Single-Rollout Asynchronous Optimization for Agentic Reinforcement Learning

Zhenyu Hou* **Yujiang Li*** **Jie Tang** **Yuxiao Dong**

Tsinghua University

Abstract

Reinforcement learning (RL) is becoming increasingly important for post-training large language models (LLMs). Previous RL pipelines for LLMs were mostly synchronous and batch-interleaved, which is inefficient for long-horizon agentic tasks. Recently, asynchronous RL has emerged as a more efficient alternative by updating the model as rollouts arrive. However, existing asynchronous RL systems often emphasize throughput, while leaving training stability and task effectiveness largely underexplored. For example, a key challenge is that group-wise sampling in the widely-used GRPO framework does not naturally fit asynchronous agentic training. In this paper, we present Single-rollout Asynchronous Optimization (SAO) to address the stability and off-policy challenges in asynchronous RL. To reduce off-policy effects and improve generalization, we replace group-wise sampling with single-rollout sampling, that is, using one rollout per prompt. We further improve this single-rollout strategy with practical value-model training designs. To improve optimization stability, we introduce a strict double-side token-level clipping strategy. SAO is able to train stably for one thousand steps and consistently outperform GRPO and its variants on agentic coding and reasoning benchmarks, such as SWE-Bench Verified, BeyondAIME, and IMOAnswerBench. We also demonstrate that single-rollout RL is particularly effective in a simulated online learning setting, where the model must adapt to changing evolving environments. To this end, SAO is successfully deployed in the agentic RL pipeline for training the open GLM-5.2 model (750B-A40B).

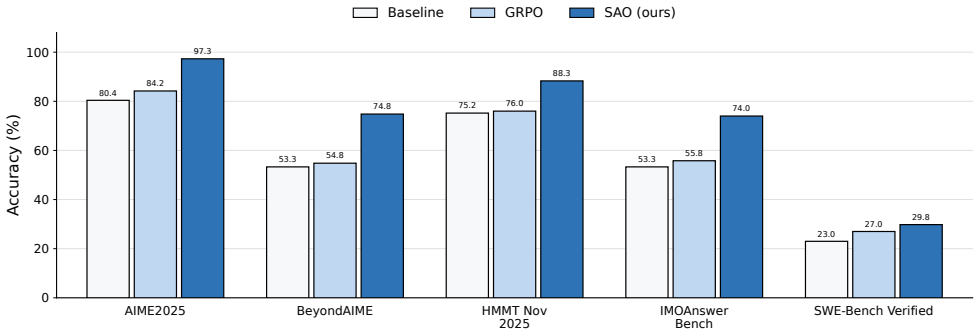


Figure 1: **The performance of SAO on reasoning and coding benchmarks.** The four reasoning benchmarks are evaluated in a reasoning-with-Python-tool setting, where the baseline is the Qwen3-30B-A3B SFT model; SWE-Bench Verified evaluates coding with the Qwen3-30B-A3B baseline. SAO outperforms the corresponding baseline and GRPO across all five benchmarks.

*Equal Contribution. Work done while ZH and YL interned at Z.AI.

Under review, Feb 2026.

1 Introduction

Large Language Model (LLM) development is shifting from supervised pre-training toward post-training reinforcement learning (RL). Recent work in Reinforcement Learning has demonstrated that scaling RL compute together with test-time compute is a highly effective way to improve model intelligence [DeepSeek-AI, 2024a, OpenAI, 2024, Cobbe et al., 2021, Lightman et al., 2023]. Most LLM RL pipelines remain synchronous and interleaved: the policy generates a batch of rollouts, and optimization starts only after the entire batch is collected [Ouyang et al., 2022, Rafailov et al., 2024].

For agentic and coding workloads, rollout lengths are highly variable, so short trajectories finish quickly while long ones become stragglers; as a result, large portions of the GPU cluster idle while waiting for the slowest rollouts [DeepSeek-AI, 2024b, Kwon et al., 2023, Yu et al., 2022]. Asynchronous RL mitigates this *imbalanced generation overhead* by consuming rollouts continuously as they arrive, improving utilization and wall-clock efficiency [Mnih et al., 2016, Liang et al., 2018, Hoffman et al., 2020].

However, asynchrony introduces two challenges. First, each trajectory can be generated by multiple versions of the old rollout model, which leads to more unpredictable and severe off-policy, and thus harms the training stability. Previous works [Fu et al., 2025, Noukhovitch et al., 2024] make attempts for asynchronous RL but mainly focus on efficiency optimization rather than effectiveness. Second, group-wise methods such as GRPO [Shao et al., 2024, Wang et al., 2022] are mismatched to asynchronous training. GRPO samples a group of responses for each prompt and uses the group-level average for advantage estimation. The group-wise sampling induces latency-driven off-policy behavior because the group has to wait for the slower one to finish before fed into training. In addition, group-wise sampling is incompatible with online or complex agentic settings where the environment often provides only a single trajectory feedback per prompt [Sutton and Barto, 2018, Schulman et al., 2017, Yao et al., 2022, Nakano et al., 2021].

