2.4 Rule-based Reward Modeling

The use of reward model usually suffers from the reward hacking problem [24–29]. Instead, we directly use the final accuracy of a verifiable task as the outcome reward, computed using the following rule:

$$R(\hat{y}, y) = \begin{cases} 1, & \texttt{is\_equivalent}(\hat{y}, y) \\ -1, & \text{otherwise} \end{cases} \tag{7}$$

where y is the ground-truth answer and $\hat{y}$ is the predicted answer. This is proved to be an effective approach to activating the base model's reasoning capability, as shown in multiple domains such as automated theorem proving [30–33], computer programming [34–37], and mathematics competition [2].

3 DAPO

We propose the **D**ecouple Clip and **D**ynamic s**A**mpling **P**olicy **O**ptimization (DAPO) algorithm. DAPO samples a group of outputs $\{o_i\}_{i=1}^{G}$ for each question q paired with the answer a, and optimizes the policy via the following objective:

$$\begin{aligned} \mathcal{J}_{\text{DAPO}}(\theta) = \; & \mathbb{E}_{(q,a)\sim\mathcal{D},\{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|q)} \\ & \left[\frac{1}{\sum_{i=1}^{G}|o_i|}\sum_{i=1}^{G}\sum_{t=1}^{|o_i|}\min\Big(r_{i,t}(\theta)\hat{A}_{i,t},\ \text{clip}\Big(r_{i,t}(\theta), 1-\varepsilon_{\text{low}}, 1+\varepsilon_{\text{high}}\Big)\hat{A}_{i,t}\Big)\right] \\ \text{s.t.} \quad & 0 < \Big|\{o_i \mid \texttt{is\_equivalent}(a, o_i)\}\Big| < G, \end{aligned} \tag{8}$$

where

$$r_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})}, \quad \hat{A}_{i,t} = \frac{R_i - \text{mean}(\{R_i\}_{i=1}^{G})}{\text{std}(\{R_i\}_{i=1}^{G})}. \tag{9}$$

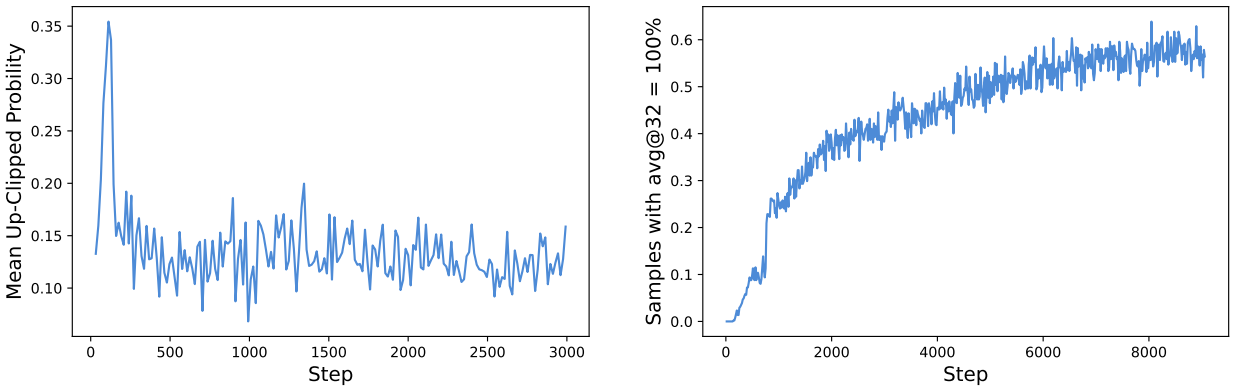
The full algorithm can be found in Algorithm 1. In this section, we will introduce the key techniques associated with DAPO.

3.1 Raise the Ceiling: Clip-Higher

In our initial experiments using naive PPO [21] or GRPO [38], we observed the entropy collapse phenomenon: the entropy of the policy decreases quickly as training progresses (Figure 2b). The sampled responses of certain groups tend to be nearly identical. This indicates limited exploration and early deterministic policy, which can hinder the scaling process.

We propose the **Clip-Higher** strategy to address this issue. Clipping over the importance sampling ratio is introduced in Clipped Proximal Policy Optimization (PPO-Clip) [21] to restrict the trust region and enhance the stability of RL. We identify that the upper clip can restrict the exploration of the policy, where making an 'exploitation' token more probable is much easier yet the probability of an unlikely 'exploration' token is too tightly bounded to be uplifted.

