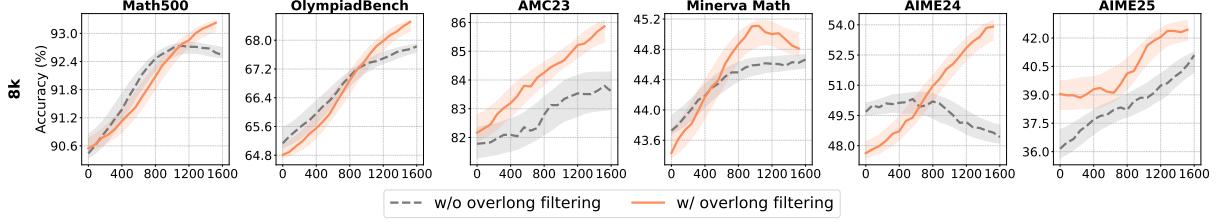


Figure 14: **Top 2 rows:** Total test accuracy and response length of Qwen3-8B-Base over training iterations under different maximum generation lengths. **Middle 3 rows:** Test accuracy of Qwen3-8B-Base over training iterations under different maximum lengths. We set different maximum lengths of 8k, 16k and 20k. **Middle 3 rows:** Validation of overlong mask effectiveness on Qwen3-8B.

4.4.1 When to use the overlong filtering

Takeaway 8

Overlong filtering shows limited effectiveness on long-tail reasoning tasks; however, it can enhance the accuracy and clarity of responses in medium and short-length reasoning tasks.

Although recent works have verified the benefits of overlong filtering for policy training (Team et al., 2025; Chen et al., 2025), however, the impact of different maximum lengths on this technique is still unclear. Therefore, we employ the widely used Qwen3-8B-Base and Qwen3-8B as the unified initial policy to compare the effects of different maximum generation lengths on the training dynamics.

The results in Figure 14 highlight the different impact on learning dynamics of various filter thresholds. Notably, when the filter threshold is restricted to 8k tokens, substantial benefits are evident from implementing the overlong filtering. However, with a longer filter threshold, i.e., 20k tokens, the benefits derived from this technique diminish significantly. After checking the response lengths, a discernible pattern emerges to explain this phenomenon. When operating under the threshold of 20k, models trained with the overlong filtering strategy exhibit a tendency to generate longer responses in comparison to the vanilla policy. Conversely, a short filter threshold, i.e., 8k, makes the model generate shorter responses.

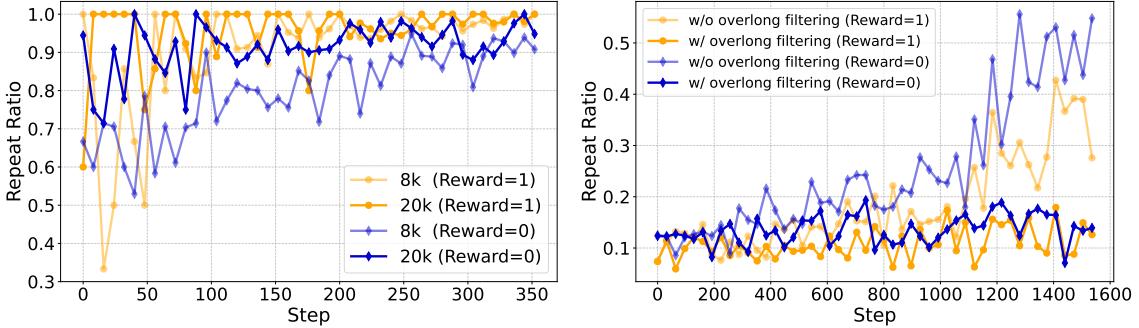


Figure 15: **Left:** Comparison of repeat ratios among four types of generations, i.e., correct (reward = 1) and incorrect (reward = 0) generations under different maximum generation lengths. **Right:** Comparison of repeat ratios among truncated samples with or without overlong filtering strategy. The statistical form of the repetition rate can be found in Appendix B.1.

To further investigate this effect, Figure 15 (Left) shows the distribution of filtered responses exceeding the maximum length. Notably, in the 20k setting, both correct and incorrect samples are filtered more frequently due to repetitive or non-terminating outputs, a hallmark of degenerate generation. This indicates that, with higher length limits, the overlong filtering strategy primarily filters out unproductive or “negative” samples that contribute little to model learning. As illustrated in Figure 15 (Right), we observed that during RL training on models fine-tuned with instructions, the proportion of “repetitive but unable to terminate normally” samples within the overall set of overlong samples gradually increased as training progressed. This indicates a degradation in the model’s ability to accurately model end-of-sequence (EOS) tokens, leading to behavioral defects in the inference stage, such as output redundancy and hard in terminating the generation. After introducing the overlong filtering mechanism, the proportion of abnormal samples that are “repetitive but unable to terminate” significantly decreased during training. This shift suggests that the model can more accurately distinguish between “completed generation” and “truncated generation” samples during training, effectively avoiding invalid learning from truncated samples. Furthermore, this mechanism may unlock the policies’ ability to accurately model termination behaviors during generation, enabling them to appropriately ignore unfinished inference samples, rather than mistakenly penalizing them as negative examples.