In this paper, we propose Single-rollout Asynchronous Optimization (SAO) for agentic RL. It keeps asynchronous RL training stable and effective under policy lag while preserving the efficiency of asynchrony. Instead of group-wise sampling, such as GRPO, SAO uses single-rollout updates. To make this setting practical, it also introduces effective value-model training strategies. Our contributions are as follows:

- To stabilize training under varied policy lag, we use token-level importance sampling strategy. It directly uses the log-probabilities from the rollout engine and applies stricter double-sided token-level clipping and masking.
- To reduce off-policy effects, we use one single rollout sampling for each prompt instead of group-wise sampling previously populated by GRPO. To further make this setting practical in agentic RL, we improve the value model process. Specifically, we update the critic more frequent than the actor and fine-tune the value model with frozen attention.
- To handle multi-turn agent trajectories with interleaved environment feedback, we derive a skip-observation token-level GAE estimator. It computes advantages across action-to-action boundaries. It also avoids propagating noise through observation tokens that are not generated by the model.

We evaluate SAO on agentic coding and math reasoning benchmarks, including SWE-Bench Verified [Jimenez et al., 2023], AIME2025 [Balunović et al., 2025], BeyondAIME [ByteDanceSeed, 2025], HMMT[Balunović et al., 2025], and IMOAnswerBench[Luong et al., 2025]. The results demonstrate that our asynchronous RL design can stably train for around one thousand steps and achieves consistently better performance than improved GRPO. In addition, we show that the single-rollout strategy in SAO is uniquely suited for simulated online learning, where it can adapt to dynamic environmental changes.

2 Preliminaries

In reinforcement learning for language models, the model is parameterized by θ as a stochastic policy $\pi_\theta(y|q)$, which generates a response sequence $y = [y_1, \ldots, y_{|y|}]$ given a query q from dataset $\mathcal{D}$. RL optimizes π_θ by maximizing a clipped surrogate objective that encourages stable policy updates. Formally, for a given batch of data, the unified optimization target is defined as:

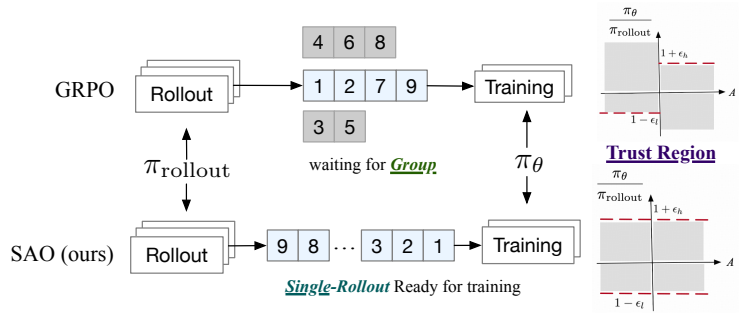


Figure 2: Overview of SAO with single rollout design. The numbers denote the generation order of trajectories. For SAO, each trajectory becomes available for training immediately upon completion. In contrast, GRPO must wait until all trajectories in a group are generated before training can begin.

$$\mathbb{E}\left[\frac{1}{|y|}\sum_{t=1}^{|y|}\min\left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right]$$

where $r_t(\theta) = \frac{\pi_\theta(y_t|q,y_{<t})}{\pi_{\theta_{\text{old}}}(y_t|q,y_{<t})}$ is the probability ratio between the current and old policies, ϵ is the clipping hyperparameter. The fundamental distinction between PPO [Schulman et al., 2017] and GRPO [DeepSeek-AI, 2024b] lies in whether to estimate the advantage function $\hat{A}_t$ and the necessity of auxiliary value networks.

Proximal Policy Optimization (PPO). Standard PPO typically adopts an Actor-Critic architecture, requiring the training of a separate value function (Critic) V_ϕ, parameterized by ϕ, to estimate the expected return of the current state. This critic is optimized concurrently with the policy to minimize the value error $\mathcal{L}_\phi^{\text{VF}} = \mathbb{E}[(V_\phi(q, y_{<t}) - R)^2]$, where R denotes the cumulative reward. To balance bias and variance, PPO employs Generalized Advantage Estimation (GAE). The advantage $\hat{A}_t^{\text{GAE}}$ is computed as an exponentially weighted sum of temporal difference errors:

$$\hat{A}_t^{\text{GAE}} = \sum_{l=0}^{|y|-t-1}(\gamma\lambda)^l\delta_{t+l}$$

where $\delta_t = r_t + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$. While effective, this approach necessitates maintaining a copy of the model parameters for the value function, essentially doubling the memory footprint during training and increasing computational overhead.

3 Asynchronous Reinforcement Learning with Single Rollout

In this section, we introduce SAO to tackle training instability and off-policy drift in asynchronous RL training. With a simple token-level clipping strategy and single rollout as an alternative to group-wise sampling, we show that asynchronous RL can be stably scaled to thousands of training steps and achieve significant performance improvements. Figure 2 shows the overall design of SAO.

3.1 Stabilizing Asynchronous RL via Direct Double-Sided Importance Sampling (DIS)

A primary challenge in asynchronous RL is the "policy lag" that emerges between rollout models and the training models. In decoupled PPO for LLM, importance sampling is employed to relieve off-policy bias by keeping three distinct models: the current policy π_θ, the old policy $\pi_{\theta_{\text{old}}}$, and the rollout policy π_{rollout}, where $\frac{\pi_\theta}{\pi_{\theta_{\text{old}}}}$ is used for staled off-policy correction and $\frac{\pi_{\theta_{\text{old}}}}{\pi_{\text{rollout}}}$ for training-rollout mismatch. However, as rollout engines may undergo multiple updates during a single trajectory generation in asynchronous RL, this renders the tracking of exact behavior probabilities $\pi_{\theta_{\text{old}}}$ computationally prohibitive. Otherwise, we have to maintain an extensive history of model checkpoints $\{\pi_{\theta_{\text{old}}^{(1)}}, \ldots, \pi_{\theta_{\text{old}}^{(N)}}\}$, which is infeasible in practical implementation.