Concretely, when $\varepsilon = 0.2$ (the default value of most algorithms) and $\hat{A}_{i,t} > 0$ (the system tries to increase the probability), consider two actions with probabilities $\pi_{\theta_{\text{old}}}(o_i \mid q) = 0.01$ and 0.9. The upper bounds of the increased probabilities $\pi_\theta(o_i \mid q)$ are 0.012 and 1.08, respectively ($\pi_{\theta_{\text{old}}} \cdot (1+\epsilon)$). This implies that 'exploitation' tokens with a higher probability (*e.g.*, 0.9) are not constrained to get even extremely larger probabilities like 0.999. Conversely, for low-probability 'exploration' tokens, achieving a non-trivial increase in probability is considerably more challenging. Empirically, we also observe that the mean probability of up-clipped tokens is low: $\pi_\theta(o_i \mid q) < 0.2$ (Figure 3a). This finding supports our intuition that the upper clipping threshold indeed restricts the probability increase of low-probability 'exploration' tokens, thereby potentially constraining the exploration of the system.



(a) Mean up-clipped probability.

(b) The proportion of samples with an accuracy of 1.

Figure 3 The mean up-clipped probability as well as the ratio of prompts with accuracy=1.

Adhering to the **Clip-Higher** strategy, we decouple the lower and higher clipping range as ε_{low} and $\varepsilon_{\text{high}}$, as highlighted in Equation 10:

$$\begin{aligned} \mathcal{J}_{\text{DAPO}}(\theta) = \; & \mathbb{E}_{(q,a)\sim\mathcal{D},\{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|q)} \\ & \left[\frac{1}{\sum_{i=1}^{G}|o_i|}\sum_{i=1}^{G}\sum_{t=1}^{|o_i|}\min\Big(r_{i,t}(\theta)\hat{A}_{i,t},\; \text{clip}\Big(r_{i,t}(\theta), 1-\varepsilon_{\text{low}}, 1+\varepsilon_{\text{high}}\Big)\hat{A}_{i,t}\Big)\right] \\ \text{s.t.} \quad & 0 < \Big|\{o_i \mid \texttt{is\_equivalent}(a, o_i)\}\Big| < G. \end{aligned} \tag{10}$$

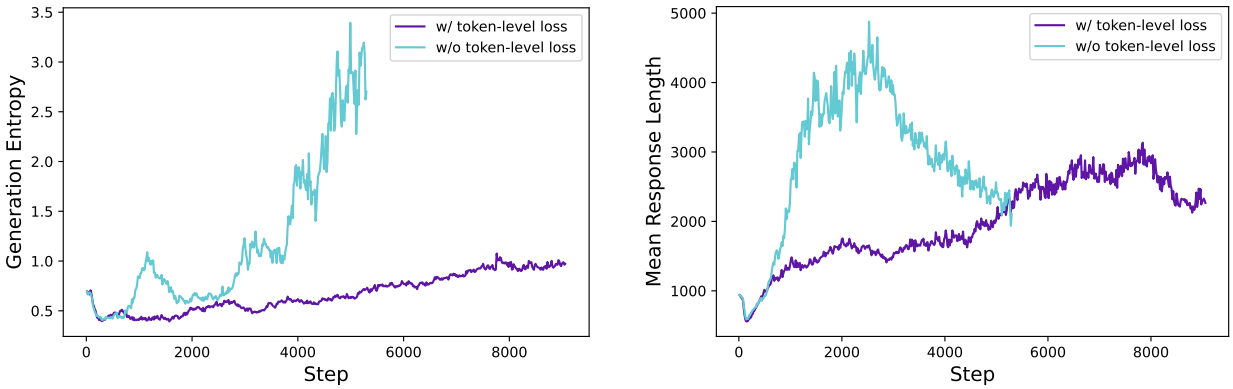
We increase the value of $\varepsilon_{\text{high}}$ to leave more room for the increase of low-probability tokens. As shown in Figure 2, this adjustment effectively enhances the policy's entropy and facilitates the generation of more diverse samples. We keep ε_{low} as it is, because increasing it will suppress the probability of these tokens to 0, resulting in the collapse of the sampling space.