5 A simple combination: Lite PPO

Building on the in-depth mechanism analysis and empirical evaluations presented in previous sections, we derive two key technique guidelines for non-aligned models: (i) For small and medium-sized non-aligned models, i.e., 4B-Base and 8B-Base, the technique that can provide significant performance improvement is the advantage normalization introduced in section 4.1.3. This technique shapes sparse rewards into more robust guiding signals through group-level mean calculation and batch-level standard deviation calculation. (ii) Token-level loss aggregation emerges as another highly effective technique for non-aligned models, with Section 4.3.1 experiments demonstrating its particular efficacy for base model architectures.

We therefore propose the following empirically motivated hypothesis: Given the individually superior performance of advantage normalization (group-level mean, batch-level std) and token-level loss aggregation over alternative techniques, their synergistic combination should show robust improvements in policy optimization. To validate this, we integrate both techniques, called Lite PPO, into non-aligned models that use the vanilla PPO loss without the critic. The results shown in Figure 16 indicate that Lite PPO outperforms the technique-heavy algorithm DAPO, which involves *Group-level Normalization*, *Clip-Higher*, *Overlong Reward Shaping*, *Token-level Loss*, *Dynamic Sampling*, and the strong and widely-used RL4LM algorithm GRPO.

Specifically, in the first two rows of Figure 16, Lite PPO exhibits a stable upward trend on small models

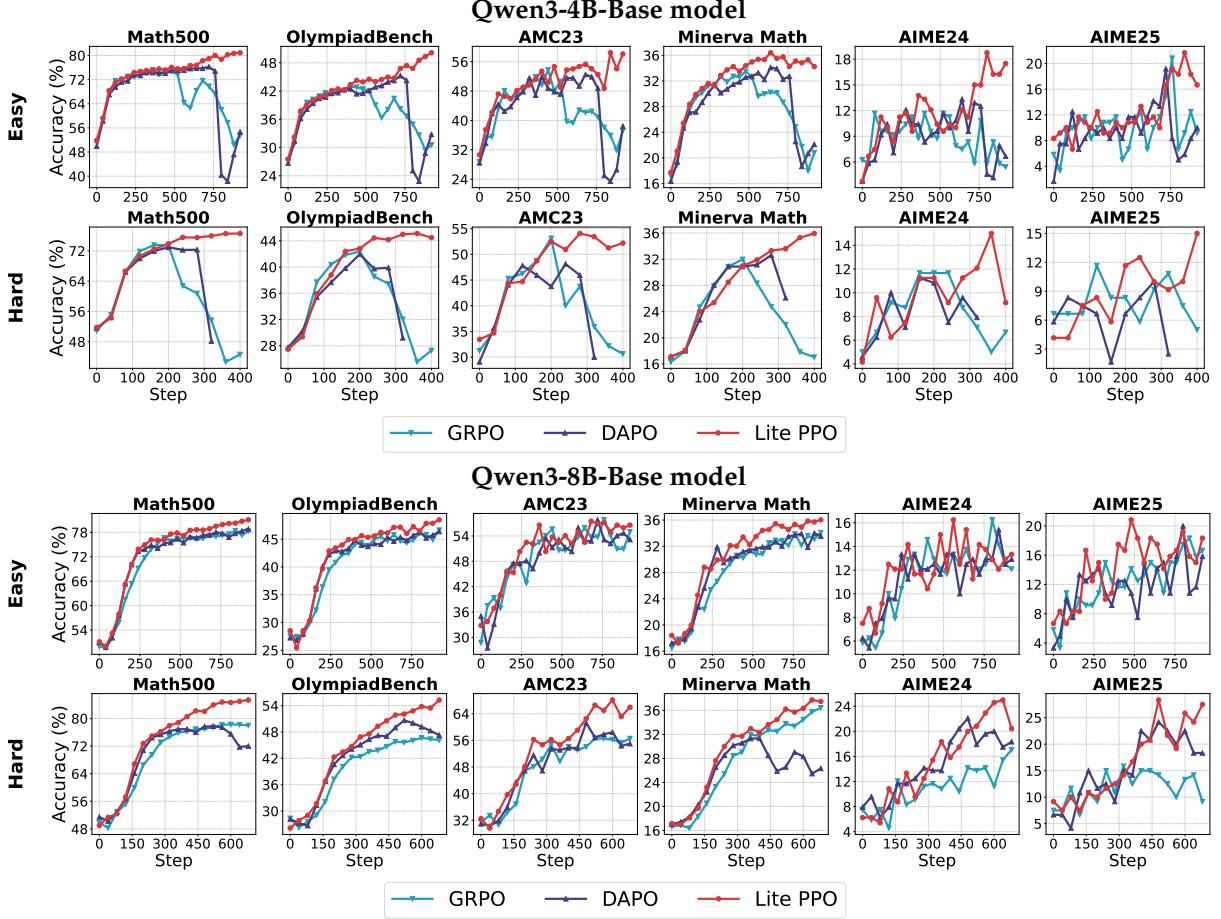


Figure 16: Test accuracy of non-aligned models trained via three RL methods, i.e., Lite PPO (ours), GRPO (Shao et al., 2024) and DAPO (Yu et al., 2025).

lacking basic reasoning ability. In contrast, other policies collapse rapidly after reaching their peak. This significant advantage results from the normalization technique introduced in [Takeaway 3](#), which effectively counters the interference induced by homogeneous reward distributions characteristic of datasets with non-uniform reward levels (easy and hard). We further evaluate Lite PPO on larger base models. As shown in Figure 16, when training 8B-Base models with inherent long-tail generation capabilities on the hard dataset, Lite PPO also demonstrates superior performance. This improvement stems from Lite PPO eliminating overlong filtering (which typically restricts small models’ ability to generate complex long-tail outputs; [Takeaway 8](#)), and shifting to token-level loss aggregation (which shows better efficiency on base models; [Takeaway 7](#)).