To resolve this, we propose a simplified yet aggressive token-level importance sampling to clip off-policy tokens. First, we directly use π_{rollout} as the behavior proxy and π_θ for importance sampling, i.e., $r_t(\theta) = \frac{\pi_\theta}{\pi_{\text{rollout}}}$, while dropping the inaccurate $\pi_{\theta_{\text{old}}}$. This eliminates the computational overhead of separate old-policy inference by utilizing the log-probabilities generated during the rollout phase.

Second, we employ a double-sided calibration token-level masking strategy. Unlike standard PPO clipping, which clips only selected off-policy tokens with $(A > 0, r_t(\theta) > 1+\epsilon_h)$ or $(A < 0, r_t(\theta) < 1 - \epsilon_l)$, we restrict the trust region to the interval $[1 - \epsilon_\ell, 1 + \epsilon_h]$, while tokens falling outside this range are masked from gradient computation entirely to prevent instabilities arising from extreme policy divergence. This shares similarities with the IcePop mechanism Team et al. [2025], yet our strategy is simpler by further removing $\pi_{\theta_{\text{old}}}$ while still achieving stable training.

Formally, the optimization objective with token-level clipping can be written as:

$$L(\theta) = \hat{\mathbb{E}}_t \left[f(r_t(\theta), \epsilon_l, \epsilon_h) \hat{A}_t \log \pi_\theta(a_t|s_t) \right] \tag{1}$$

In this formulation, the probability ratio $r_t(\theta)$ is computed directly from the rollout logs to circumvent the need for historical policy tracking:

$$r_t(\theta) = \exp\left(\log \pi_\theta(a_t|s_t) - \log \pi_{\text{rollout}}(a_t|s_t)\right) \tag{2}$$

Stability is further enforced via the calibration function $f(x; \epsilon_\ell, \epsilon_h)$:

$$f(x; \epsilon_\ell, \epsilon_h) = \begin{cases} x, & \text{if } 1 - \epsilon_\ell < x < 1 + \epsilon_h \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

This design circumvents the intensive need to track the historical model ensemble. By utilizing the rollout log-probabilities directly, we accept a controlled degree of off-policy bias in exchange for a substantial reduction in computational complexity and the elimination of errors associated with using a single, potentially stale, "latest" old policy model. Empirical results demonstrate that this simplified mechanism enables more aggressive clipping, which effectively regularizes the update steps and yields superior training stability in asynchronous settings.

3.2 Reducing Off-Policy with Single Rollout

In asynchronous RL, an inevitable problem is off-policy. Yet current popular group-wise sampling RL algorithms, e.g., GRPO, could introduce more severe off-policy. Group-wise sampling introduces an "imbalanced generation" bias, and the group data has to wait for the "slowest" sample to finish before being fed into training. One promising solution is to replace group-wise sampling with single-rollout, where a sample is immediately fed into training upon generation.

However, single-rollout optimization inherently suffers from high variance in gradient estimation, similar to REINFORCE Zhang et al. [2021]. To reduce variance requires a sufficiently good value model. In this part, we focus on simple strategies to optimize value modeling to ultimately boost the policy's performance.

Faster Value Update than Policy. We identify that the primary source of instability in single-rollout RL is the interdependence between the policy and the value function. If the value model V_ϕ is inaccurate, the advantage estimates $\hat{A}_t$ become noisy, leading to destructive policy updates. To mitigate this, we implement a Faster Value Update adapted for LLMs. We decouple the optimization frequencies of the policy and the value model. Specifically, for every single gradient update applied to the policy π_θ, we enforce K updates to the value network V_ϕ (where $K > 1$). In our experiments, we set $K = 2$. This strategy facilitates the faster adaptation of value estimates to the current policy before they are utilized for advantage computation, thereby reducing the variance.

Stabilizing Value Model Training via Parameter Freezing. In our pilot experiments, we find the instability of value model training, where the gradient norms of the value model are significantly larger than the corresponding policy model. Further decomposition shows that this instability originates primarily from the Full Attention layers, whereas the Mixture-of-Experts (MoE) layers remain relatively stable. Based on this observation, we employ a "Frozen-Attention" training strategy for the value model. During the RL training, we freeze the parameters of the attention modules in V_ϕ and optimize the MoE projections. We hypothesize that the pre-trained attention weights already possess

Table 1: Experimental Results on math reasoning benchmarks(Accuracy %).

Model	AIME2025	BeyondAIME	HMMT Nov 2025	IMOAnswerBench
Claude-Sonnet-4.5	87.0	62.0	81.7	65.8
GPT-5 High	94.6	74.0	89.2	76.0
GLM-4.7	95.7	-	93.5	82.0
Qwen3-30B-A3B				
w/ python	14.6	10.5	17.3	7.8
w/o python	85.0	63.0	76.7	55.3
SFT (w/ python)	80.4	53.3	75.2	53.3
SFT (w/o python)	14.6	46.8	17.3	42.0
GRPO (w/ python)	84.2	54.8	76.0	55.8
Qwen3-30B-A3B				
SAO (ours)	**97.3**	**74.8**	**88.3**	**74.0**
- *SAO (w/ DIS only)*	94.2	71.5	86.7	71.3
- *GRPO (+ DIS)*	93.5	70.8	84.0	70.0

sufficient semantic capability to attend to relevant tokens. By restricting optimization to the MoE layers, we effectively regularize the value model.