3.2 The More the Merrier: Dynamic Sampling

Existing RL algorithm suffers from the gradient-decreasing problem when some prompts have accuracy equal to 1. For example for GRPO, if all outputs $\{o_i\}_{i=1}^{G}$ of a particular prompt are correct and receive the same reward, the resulting advantage for this group is *zero*. A zero advantage results in zero policy gradients, shrinking the magnitude and increasing the noise sensitivity of the batch gradient, thereby degrading sample efficiency. Empirically, the number of samples with accuracy equal to 1 continues to increase, as shown in Figure 3b. This means that the effective number of prompts in each batch keeps decreasing, which can lead to larger variance in gradient and dampens the gradient signals for model training.

To this end, we propose to **over-sample and filter out prompts with the accuracy equal to 1 and 0** as illustrated in Equation 11, leaving all prompts in the batch with effective gradients and keeping a consistent number of prompts. The sampling cost for each batch is dynamic. Before training, we keep sampling until the batch is fully filled with samples whose accuracy is neither 0 nor 1.

$$\begin{aligned} \mathcal{J}_{\text{DAPO}}(\theta) = \; & \mathbb{E}_{(q,a)\sim\mathcal{D},\{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|q)} \\ & \left[\frac{1}{\sum_{i=1}^{G}|o_i|}\sum_{i=1}^{G}\sum_{t=1}^{|o_i|}\min\Big(r_{i,t}(\theta)\hat{A}_{i,t},\; \text{clip}\Big(r_{i,t}(\theta), 1-\varepsilon_{\text{low}}, 1+\varepsilon_{\text{high}}\Big)\hat{A}_{i,t}\Big)\right] \\ \text{s.t.} \quad & 0 < \Big|\{o_i \mid \texttt{is\_equivalent}(a, o_i)\}\Big| < G. \end{aligned} \tag{11}$$



(a) Entropy of actor model's generation probabilities. **(b)** Average length of actor model-generated responses

Figure 4 The entropy of the probability distribution of the actor model, as well as the changes in response length.

Note that this strategy does not necessarily impede training efficiency, because the generation time is typically dominated by the generation of long-tail samples if the RL system is synchronized and the generation stage is not pipelined. Besides, we find that with dynamic sampling the experiment achieves the same performance faster as shown in Figure 6.

3.3 Rebalancing Act: Token-Level Policy Gradient Loss

The original GRPO algorithm employs a sample-level loss calculation, which involves first averaging the losses by token within each sample and then aggregating the losses across samples. In this approach, each sample is assigned an equal weight in the final loss computation. However, we find that this method of loss reduction introduces several challenges in the context of long-CoT RL scenarios.

Since all samples are assigned the same weight in the loss calculation, tokens within longer responses (which contain more tokens) may have a disproportionately lower contribution to the overall loss, which can lead to two adverse effects. First, for high-quality long samples, this effect can impede the model's ability to learn reasoning-relevant patterns within them. Second, we observe that excessively long samples often exhibit low-quality patterns such as gibberish and repetitive words. Thus, sample-level loss calculation, due to its inability to effectively penalize those undesirable patterns in long samples, leads to an unhealthy increase in entropy and response length, as shown in Figure 4a and Figure 4b.

We introduce a **Token-level Policy Gradient Loss** in the long-CoT RL scenario to address the above limitations:

$$\begin{aligned} \mathcal{J}_{\text{DAPO}}(\theta) = \quad & \mathbb{E}_{(q,a)\sim\mathcal{D},\{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|q)} \\ & \left[\frac{1}{\sum_{i=1}^{G}|o_i|}\sum_{i=1}^{G}\sum_{t=1}^{|o_i|}\min\left(r_{i,t}(\theta)\hat{A}_{i,t},\ \text{clip}\Big(r_{i,t}(\theta),1-\varepsilon_{\text{low}},1+\varepsilon_{\text{high}}\Big)\hat{A}_{i,t}\right)\right], \\ \text{s.t.} \quad & 0 < \Big|\{o_i \mid \texttt{is\_equivalent}(a,o_i)\}\Big| < G. \end{aligned} \tag{12}$$

In this setting, longer sequences can have more influence on the overall gradient update compared to shorter sequences. Moreover, from the perspective of individual tokens, if a particular generation pattern can lead to an increase or decrease in reward, it will be equally prompted or suppressed, regardless of the length of the response in which it appears.

3.4 Hide and Seek: Overlong Reward Shaping

In RL training, we typically set a maximum length for generation, with overlong samples truncated accordingly. We find that improper reward shaping for truncated samples can introduce reward noise and significantly disrupt the training process.