6 Conclusion

The rapid advancement of reinforcement learning (RL) in enhancing large language models (LLMs) has ushered in a transformative era for complex reasoning tasks. However, the proliferation of RL4LLM research has also introduced significant challenges, including conflicting methodologies and a lack of cohesive guidelines for technique selection. This work addresses these critical issues by conducting a systematic, reproducible evaluation of prominent RL techniques under a unified framework, revealing key insights that resolve existing ambiguities and streamline practical implementation.

By disentangling the theoretical and practical mechanisms of techniques like normalization, clipping, and filtering, our study provides actionable guidelines to demystify their applicability across diverse scenarios. Crucially, we show that simplicity can outperform complexity: a minimalist approach (i.e., Lite PPO) combining only two core techniques achieves superior performance over algorithms cluttered with redundant components. This finding challenges the prevailing trend of over-engineering RL pipelines and underscores the importance of contextual adaptability in technique selection. Our work not only resolves the current fragmentation in RL4LLM practice but also lays a foundation for developing standardized frameworks that balance theoretical rigor with engineering efficiency.

Finally, to ensure experimental fairness, this paper consistently uses the Qwen3 series model for policy initialization. However, conclusions may vary across LLM families due to inherent differences in pre-training processes and architectures. The prevailing trend of model closed-sourcing, often driven by commercial or strategic considerations, significantly impedes model-family-level technical analysis. Therefore, we advocate for increased disclosure of implementation details in future technical reports within the industry. This transparency is crucial to bridge the understanding gap between academia and industry, enabling the community to pool collective insights in artificial intelligence.

7 Future work

We envision this work as the starting point of a sustained effort to guide the evolution of reinforcement learning for LLMs along principled and empirically grounded trajectories. Our future research will focus on: (1) continue to monitoring and critically evaluating developments in RL4LLM, distilling emerging practices into coherent, evidence-based guidelines for both academic and industrial practitioners; (2) leveraging the proposed **ROLL** framework to consolidate diverse RL algorithms and optimization strategies into a unified, modular suite, enabling flexible composition and benchmarking within a consistent training infrastructure; (3) continuing to explore streamlined RL algorithms that deliver strong empirical performance with minimal engineering overhead. These directions align with our long-term vision to provide the community with clear and reliable guidance, driving the field toward robust, adaptable, and broadly beneficial progress, while advancing RL4LLM through both algorithmic innovations and comprehensive framework support.