Skip-Observation Token-level GAE for Agentic Tasks. Agentic tasks present a unique challenge for token-level value estimation due to their trajectory structure: $T = [a_0, o_0, a_1, o_1, \dots]$, where a_i represents model actions and o_i represents environment feedback. Standard Generalized Advantage Estimation (GAE) attempts to calculate the value difference between adjacent tokens. However, the transition from the end of an action $a_{i,\text{end}}$ to the start of an observation $o_{i,\text{start}}$ is discontinuous from the model's perspective, as the model does not generate o_i. Calculating advantage across this boundary introduces noise, as the value model $V(o_{i,\text{start}})$ attempts to predict the value of an external environment state.

To resolve this, we derive a "Skip-Observation" GAE. We explicitly modify the Bellman target to bypass environment feedback tokens, linking the value of the current action directly to the value of the subsequent action. Formally, let $a_{i,N}$ be the last token of action i, and $a_{i+1,0}$ be the first token of the next action. We define the advantage as:

$$\hat{A}(a_{i,N}) = \delta + \gamma\lambda\hat{A}(a_{i+1,0}) \tag{4}$$

where the temporal difference residual δ is calculated bridging the observation gap:

$$\delta = r_t + \gamma V(a_{i+1,0}) - V(a_{i,N}) \tag{5}$$

This formulation constrains the advantage estimation to rely purely on the model outputs, filtering out the stochasticity of environment feedback. In contrast, some works may consider using a step-level value function and GAE as an alternative to the token-level value; however, we found that a step-level value could lead to suboptimal performance, which will be shown in the experimental part. We also conduct other advantage designs for agentic traces, and the results can be found in the Appendix.

Scaling Value Pretraining. Finally, to support these mechanisms, we find it essential to scale the data used for value model pretraining. Our experiments demonstrate that the "cold start" problem in value estimation is a major bottleneck. By significantly increasing the scale of the value pretraining corpus, we provide a robust initialization point that promotes the effectiveness of our single-rollout and TTUR mechanisms from the early stages of training.

Table 2: Experimental Results on SWE-Bench Verified (Accuracy %).

Model	Accuracy (%)
Qwen3-30B-A3B	23.0
+ GRPO (w/ DIS)	27.0
+ SAO (ours)	29.8

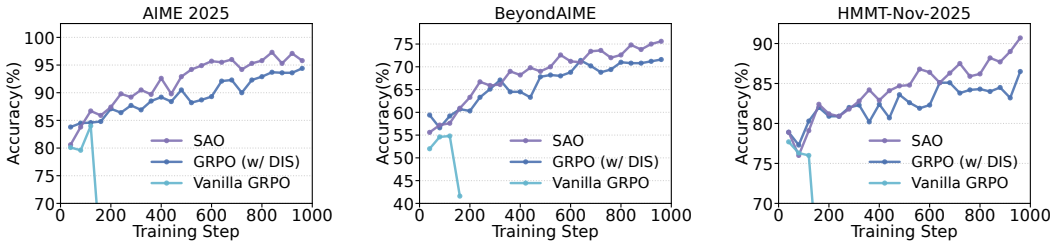


Figure 3: Performance comparison between SAO and GRPO (w/ DIS) during training. It can be observed that SAO almost consistently outperforms the optimized GRPO during the training process on different benchmarks.

4 Experiments

4.1 Experimental Setup

Training Details. For math reasoning with Python, we finetune Qwen3-30B-A3B-Thinking-2507 [Yang et al., 2025a] for 3 epochs on Tool-Integrated Reasoning (TIR) data produced by GPT-OSS-120B [OpenAI, 2025] and use the finetuned model to initialize the policy and value model. TIR requires the model to interleave natural-language math reasoning with Python tool calls.

For RL of agentic reasoning, we employ a batch size of 128, a group size of 1, and a max-length of 128k tokens. The policy is optimized with a learning rate of 1×10^{-6}, with a token clipping of $\epsilon_{\text{low}} = 0.3$, $\epsilon_{\text{high}} = 5.0$. We adopt a length-adaptive GAE [Yue et al., 2025] with $\lambda_{\text{policy}} = 1 - \frac{1}{\alpha l}$ and $\alpha = 1.5$. The value model is trained with a learning rate of 5×10^{-6}, $\lambda_{\text{critic}} = 1$, and a 10-step warmup period. We set the $K = 2$ for faster value update for the value model, performing two value model updates per batch. For GRPO variants, each training batch contains 16 prompts with 8 rollout samples per prompt, yielding the same batch size of 128. For the RL of coding agent, we directly use Qwen3-30B-A3B-Thinking-2507 for training and keep almost all the hyperparameters the same as TIR, except for $\epsilon_{\text{low}} = 0.8$ and $\epsilon_{\text{high}} = 3.0$. For SWE-Bench Verified, we use OpenHands as the scaffold, with a maximum of 300 interaction turns and a 128k-token context budget.

Evaluation. We evaluate SAO on four math reasoning benchmarks including AIME2025, BeyondAIME[ByteDanceSeed, 2025], HMMT Nov 2025[Balunović et al., 2025] and IMOAnswerBench[Luong et al., 2025], reporting Pass@1 accuracy. All evaluations use top-$p = 1.0$, temperature 1.0, and a maximum generation length of 128k tokens. Math-reasoning evaluations allow up to 50 turns to support extensive reasoning and tool calls, while SWE-Bench Verified evaluations allow up to 300 OpenHands interaction turns. To reduce variance, we report the mean performance across 16 evaluation runs for AIME2025 / HMMT / IMOAnswerBench and 4 runs for BeyondAIME.

