# Enhancing Agentic RL with Progressive Reward Shaping and Value-based Sampling Policy Optimization

**Jianghao Su[1*], Xia Zeng[1†], Luhui Liu[1], Luo Chao[1], Ye Chen[1], Zhuoran Zhuang[1]**

[1]Fliggy Alibaba
Zhejiang, Hangzhou
jh.su@stu.xjtu.edu.cn, {zengxia.xz, liuluhui.llh, guotai.lc, nantian, chen.yechen}@alibaba-inc.com

## Abstract

Large Language Models (LLMs) empowered with Tool-Integrated Reasoning (TIR) can iteratively plan, call external tools, and integrate returned information to solve complex, long-horizon reasoning tasks. Agentic Reinforcement Learning (Agentic RL) optimizes such models over full tool-interaction trajectories, but two key challenges hinder effectiveness: (1) Sparse, non-instructive rewards, such as binary 0–1 verifiable signals, provide limited guidance for intermediate steps and slow convergence; (2) Gradient degradation in Group Relative Policy Optimization (GRPO), where identical rewards within a rollout group yield zero advantage, which reducing sample efficiency.

To address these challenges, we propose two complementary techniques: Progressive Reward Shaping (PRS) and Value-based Sampling Policy Optimization (VSPO). PRS is a curriculum-inspired reward design that introduces dense, stage-wise feedback — encouraging models to first master parseable and properly formatted tool calls, then optimize for factual correctness and answer quality. We instantiate PRS for short-form QA (with a length-aware BLEU to fairly score concise answers) and long-form QA (with LLM-as-a-Judge scoring to prevent reward hacking). VSPO is an enhanced GRPO variant that replaces zero advantages samples with prompts selected by a task-value metric balancing difficulty and uncertainty, and applies value-smoothing clipping to stabilize gradient updates.

Experiments on multiple short-form and long-form QA benchmarks show that PRS consistently outperforms traditional binary rewards, and VSPO achieves superior stability, faster convergence, and higher final performance compared to SFT, PPO and GRPO baselines. Together, PRS and VSPO yield LLM-based TIR agents that generalize better across domains.

## Introduction

Large language models (LLMs) have shown remarkable capabilities in complex reasoning and tool-augmented decision-making, enabling applications such as question answering, planning, and interactive agents (Zhao et al. 2023; Huang et al. 2024). When deployed as Tool-Integrated Reasoning (TIR) agents, LLMs are equipped with external tools — e.g., search APIs or knowledge bases — and can iteratively plan, call tools, and integrate their outputs to produce final answers. This multi-step, interleaved reasoning-with-action process is particularly effective in long-horizon tasks such as multi-hop question answering or problem solving in dynamic environments.

Training high-performing LLM-based TIR agents goes beyond single-turn reinforcement learning (Zhang et al. 2025). Agentic reinforcement learning (Agentic RL) explicitly optimizes the LLM policy over full tool-interaction trajectories, with feedback signals derived from the correctness and quality of the intermediate and final outputs. However, in practice, Agentic RL for TIR faces two key challenges:

- **Sparse and non-instructive rewards:** Widely used reinforcement learning with verifiable rewards (RLVR) (Liu et al. 2025) often employs binary 0-1 signals to indicate pass/fail based on the final answer. These sparse signals provide limited guidance for improving intermediate tool-use steps, offer no notion of partial progress, and can lead to inefficient exploration and unstable optimization.
- **Low sample efficiency in GRPO:** Group Relative Policy Optimization (GRPO) (Guo et al. 2025) is a strong baseline for multi-sample policy improvement, but when all rollouts in a group receive identical rewards, their computed advantages become zero. This results in dampened gradient updates and low sample efficiency.

To address these issues, we propose two complementary techniques for training LLM-based TIR agents:

- *Progressive Reward Shaping* (PRS): a curriculum-inspired reward design that augments sparse verifiable rewards with dense intermediate signals, enabling the model to first master essential capabilities — generating parseable and correctly formatted tool calls — before optimizing more challenging objectives such as factual correctness and answer quality. In short-form QA, PRS incorporates a length-aware BLEU to avoid unfair penalization of correct short answers; in long-form QA, it integrates LLM-as-a-Judge scoring to mitigate reward hacking.
- *Value-based Sampling Policy Optimization* (VSPO): an improved variant of GRPO that replaces low-value (zero-advantage) samples with prompts selected according to a task-value metric balancing uncertainty and difficulty.

[*]Also affiliated with Xi'an Jiaotong University, work done during internship at Fliggy Alibaba

[†]Corresponding Author

VSPO further applies value-smoothing clipping to maintain stable gradient scales even when high-value prompts are repeatedly sampled.

Experiments on long-form and short-form QA benchmarks demonstrate that PRS consistently outperforms traditional binary reward schemes, and VSPO achieves more stable training and superior performance compared to SFT, PPO and GRPO baselines. Together, PRS and VSPO yield LLM-based TIR agents that converge faster, generalize better across domains.