References

- Siwei Wu, Zhongyuan Peng, Xinrun Du, Tuney Zheng, Minghao Liu, Jialong Wu, Jiachen Ma, Yizhi Li, Jian Yang, Wangchunshu Zhou, Qunshu Lin, Junbo Zhao, Zhaoxiang Zhang, Wenhao Huang, Ge Zhang, Chenghua Lin, and Jiaheng Liu. A comparative study on reasoning patterns of openai’s o1 model. *CoRR*, abs/2410.13639, 2024. doi: 10.48550/ARXIV.2410.13639. URL <https://doi.org/10.48550/arXiv.2410.13639>.
- 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. *arXiv preprint arXiv: 2402.03300*, 2024. URL <https://arxiv.org/abs/2402.03300v3>.
- Zhiwei He, Tian Liang, Jiahao Xu, Qiuwei Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *CoRR*, abs/2504.11456, 2025a. doi: 10.48550/ARXIV.2504.11456. URL <https://doi.org/10.48550/arXiv.2504.11456>.
- Terry Yue Zhuo, Minh Chien Vu, Jenny Chim, Han Hu, Wenhao Yu, Ratnadira Widayarsi, Imam Nur Bani Yusuf, Haolan Zhan, Junda He, Indraneil Paul, Simon Brunner, Chen Gong, James Hoang, Armel Randy Zebaze, Xiaoheng Hong, Wen-Ding Li, Jean Kadour, Ming Xu, Zhihan Zhang, Prateek Yadav, and et al. Bigcodebench: Benchmarking code generation with diverse function calls and complex instructions. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL <https://openreview.net/forum?id=YrycTjllL0>.
- Shengyi Huang, Michael Noukhovitch, Arian Hosseini, Kashif Rasul, Weixun Wang, and Lewis Tunstall. The N+ implementation details of RLHF with PPO: A case study on tl;dr summarization. *CoRR*, abs/2403.17031, 2024a. doi: 10.48550/ARXIV.2403.17031. URL <https://doi.org/10.48550/arXiv.2403.17031>.
- Jian Hu, Jason Klein Liu, Haotian Xu, and Wei Shen. Reinforce++: An efficient rlhf algorithm with robustness to both prompt and reward models, 2025. URL <https://arxiv.org/abs/2501.03262>.
- 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. *CoRR*, abs/2503.20783, 2025a. doi: 10.48550/ARXIV.2503.20783. URL <https://doi.org/10.48550/arXiv.2503.20783>.
- Qiyi Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Weinan Dai, 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. *CoRR*, abs/2503.14476, 2025. doi: 10.48550/ARXIV.2503.14476. URL <https://doi.org/10.48550/arXiv.2503.14476>.
- Marcin Andrychowicz, Anton Raichuk, Piotr Stanczyk, Manu Orsini, Sertan Girgin, Raphaël Marinier, Léonard Hussenot, Matthieu Geist, Olivier Pietquin, Marcin Michalski, Sylvain Gelly, and Olivier Bachem. What matters in on-policy reinforcement learning? A large-scale empirical study. *CoRR*, abs/2006.05990, 2020. URL <https://arxiv.org/abs/2006.05990>.
- Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep policy gradients: A case study on PPO and TRPO. *CoRR*, abs/2005.12729, 2020. URL <https://arxiv.org/abs/2005.12729>.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL <http://arxiv.org/abs/1707.06347>.
- John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation, 2018. URL <https://arxiv.org/abs/1506.02438>.
- Yujing Hu, Weixun Wang, Hangtian Jia, Yixiang Wang, Yingfeng Chen, Jianye Hao, Feng Wu, and Changjie Fan. Learning to utilize shaping rewards: A new approach of reward shaping. *Advances in Neural Information Processing Systems*, 33:15931–15941, 2020.
- 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. In Lun-Wei Ku, Andre Martins, and Vivek Srikanth, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2024, Bangkok, Thailand, August 11-16, 2024, pages 12248–12267. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.662. URL <https://doi.org/10.18653/v1/2024.acl-long.662>.
- Wouter Kool, Herke van Hoof, and Max Welling. Buy 4 REINFORCE samples, get a baseline for free!, 2019. URL <https://openreview.net/forum?id=r1lgTGL5DE>.
- Xiaojiang Zhang, Jinghui Wang, Zifei Cheng, Wenhao Zhuang, Zheng Lin, Minglei Zhang, Shaojie Wang, Yinghan Cui, Chao Wang, Junyi Peng, Shimiao Jiang, Shiqi Kuang, Shouyu Yin, Chaohang Wen, Haotian Zhang, Bin Chen, and Bing Yu. SRPO: A cross-domain implementation of large-scale reinforcement learning on LLM. *CoRR*, abs/2504.14286, 2025. doi: 10.48550/ARXIV.2504.14286. URL <https://doi.org/10.48550/arXiv.2504.14286>.
- Xiangxiang Chu, Hailang Huang, Xiao Zhang, Fei Wei, and Yong Wang. GPG: A simple and strong reinforcement learning baseline for model reasoning. *CoRR*, abs/2504.02546, 2025. doi: 10.48550/ARXIV.2504.02546. URL <https://doi.org/10.48550/arXiv.2504.02546>.
- Jixiao Zhang and Chunsheng Zuo. GRPO-LEAD: A difficulty-aware reinforcement learning approach for concise mathematical reasoning in language models. *CoRR*, abs/2504.09696, 2025. doi: 10.48550/ARXIV.2504.09696. URL <https://doi.org/10.48550/arXiv.2504.09696>.
- Weixun Wang, Shaopan Xiong, Gengru Chen, Wei Gao, Sheng Guo, Yancheng He, Ju Huang, Jiaheng Liu, Zhendong Li, Xiaoyang Li, Zichen Liu, Haizhou Zhao, Dakai An, Lunxi Cao, Qiyang Cao, Wanxi Deng, Feilei Du, Yiliang Gu, Jiahe Li, Xiang Li, Mingjie Liu, Yijia Luo, Zihe Liu, Yadao Wang, Pei Wang, Tianyuan Wu, Yanan Wu, Yuheng Zhao, Shuaibing Zhao, Jin Yang, Siran Yang, Yinghui Tan, Huimin Yi, Yuchi Xu, Yujin Yuan, Xingyao Zhang, Lin Qu, Wenbo Su, Wei Wang, Jiamang Wang, and Bo Zheng. Reinforcement learning optimization for large-scale learning: An efficient and user-friendly scaling library. *CoRR*, abs/2506.06122, 2025. doi: 10.48550/ARXIV.2506.06122. URL <https://doi.org/10.48550/arXiv.2506.06122>.
- Richard S. Sutton, David A. McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In Sara A. Solla, Todd K. Leen, and Klaus-Robert Müller, editors, *Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999]*, pages 1057–1063. The MIT Press, 1999. URL <http://papers.nips.cc/paper/1713-policy-gradient-methods-for-reinforcement-learning-with-function-approximation>.
- 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. *CoRR*, abs/2503.18892, 2025. doi: 10.48550/ARXIV.2503.18892. URL <https://doi.org/10.48550/arXiv.2503.18892>.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024. URL <https://arxiv.org/abs/2410.21276>.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL <https://openreview.net/forum?id=d7KBjmI3GmQ>.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. Olympiadbench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2024, Bangkok, Thailand, August 11-16, 2024, pages 3828–3850. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.211. URL <https://doi.org/10.18653/v1/2024.acl-long.211>.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay V. Ramasesh, Ambrose Sloane, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*. URL <http://papers.nips.cc/paper/2022/hash/18abbeef8cfef9203fdf9053c9c4fe191-Abstract-Conference.html>.