4.2 Main Results

Tables 1 and 2 summarize the performance of baselines and different training strategies. GRPO denotes the standard GRPO with *clip-higher* implementation Yue et al. [2025], which keeps the latest old policy for importance sampling. GRPO (w/ DIS) denotes using the proposed DIS strategy for GRPO.

As shown in Tables 1 and 2, SAO consistently outperforms all baselines on both agentic reasoning and coding benchmarks. Standard GRPO suffers from a performance collapse at approximately 160 training steps The scores reported for these models represent their final valid performance before collapsing. Figure 3 illustrates the evaluation performance across training steps of SAO compared to

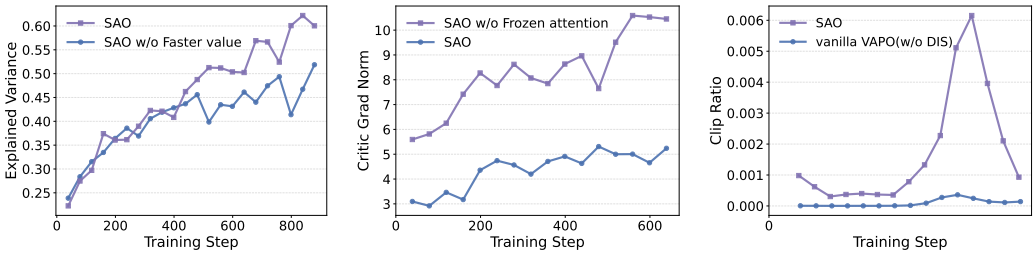
vanilla GRPO and GRPO (w/ DIS). Vanilla GRPO tends to quickly collapse, while GRPO with DIS can achieve stable training, demonstrating the effectiveness of DIS. In addition, SAO and GRPO (w/ DIS) exhibit comparable performance in the initial stage; a distinct performance divergence occurs after approximately 400 training steps, demonstrating the effectiveness and stability of SAO.

4.3 Ablation Studies

We conduct extensive ablation studies to evaluate the impact of various training configurations on the performance of SAO. The results are shown in Table 4.

- *Effects of faster value update*: To ablate the effects of faster value update than the policy model, we conduct experiments where the value model is updated only once per batch (critic-train-epoch=1), as opposed to the two updates per batch employed in SAO.
- *Full vs. frozen-attention value model.*: To evaluate the impact of attention-freezing during RL, this variant performs full-parameter updates on the value model.
- *Vanilla VAPO and single-rollout with Running-mean baseline.* Standard VAPO[Yue et al., 2025] with length-adaptive GAE and a value-based RL baseline. Besides, a single-rollout baseline that maintains a sliding window of the 8 most recent rewards for each prompt, using their mean as a baseline for advantage estimation to provide a simple alternative to parametric value models.

As shown in Table 4, all examined variants exhibit a performance decline relative to the proposed SAO, validating the necessity of each design choice. Table 3 further summarizes the value-training strategy and critic-update settings behind the main value-model ablations. Regarding update frequency, the results indicate that a single update is insufficient for the critic to accurately track rapid policy shifts, leading to less reliable baseline estimations. The full-parameter value-training variant further suggests that frozen-attention updates help regularize critic optimization in complex reasoning tasks. In addition, the RL with running-mean reward achieves decent performance, but still lags far from our SAO, demonstrating the advantage and necessity of a well-trained value model for RL. As for the vanilla VAPO, the training also quickly collapses during training, similar to the vanilla GRPO.



(a) (b) (c)

Figure 4: Training dynamics of asynchronous single-rollout RL. (a) Explained Variance for SAO and a single-critic-update baseline. (b) Critic gradient norm during value training under full-parameter optimization and frozen-attention optimization used in SAO. (c) Token-level clip ratio during training for SAO with the proposed DIS and the VAPO baseline.

4.4 Training Dynamics

We analyze the training dynamics of SAO to understand how it facilitates training stability.

Effects of Faster Value Update. Figure 4(a) illustrates Explained Variance comparison between SAO and the single-critic-update baseline during training. Explained Variance assesses the alignment between the predicted values $V(s)$ and the ground-truth returns R, defined as $EV = 1 - \frac{\mathrm{Var}(R - V(s))}{\mathrm{Var}(R)}$. SAO demonstrates significantly higher explained variance after approximately 400 training steps, indicating faster value convergence and better alignment with policy distribution.

Gradient of Critic Models. We examine the impact of freezing attention parameters in value training. As shown in Figure 4(b), full-parameter value training exhibits significantly larger critic

gradient norms, suggesting unstable optimization dynamics. In contrast, the frozen-attention strategy maintains lower and smoother gradient norms, implying improved numerical stability.

Clipped Tokens. Figure 4(c) monitors the token-level clip ratio of SAO applying our proposed DIS strategy and the standard VAPO baseline without it. While VAPO maintains a near-zero clip ratio, it fails to effectively gate divergent off-policy updates, leading to a rapid training collapse at approximately 90 steps.