Our contributions are summarized as follows:

- We design PRS, a general reward shaping framework that integrates stage-wise progression and dense intermediate feedback, instantiated for both short- and long-form QA.
- We develop VSPO, a GRPO variant with value-based sampling and value smoothing clipping, improving sample efficiency and training stability and performance.
- We empirically validate PRS and VSPO across multiple QA benchmarks, showing consistent gains over strong baselines.

## Related Work

**Tool-Integrated Reasoning.** Tool-Integrated Reasoning (TIR) (Lin and Xu 2025) augments LLMs by enabling them to interleave natural-language reasoning with external tool calls, such as code execution or search, thereby improving adaptability in complex tasks. Early TIR systems based on in-context learning or SFT (Yao et al. 2022; Schick et al. 2023) achieve promising results but often generalize poorly to unseen tasks and new tool configurations. Recent advances employ Reinforcement Learning with Verifiable Rewards (RLVR) (Liu et al. 2025), allowing agents to learn robust tool-use strategies through exploration, outperforming purely imitative methods. RL frameworks such as ToolRL (Qian et al. 2025), AutoTIR (Wei et al. 2025), and VTool-R1 (Wu et al. 2025) interleave symbolic computation with language reasoning in unified rollouts, balancing precise tool-mediated operations with flexible verbal inference.

**Agentic RAG.** Agentic RAG (Ravuru, Sakhinana, and Runkana 2024) enhances traditional RAG frameworks by embedding autonomous agents that govern the retrieval-generation process, thereby enabling dynamic decision-making, iterative refinement, and adaptive orchestration. Agents leverage established patterns such as *reflection* (Madaan et al. 2023), *planning* (Madaan et al. 2023), *tool use*, and *multi-agent collaboration* (Guo et al. 2024) to manage retrieval strategies and integrate diverse information sources. Architectures range from single-agent systems to multi-agent and hierarchical designs, with some frameworks incorporating graph-based reasoning (*Agent-G* (Lee et al.)) for richer contextual integration. Compared to static RAG, Agentic RAG achieves greater flexibility, scalability, and contextual accuracy in domains such as healthcare, finance, and education (Singh et al. 2024).

**Training paradigm.** Post-pretraining optimization of LLMs encompasses Supervised Fine-Tuning (SFT) and Reinforcement Learning (RL). SFT leverages high-quality expert trajectories to align model behavior through imitation, but often exhibits limited generalization capability on out-of-distribution inputs or tasks demanding novel reasoning patterns not present in the training data. In contrast, RL facilitates active exploration by LLMs, enabling them to identify and reinforce superior behaviors that enhance generalization performance. RL methods for LLMs can be categorized as critic-based or critic-free. Critic-based approaches like PPO (Schulman et al. 2017) and VAPO (Yue et al. 2025) use a value function to provide fine-grained feedback during training. Critic-free methods like GRPO (Shao et al. 2024) and CISPO (Chen et al. 2025) use only sequence-level rewards, offering greater computational efficiency and scalability.

## Preliminary

### Problem Formalization

In this section, we provide a formal defination of TIR (Tool-Integrated Reasoning).

Given a task $x = (h, q)$ where $q$ is the questioin and $h$ is the related background information (e.g., historical dialogue records in dialogue system), the environment $\varepsilon$ provides access to a set of tools $\mathcal{T} = \{t_0, t_1, \dots t_n\}$, the language model $\mathcal{M}$ can proactively and iteratively interact with the environment by reasoning and calling specific tools in $\mathcal{T}$, obtaining the corresponding tool results from $\varepsilon$. This process repeats until $\mathcal{M}$ provides the final answer to $x$. This interaction process can be expressed as a multi-step trajectory $\mathbb{T}$.

The steps in $\mathbb{T}$ can be divided into two categories: intermediate tool calls and final answer. The intermediate tool call step $\tau_k$ can be defined as

$$\tau_k = (r_k, tc_k, o_k), \tag{1}$$

where $r_k$, $tc_k$ and $o_k$ denote the reasoning, tool call and returned observation respectively. The complete trajectory $\mathbb{T}$ is then defined as

$$\mathbb{T} = \tau_1 \cdot \tau_2 \cdot \ldots \tau_n \cdot \tau_{\text{final}}, \tag{2}$$

where$\tau_{\text{final}} = (r_{\text{final}}, a)$ and $a$ denotes the final answer generated by $\mathcal{M}$.

### Agentic Reinforcement Learning

Based on Section , the Agentic RL training objective is:

$$\max_{\pi_\theta} \mathbb{E}_{q\sim\mathcal{D},\mathbb{T}\sim\pi_\theta(\cdot|x;\mathcal{E})} \left[r_\phi(q, \mathbb{T})\right] - \beta \mathbb{D}_{kl} \left[\pi_\theta(\mathbb{T} \mid x; \mathcal{E}) \| \pi_{\text{ref}}(\mathbb{T} \mid x; \mathcal{E})\right], \tag{3}$$

where $\mathbb{T}$ denotes the trajectory generated by $\mathcal{M}$, $\pi_\theta$, and $\pi_{ref}$ denotes the policy model and reference model, respectively. $r_\phi$ and $D_{kl}$ denotes the returns and KL divergence respectively.