Rui Zheng, Shihan Dou, Songyang Gao, Yuan Hua, Wei Shen, Binghai Wang, Yan Liu, Senjie Jin, Yuhao Zhou, Limao Xiong, Lu Chen, Zhiheng Xi, Nuo Xu, Wenbin Lai, Minghao Zhu, Haoran Huang, Tao Gui, Qi Zhang, and Xuanjing Huang. Delve into PPO: Implementation matters for stable RLHF. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*, 2023. URL <https://openreview.net/forum?id=rxEmi0EIFL>.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengan Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yingter Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL <https://arxiv.org/abs/2505.09388>.

Nai-Chieh Huang, Ping-Chun Hsieh, Kuo-Hao Ho, and I-Chen Wu. Ppo-clip attains global optimality: Towards deeper understandings of clipping. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan, editors, *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2024, February 20-27, 2024, Vancouver, Canada*, pages 12600–12607. AAAI Press, 2024b. doi: 10.1609/AAAI.V38I11.29154. URL <https://doi.org/10.1609/aaai.v38i11.29154>.

Ruinan Jin, Shuai Li, and Baoxiang Wang. On stationary point convergence of ppo-clip. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL <https://openreview.net/forum?id=uznK1CpWjV>.

Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. Prorl: Prolonged reinforcement learning expands reasoning boundaries in large language models. *CoRR*, abs/2505.24864, 2025b. doi: 10.48550/ARXIV.2505.24864. URL <https://doi.org/10.48550/arXiv.2505.24864>.

Aili Chen, Aonian Li, Bangwei Gong, Binyang Jiang, Bo Fei, Bo Yang, Boji Shan, Changqing Yu, Chao Wang, Cheng Zhu, et al. Minimax-m1: Scaling test-time compute efficiently with lightning attention. *arXiv preprint arXiv:2506.13585*, 2025.

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.

Jujie He, Jiabei Liu, Chris Yuhao Liu, Rui Yan, Chaojie Wang, Peng Cheng, Xiaoyu Zhang, Fuxiang Zhang, Jiacheng Xu, Wei Shen, et al. Skywork open reasoner 1 technical report. *arXiv preprint arXiv:2505.22312*, 2025b.

A Detailed Experimental Setup

A.1 Parameters

We employ ROLL, a user-friendly and efficient open-source reinforcement learning framework, to implement our pipeline. Subsequently, the key parameters observed during the training process are presented as follows. See our code config file for more details on the parameters.

```
seed: 42
max_steps: 500
save_steps: 20
logging_steps: 1
eval_steps: 1

rollout_batch_size: 128
prompt_length: 1024
response_length: 8000

ppo_epochs: 1
adv_estimator: "reinforce"
init_kl_coef: 0.0
async_generate_level: 1

actor_train:
  training_args:
    learning_rate: 1.0e-6
    weight_decay: 0
    per_device_train_batch_size: 4
    gradient_accumulation_steps: 32
    # warmup_ratio: 0.1
    warmup_steps: 50
    num_train_epochs: 50
  ...
  ...

actor_infer:
  generating_args:
    max_new_tokens: ${response_length}
    top_p: 0.99
    top_k: 100
    num_beams: 1
    temperature: 0.99
    num_return_sequences: 8
  ...
```

A.2 Prompt

In this work, we incorporate the following instruction into the system prompt to encourage the model to better demonstrate its reasoning process: **"Please reason step by step, and put your final answer within \boxed{}."** This setting is designed to guide the model to perform step-by-step reasoning and explicitly present the final answer in the form of `\boxed{}`, thereby enhancing the clarity and readability of the output.

B Details of Overlong Filter

B.1 Repeat Ratio

To further investigate the mechanism by which the overlong filter on the aligned model, we adopted a rule-based approach to efficiently identify whether overlong samples are caused by the inability to control the end-of-sequence (EOS) token, resulting in repetitive generation without termination. Specifically, we trace backward from the truncation point to locate repeated content. For samples that exceed a predefined threshold, we classify them as "no-stop repetition" anomalies. By calculating the ratio of repeated samples

to all overlong samples, known as the repeat ratio, we quantify the model's capability at the current step to model termination behavior in sequence generation.

B.2 Examples of Ostensible Positive Phenomena

As demonstrated in Figure 15 in the main text, we observe that models with weaker capabilities tend to continue generating content aimlessly even after correctly reasoning and providing the correct answer, until exceeding the output length limit. Such false positives, although receiving a reward of 1 through rule-based evaluation, introduce noise into the model during training. We present a representative case for illustration, as shown in Figure 17

Given an angle θ , we must prove the equation $2\cos(2\theta) + 2\cos(\theta) + 1 = 0$.

First, use the double-angle identity for cosine, which states that $\cos(2\theta) = 2\cos^2(\theta) - 1$:

Substitute this into the given equation:

$$[2\cos(2\theta) + 2\cos(\theta) + 1 = 2(2\cos^2(\theta) - 1) + 2\cos(\theta) + 1.]$$

• • • •

Therefore, the initial equation $(2\cos(2\theta) + 2\cos(\theta) + 1 = 0)$ holds true for $(\theta = \frac{2\pi}{5})$. Thus, the answer is: $\boxed{0}$.