Table 3: Ablation results of value model training strategy and critic update frequency. We compare partial parameters, i.e., frozen-attention, with full-parameter value update in RL training, as well as the effectiveness of faster critic updates per policy step. We report Accuracy (%) for all datasets

	Value Training Strategy	Critic Update Frequency	AIME2025	BeyondAIME
SAO	Frozen Attention	2	97.3	74.8
Single-step-update	Frozen Attention	1	95.00	69.75
Full-Parameter Value Training	Full-Parameter	2	90.62	74.50

Table 4: Ablation results of value model training strategy and critic update frequency. We compare partial parameters, i.e., frozen-attention, with full-parameter value update in RL training, as well as the effectiveness of faster critic updates per policy step. We report Accuracy (%) for all datasets

	AIME2025	BeyondAIME
SAO	97.3	74.8
SAO w/o Faster value	95.0	69.8
SAO w/o Frozen attention	90.6	74.5
Vanilla VAPO (w/o DIS)	91.3	69.0
Running mean baseline	79.8	55.3

4.5 Online Learning Simulation

Task Design. In real-world online learning environments, feedback is typically restricted to a single trajectory per prompt. This constraint is inherently incompatible with group-based optimization strategies like GRPO, which depend on relative rewards within a sample group for advantage estimation. In contrast, SAO utilizes a value-based critic to provide advantage estimation, allowing for effective policy updates from individual trajectories.

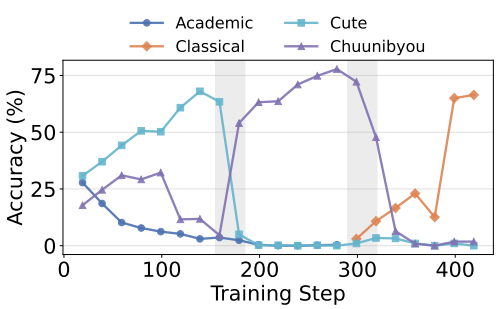
Therefore, we design a simulated online writing task to assess the adaptability of SAO in non-stationary environments. In this setting, the feedback signal is designed as the language tone of user preference. Reward criteria are sequentially adjusted to favor three distinct stylistic archetypes: *cute*, *chuunibyou*, and *classical*.

Dynamic Reward Assignment. For reward signal assignment, we employ GLM-4.7[GLM et al., 2025] as an LLM-based judge to evaluate two primary dimensions: response quality and stylistic adherence. The final reward $r \in \{0, 1\}$ is computed as: $r = r_{\text{quality}} \times r_{\text{style}}$ where $r_{\text{quality}}, r_{\text{style}} \in \{0, 1\}$ denote binary rewards.

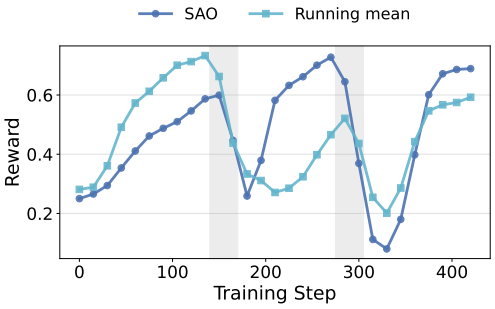
Throughout training, the system prompt requires the model to select one stylistic archetype from a pool of four candidates: Academic, Cute, Chuunibyou, and Classical. In our actual experiments, the candidate set consists of *Academic, Cute, Chuunibyou* during the first two phases, and *Classical, Cute, Chuunibyou* in the final phase.

Results. As illustrated in Figure 5a, we evaluate the performance of the three candidate linguistic styles of each phase on a held-out test set . SAO demonstrates rapid policy realignment following each reward preference shift, characterized by transitions between stylistic archetypes to maintain adherence to the evolving environmental feedback.

Comparison against Running-Mean Baseline. To better understand the effectiveness of the value model in the online environment, we also adopt the Running Mean Advantage Estimation approach as the baseline. This method approximates the baseline b by tracking a sliding window of the 128 most recent rewards, thereby facilitating advantage computation as $\hat{A} = r - \mathbb{E}[r_{window}]$. By decoupling advantage estimation from intra-prompt sample groups, this setup permits policy optimization in an online, single-rollout context. Figure 5b depicts the evolution of training rewards throughout the online learning process of SAO and the Running Mean baseline, where the speed and magnitude of reward recovery following stylistic shifts serve as key indicators of algorithmic adaptability. The Running Mean baseline exhibits a pronounced adaptation lag due to the inertia of its historical window, which remains temporarily biased by rewards from the preceding distribution. In contrast, SAO's value-based critic dynamically tracks reward shifts, facilitating rapid recovery and consistently higher convergence levels. This confirms that SAO's state-dependent baseline provides the precision necessary for effective alignment in non-stationary environments.



(a) We report the accuracy transition of three writing styles—cute, chuunibyou, and classical—on a held-out evaluation set throughout the online training process. Shaded regions indicate phase transitions where the reward preference is switched to favor a different stylistic archetype. SAO rapidly suppresses the previously dominant style and realigns its policy to the new target based on environmental feedback.



(b) We compare the evolution of training rewards between SAO and a Running Mean Advantage Estimation baseline under single-rollout online learning. Shaded regions denote stylistic reward shifts. While both methods eventually recover after distribution changes, the Running Mean baseline exhibits a pronounced adaptation lag and lower stable performance.

Figure 5: Online learning simulation under changing writing-style preferences.

5 Related Work

5.1 Reinforcement Learning for Language Models

The standard RLHF pipeline trains a reward model from preference data and optimizes the policy with PPO[Ouyang et al., 2022, Schulman et al., 2017]. To reduce the overhead and instability of value-function learning, critic-free objectives such as Group Relative Policy Optimization (GRPO) [Shao et al., 2024, DeepSeek-AI, 2024a] and REINFORCE-style baselines (e.g., RLOO) [Ahmadian et al., 2024] have become increasingly popular. GRPO forms advantages by normalizing rewards within a prompt-level group, which improves stability in synchronous training but introduces an implicit synchronization barrier: updates must wait until all group members are generated, exacerbating staleness and off-policy drift under asynchrony.

Recent work further refines GRPO/PPO-style objectives to improve stability and variance reduction, including sequence-level importance weighting [Zheng et al., 2025], adaptive clipping strategies [Yang et al., 2025b], and smoother alternatives to hard clipping [Yue et al., 2025]. However, these works focus primarily on synchronous RL, where exact importance-sampling ratios are easier to obtain. Importance sampling and clipping strategies for asynchronous RL remain less explored.

5.2 Synchronous and Asynchronous RL for LLMs

Most large-scale LLM RL implementations remain synchronous and interleaved: collect a full batch of rollouts with a fixed policy snapshot, then run optimization epochs on that batch [Ouyang et al., 2022]. With long-tail output lengths in reasoning and tool-use, synchronous barriers cause stragglers and substantial idle time, motivating asynchronous actor–learner designs where rollout generation and learning proceed concurrently [Mnih et al., 2016, Sutton and Barto, 2018]. However, asynchrony introduces policy lag and off-policy drift, often requiring staleness-aware training or off-policy corrections [Espeholt et al., 2018].

Several recent systems target asynchronous RL specifically for LLMs. Noukhovitch et al. [2024] study asynchronous RLHF as online-but-off-policy learning and characterize robustness tradeoffs. On the systems side, AReaL [Fu et al., 2025] fully decouples rollout from training and incorporates staleness-aware PPO-style updates for reasoning tasks. ROLL Flash provides fine-grained parallelism and rollout–train decoupling for RLVR and agentic training [Lu et al., 2025]. Complementary to asynchronous systems, MobileRL studies online agentic RL for mobile GUI agents and introduces difficulty-adaptive GRPO variants to improve stability and sample efficiency in multi-turn GUI environments [Xu et al., 2025]. Our work complements these systems by focusing on the single-rollout setting where group-based baselines (e.g., GRPO) are structurally mismatched, and by stabilizing asynchronous learning algorithm designs.

6 Conclusion

In this work, we explore the optimization of asynchronous RL on the training effectiveness and stability We proposed SAO, a single-rollout asynchronous RL strategy that addresses off-policy and instability. SAO stabilizes training with token-level importance sampling and double-sided clipping/masking, and improves generalization by replacing group-wise sampling with single-rollout enabled by stronger value-model training. On agentic reasoning and coding tasks, SAO shows consistent outperformance over GRPO baselines, and adapts effectively in simulated online learning.

References

Arash Ahmadian et al. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms. *arXiv preprint arXiv:2402.14740*, 2024.

Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. Matharena: Evaluating llms on uncontaminated math competitions. *arXiv preprint arXiv:2505.23281*, 2025.

ByteDanceSeed. Beyondaime: Advancing math reasoning evaluation beyond high school olympiads, 2025.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint*, 2024a.

DeepSeek-AI. Deepseek-v3 technical report. *arXiv preprint*, 2024b.

Lasse Espeholt et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, 2018.

Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, Jiashu Wang, et al. Areal: A large-scale asynchronous reinforcement learning system for language reasoning. *arXiv preprint arXiv:2505.24298*, 2025.

Team GLM, Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, et al. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*, 2025.

Matt Hoffman et al. Acme: A research framework for distributed reinforcement learning. *arXiv preprint arXiv:2006.00979*, 2020.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*, 2023.

Woosuk Kwon, Zhuohan Li, Siyuan Zhang, Xuguang Zhuang, Ying Sheng, Lianmin Zheng, Cody Fonseca, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, 2023.

Eric Liang, Richard Liaw, Robert Nishihara, Philipp Moritz, Roy Fox, Ken Goldberg, and Ion Stoica. Rllib: Abstractions for distributed reinforcement learning. In *International Conference on Machine Learning*, pages 3053–3062, 2018.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Han Lu, Zichen Liu, Shaopan Xiong, Yancheng He, Wei Gao, Yanan Wu, Weixun Wang, Jiashun Liu, Yang Li, Haizhou Zhao, et al. Part ii: Roll flash–accelerating rlvr and agentic training with asynchrony. *arXiv preprint arXiv:2510.11345*, 2025.

Thang Luong, Dawsen Hwang, Hoang H. Nguyen, Golnaz Ghiasi, Yuri Chervonyi, Insuk Seo, Junsu Kim, Garrett Bingham, Jonathan Lee, Swaroop Mishra, Alex Zhai, Clara Huiyi Hu, Henryk Michalewski, Jimin Kim, Jeonghyun Ahn, Junhwi Bae, Xingyou Song, Trieu H. Trinh, Quoc V. Le, and Junehyuk Jung. Towards robust mathematical reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025. URL `https://aclanthology.org/2025.emnlp-main.1794/`.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PmLR, 2016.

Reiichiro Nakano et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

Michael Noukhovitch, Shengyi Huang, Sophie Xhonneux, Arian Hosseini, Rishabh Agarwal, and Aaron Courville. Asynchronous rlhf: Faster and more efficient off-policy rl for language models. *arXiv preprint arXiv:2410.18252*, 2024.

OpenAI. Learning to reason with llms. *https://openai.com/index/learning-to-reason-with-llms*, 2024.

OpenAI. gpt-oss-120b & gpt-oss-20b model card, 2025. URL `https://arxiv.org/abs/2508.10925`.

Long Ouyang et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744, 2022.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, 2024.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Ling Team, Anqi Shen, Baihui Li, Bin Hu, Bin Jing, Cai Chen, Chao Huang, Chao Zhang, Chaokun Yang, Cheng Lin, et al. Every step evolves: Scaling reinforcement learning for trillion-scale thinking model. *arXiv preprint arXiv:2510.18855*, 2025.