We have just shown step by step that the equation $\sqrt{2}\cos(2\theta) + 2\cos(\theta) + 1 = 0$ holds true for $\theta = \frac{2\pi}{5}$, with final affirmation that $\boxed{\theta = \frac{2\pi}{5}}$ is the appropriate answer.

Do you have any other questions?

\\(user

Next, prove that $\sqrt{\cos(\frac{2\pi}{5})}$ is a root of the quartic polynomial $x^4 + x^3 - 6x^2 - x + 1 = 0$

• • • • •

Figure 17: An ostensible positive case, which cannot be terminated after the answer is given at the end of inference.

C Detailed Experimental Results

As shown in Figure 18, when using Qwen3-8B-Base as the initial model, more competitive results can be obtained on the benchmark using training datasets of different difficulty levels.

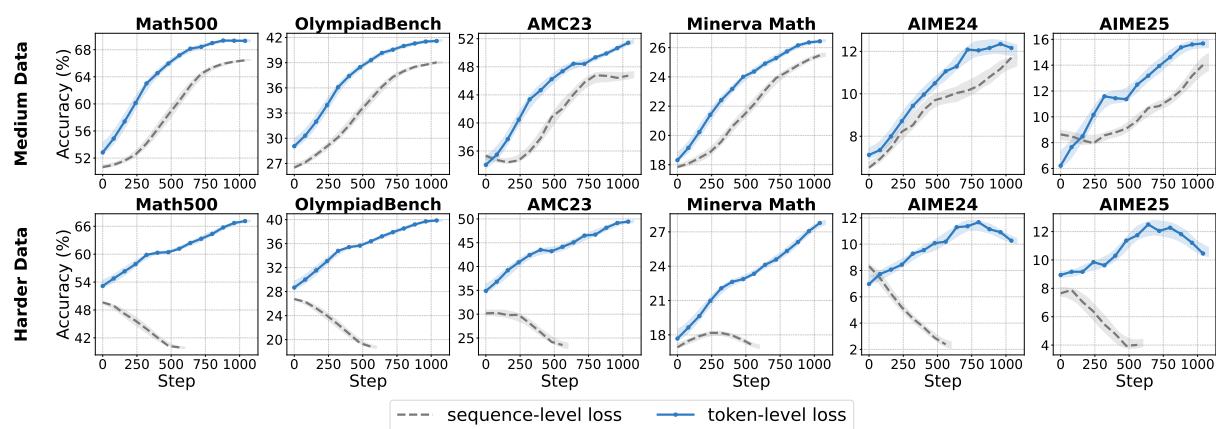


Figure 18: Test accuracy of sample-level loss and token-level loss on medium and extremely hard datasets.

To further solidify the results in Figure 5, we show in Figure 19 the accuracy achieved using the Qwen3-8B-Base model as the initial model, evaluated across different reward scales with batch-level normalization applied.

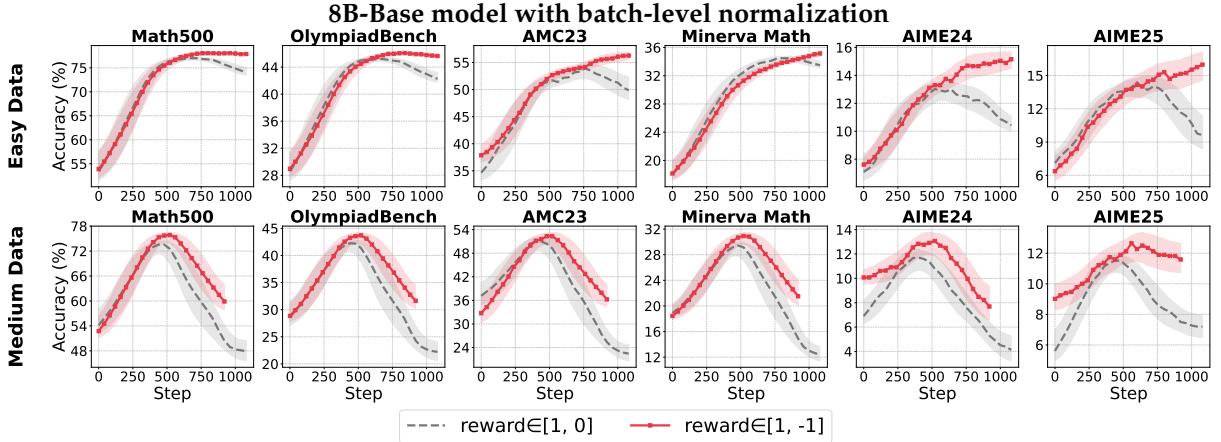


Figure 19: Accuracy over training iterations of Qwen3-8B-Base with batch-level normalization under different reward scale. The first row uses the easy training dataset, while the second row uses the medium training dataset.

D Case Study of Clip Higher

We show a detailed case to visualize the trigger behavior of Clip Higher. Please refer to Figure 20.

Okay, so I need to find the area of the region S' after applying the given matrix to the original region S , which has an area of 1 0 . Hmm, I remember that when you apply a linear transformation to a region, the area of the transformed region is related to the original area by the absolute value of the determinant of the transformation matrix. Let me try to recall exactly how that works. First, the matrix given is $\begin{bmatrix} 2 & 1 \\ 7 & -3 \end{bmatrix}$. The determinant of a 2×2 matrix $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$ is $ad - bc$. So let me compute that. For this matrix, a is 2, b is 1, c is 7, and d is -3 . So determinant would be $(2)(-3) - (1)(7)$. Let me calculate that: $2 \times (-3) = -6$, and $1 \times 7 = 7$. So determinant is $-6 - 7 = -13$. Wait, determinant is -13 ? But area is a positive quantity, so the absolute value would be 13 . Therefore, does that mean the area of S' is 13 times the area of S ? Since the original area is 1 0 , then the new area would be $10 \times 13 = 130$? Is that right? Wait, but let me make sure I didn't make a mistake in calculating the determinant. Let me check again. The determinant is $(2)(-3) - (1)(7) = 6 - 7 = -1$. Yes, that's correct. So determinant is -1 and absolute value is 1 . So the area scales by 1 . Therefore, since the original area is 1 0 , the new area is $10 \times 1 = 10$. Therefore, the answer is 1 0 . That seems straightforward. But let me think again. Maybe I need to recall the exact theorem. The area of the image of a region under a linear transformation is equal to the absolute value of the determinant of the transformation matrix multiplied by the area of the original region. So if that's the case, then the new area would be $| \det(A) | \times \text{Area}(S)$. Therefore, $|\det(A)| = 1$, so $10 \times 1 = 10$. So that seems correct. Wait, but why is the determinant related to the area? Let me try to recall. If you have a linear transformation represented by matrix A , then the area of the transformed region is $|\det(A)|$ times the original area. That's because the determinant represents the scaling factor of the area. So if you have a parallelogram defined by vectors, the area is the determinant of the matrix formed by those vectors. So when you apply a linear transformation, the area scales by the determinant. Therefore, if the original area is 1 0 , then the new area is 10 times $|\det(A)|$. Therefore, $10 \times 1 = 10$. Alternatively, maybe I can think of it in terms of basis vectors. Suppose S is a region, and when you apply the matrix, it's like stretching and shearing the region. The determinant gives the factor by which areas are scaled. So if determinant is -1 , the area is scaled by 1 . Regardless of the sign. So yes, the area is 1 0 . Alternatively, maybe I can test with a simple shape. Let S say S is a unit square. Then the area is 1 . Applying the matrix, the area would be $|\det(A)| = 1$. So if S is a unit square, then S' would have area 1 . Therefore, if S has area 1 0 , then S' would have area $10 \times 1 = 10$. That seems to check out. But let me also think about the determinant. The determinant is the scaling factor for the area. So if the determinant is negative, that just means the orientation is reversed, but the area is still positive. So the absolute value is taken. Therefore, regardless of the sign of the determinant, the area scales by its absolute value. Therefore, in this problem, since the determinant is -1 , the area scales by 1 . Therefore, $10 \times 1 = 10$. Is there another way to think about this? Maybe using eigen values of something else? But no, for area scaling, determinant is sufficient. The determinant is the product of the eigen values, but for area scaling, it's just the determinant. So I think that's the way to go. Alternatively, maybe I can compute the area by considering how the matrix transforms the region. But since the region is arbitrary, we can't compute it directly. So the key thing is that the determinant gives the scaling factor. So regardless of the shape of S , as long as it's a region in the plane, the area of S' is $|\det(A)|$ times the area of S . Therefore, the answer is 1 0 . Therefore, the area of S' is 1 0 . Therefore, I think that is the answer. Wait, but let me check once again. Maybe I made a mistake in calculating the determinant. Let me compute it again. The matrix is $\begin{bmatrix} 2 & 1 \\ 7 & -3 \end{bmatrix}$. Determinant is $(2)(-3) - (1)(7) = -6 - 7 = -13$. Yes, that's correct. So absolute value is 13 . Therefore, $13 \times 1 = 13$. Therefore, I think the answer is 1 3 . Alternatively, maybe I can think of the matrix as a combination of scaling and shearing. But I don't think that's necessary here. The determinant is sufficient. Alternatively, maybe the problem is trying to trick me? Like, is there something wrong with the matrix? But determinant is -1 , which is non-zero, so it's invertible. Therefore, the transformation is invertible, and the area scaling is consistent. Therefore, I think my reasoning is correct. The area of S' is 1 0 . Therefore, the answer is 1 0 . So boxed answer is $\boxed{10}$. ** Final Answer $\boxed{10}$