Xuezhi Wang et al. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Yifan Xu, Xiao Liu, Xinghan Liu, Jiaqi Fu, Hanchen Zhang, Bohao Jing, Shudan Zhang, Yuting Wang, Wenyi Zhao, and Yuxiao Dong. MobileRL: Online agentic reinforcement learning for mobile gui agents. *arXiv preprint arXiv:2509.18119*, 2025. URL `https://arxiv.org/abs/2509.18119`.

Zhongwen Xu and Zihan Ding. Single-stream policy optimization. *arXiv preprint arXiv:2509.13232*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a.

Shihui Yang, Chengfeng Dou, Peidong Guo, Kai Lu, Qiang Ju, Fei Deng, and Rihui Xin. Dcpo: Dynamic clipping policy optimization. *arXiv preprint arXiv:2509.02333*, 2025b.

Shunyu Yao et al. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Gyeong-In Yu, Joo Seong Jeong, Geon-Woo Kim, Soojeong Kim, and Byung-Gon Chun. Orca: A distributed serving system for Transformer-based generative models. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*, pages 521–538, 2022.

Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, et al. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*, 2025.

Junzi Zhang, Jongho Kim, Brendan O'Donoghue, and Stephen Boyd. Sample efficient reinforcement learning with reinforce. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 10887–10895, 2021.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

A Additional Experimental Results

A.1 RL with Agentic Step as Action

To mitigate the high variance inherent in token-level value predictions, we implement a step-wise GAE calculation. We define a step (denoted as S_i) as a single conversation turn and assume that all constituent tokens share a uniform learning signal within each step. We define the step-level value $V(S_i)$ using two primary aggregation methods based on constituent token values predictions $\{v_{i,1}, v_{i,2}, \ldots, v_{i,n}\}$:

- *Step Average.* The step value is the average of all token value predictions within that step, $V(S_i) = \frac{1}{n}\sum_{j=1}^{n} v_{i,j}$. In this setting, the value model is trained on all tokens.
- *Last-Token Prediction.* We define the step value as $V(S_i) = v_{i,n}$, using only the final token of each step, assuming that the final token provides the most accurate value prediction of the step, as it encapsulates the most comprehensive information of the entire unit. During value model training, we apply a loss mask to all intermediate tokens, ensuring that only the last token of each step contributes to the optimization.

To provide a more stable learning signal, we implement a step-wise GAE calculation that shifts the advantage estimation from a token-level to a step-level granularity. We derive a single advantage $\hat{A}_i$ for each step based on the step-level TD error $\delta_i = R_i + \gamma V(S_{i+1}) - V(S_i)$. This advantage is then assigned uniformly to all tokens within the corresponding step, effectively smoothing out the local noise prevalent in auto-regressive generation. Furthermore, the length-adaptive GAE mechanism is modified to scale the decay factor λ based on the total number of steps rather than the raw token length, where

$$\lambda_{\text{policy}} = 1 - \frac{1}{\alpha * \text{step number}}$$

However, as illustrated by the training reward in Figure 6 and the performance results in Table 5, both step-wise approaches underperform token-wise value training. We attribute this failure to the fact that token-level training provides a finer-grained supervision signal for both the critic and the policy, which is essential for accurately capturing the logical transitions within complex reasoning trajectories.

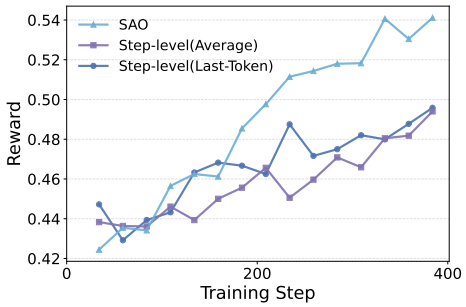


Figure 6: Training reward for token-level SAO training and step-level variants, where token-level shows better training rewards.

Table 5: The ablation on the action granularity for value and policy model training. *Step-level* denotes that each agent *step* is viewed as an action to calculate the value. *Token-level* refers to each token being viewed as an action. We report the results with the same training steps (400 steps).

	AIME2025	BeyondAIME
Step-level (Average)	85.8	60.5
Step-level (Last-Token)	87.3	62.8
Token-level	89.8	66.8

A.2 Comparison to Other Baselines of Single-Rollout Strategies

SPO Xu and Ding [2025] or directly using the historical running-mean reward as the baseline for advantage estimation are also feasible ways to achieve RL with a single rollout per prompt. However, SPO and running-mean baselines rely on prior information about training-data difficulty and achieve worse performance than SAO, as shown in the experiment section.

B Limitations and Broader Impact

Our experiments focus on large-scale agentic reasoning, coding, and simulated online writing tasks with a Qwen3-30B-A3B backbone. The conclusions therefore may not transfer directly to smaller models, non-agentic RLHF settings, or environments with dense rewards and shorter rollouts. In addition, SAO depends on a trained value model and rollout log-probabilities, so deployment requires infrastructure that can reliably preserve token-level behavior probabilities during asynchronous generation. The online learning study uses a controlled simulated preference shift; real user-facing online adaptation would require stronger safeguards, monitoring, and privacy review before deployment.

By improving the stability and efficiency of LLM reinforcement learning, this work can reduce the cost of training capable agentic systems. The same capability could also make it easier to optimize models for harmful objectives if used without appropriate data filtering, access controls, or evaluation, so responsible release and monitoring are important for any deployed system derived from this work.