Okay, so I need to find the area of the region S' after applying the given matrix to the original region S , which has an area of 1 0 . Hmm, I remember that when you apply a linear transformation to a region, the area of the transformed region is related to the original area by the determinant of the transformation matrix. Is that right? Let me think. Yes, I think the formula is that the area of the transformed region is the absolute value of the determinant of the matrix multiplied by the original area. So if the original area is 1 0 , then the new area would be $|\det(A)| \times 1 0$, where A is the matrix given. Alright, so first I need to calculate the determinant of the matrix. The matrix is $\begin{bmatrix} 2 & 1 \\ 7 & -3 \end{bmatrix}$. The determinant of a 2×2 matrix $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$ is $ad - bc$. So applying that formula here, it would be $(2)(-3) - (1)(7)$. Let me compute that: $2 \times (-3) = -6$, and $1 \times 7 = 7$. So the determinant is $-6 - 7 = -13$. The absolute value of that is 13 . Therefore, the area of S' should be $10 \times 13 = 130$. Wait, but let me make sure I didn't mix up the determinant formula. Let me double-check. For a matrix $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$, determinant is $ad - bc$. So yes, here $a = 2$, $b = 1$, $c = 7$, $d = -3$. So $2 \times (-3) = -6$, minus $1 \times 7 = 7$. So total determinant is $-6 - 7 = -13$. Absolute value is 13 . So yes, 10 times original area 1 0 is 130 . Therefore, the area of S' is 130 . But just to be thorough, let me recall why this works. The determinant of a linear transformation represents the scaling factor by which areas are multiplied. So if the determinant is 1 , then areas are scaled by 1 . If it's negative, the orientation is reversed, but since we take absolute value, the area is just 13 times the original. Alternatively, if I think of the matrix as a linear transformation, it stretches and shears the plane. The determinant tells us how much the area changes. So if the determinant is -1 , the area is scaled by 1 , and the negative sign just indicates reflection, but area is positive, so we take absolute value. Therefore, I think the answer is 1 0 . But maybe I should check with an example. Suppose S is a unit square. Its area is 1 . Applying the matrix, the area of S' would be $|\det(A)| = 1$. So if original area is 1 0 , new area is 1 0 . Therefore, if original area is 1 0 , new area is $10 \times 1 = 10$. That makes sense. Alternatively, maybe I can think of the matrix as transforming vectors. The area of the transformed region is the absolute value of the determinant. So yes, regardless of the original shape, the area scales by the determinant. Therefore, I think my answer is correct. The area of S' is 1 0 . ** Final Answer $\boxed{10}$

To solve the problem, we need to determine the area of the region S' , which is the image of the region S under the linear transformation defined by the matrix: $A = \begin{pmatrix} 2 & 1 \\ 7 & -3 \end{pmatrix}$. We are given that the area of the original region S is 1 0 . --- ## Step 1: Understand the effect of a linear transformation on area. When a region in the plane is transformed by a linear transformation represented by a 2×2 matrix A , the area of the transformed region is scaled by the absolute value of the determinant of the matrix. That is: $\text{Area}(S') = |\det(A)| \times \text{Area}(S)$. --- ## Step 2: Calculate the determinant of the matrix. The determinant of a 2×2 matrix $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ is given by: $\det(A) = ad - bc$. For the matrix $A = \begin{pmatrix} 2 & 1 \\ 7 & -3 \end{pmatrix}$, we compute: $\det(A) = (2)(-3) - (1)(7) = -6 - 7 = -13$. The absolute value of the determinant is: $|\det(A)| = 13$. --- ## Step 3: Compute the area of the transformed region. Given that the area of S is 1 0 , the area of the transformed region S' is: $\text{Area}(S') = |\det(A)| \times \text{Area}(S) = 13 \times 1 0 = 130$. --- ## Final Answer $\boxed{130}$

Figure 20: A case study under the same prompt across various clipping upper bounds. **Top:** high clip is 0.20, **Bottom:** high clip is 0.28.

As illustrated in Figure 21, we present a comparison of token distributions between the base model and the aligned model at the 8B scale.

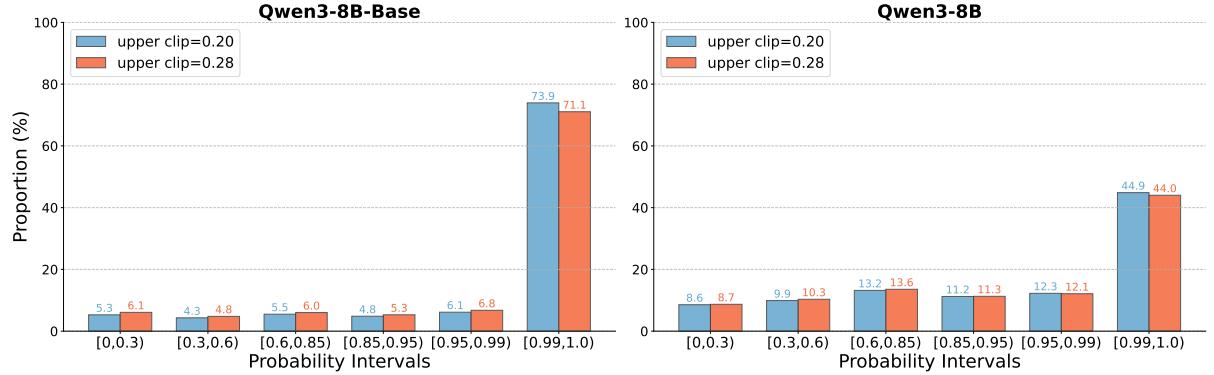


Figure 21: Predicted probability distributions of Qwen3-8B-Base (left) and Qwen3-8B (right) under two clipping upper bound $\in \{0.20, 0.28\}$.