There are two well-established policy-gradient RL methods used in Agentic RL, Proximal Policy Optimization

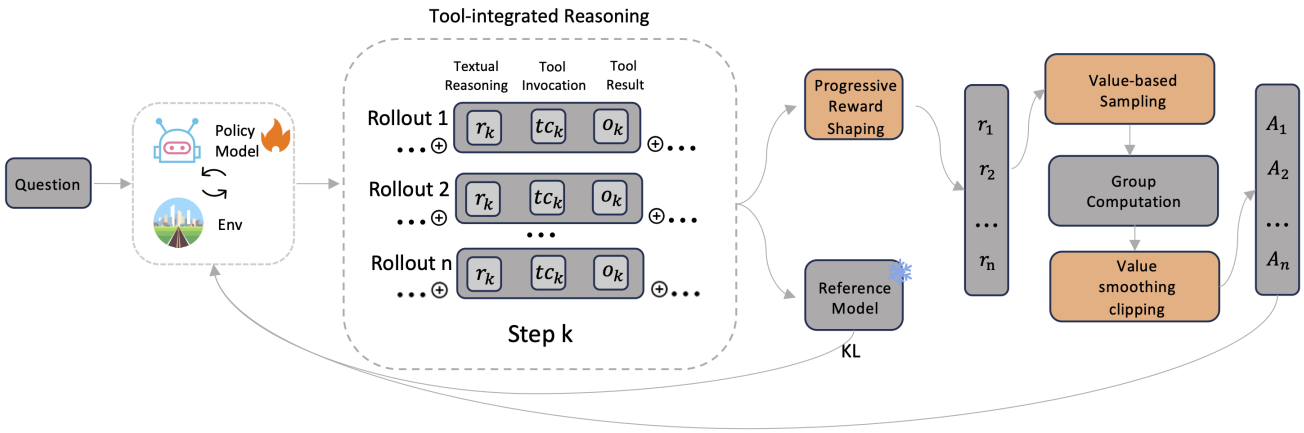



Figure 1: Overview of PRS (rogressive Reward Shaping ) and VSPO(alue-based Sampling Policy Optimization).

(PPO) (Schulman et al. 2017) and Group Relative Policy Optimization (GRPO) (Guo et al. 2025).

For PPO, based on the form of 3, the $r_\phi(q, \mathbb{T})$ of PPO is

$$r_\phi(q,\mathbb{T}) = \frac{1}{N}\sum_{i=1}^{N} \min\left(\frac{\pi_\theta\left(a_t^{(i)} \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t^{(i)} \mid s_t\right)} A\left(s_t, a_t^{(i)}\right),\right.$$
$$\left.\text{clip}\left(\frac{\pi_\theta\left(a_t^{(i)} \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t^{(i)} \mid s_t\right)}, 1-\epsilon, 1+\epsilon\right) A\left(s_t, a_t^{(i)}\right)\right), \tag{4}$$

where $a_t^{(i)} \sim \pi_{\theta_{\text{old}}}(a \mid s_t)$ is the i-th sampled token from the old policy $\pi_{\theta_{\text{old}}}$. $A_t$ is the estimated advantage given by

$$A(s_t, a_t) = \mathcal{R}(s_t, a_t) - V(s_t), \tag{5}$$

where $V_\theta(s)$ is the learned value function, which is derived from a critic network.

For GRPO, the $r_\phi(q, \mathbb{T})$ of GRPO is

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\ \{y_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|x)}$$
$$\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|y_i|}\sum_{t=1}^{|y_i|}\right.$$
$$\left.\min\left(w_{i,t}(\theta)\hat{A}_{i,t},\ \text{clip}\left(w_{i,t}(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_{i,t}\right)\right],$$
$$w_{i,t}(\theta) = \frac{\pi_\theta\left(y_{i,t} \mid x, y_{i,<t}\right)}{\pi_{\theta_{\text{old}}}\left(y_{i,t} \mid x, y_{i,<t}\right)},$$
$$\hat{A}_{i,t} = \hat{A}_i = \frac{R(x, y_i) - \text{mean}\left(\{R(x, y_j)\}_{j=1}^{G}\right)}{\text{std}\left(\{R(x, y_j)\}_{j=1}^{G}\right)}. \tag{6}$$

Here $x_i$ is the task (A.K.A prompt) and $y_i$ is the multi-step trajectory (A.K.A rollout) defined is equation 2. All tokens in $y_i$ share the same advantage as $\hat{A}_i$.

## Reward Shaping

Reward shaping belongs to the category of reward design, which is a reinforcement learning (RL) technique that augments the original reward function with additional signals to accelerate and guide the agent's learning process, while preserving the optimal policy(Ibrahim et al. 2024). In many real-world RL tasks, rewards are sparse, delayed, or noisy, making it difficult for agents to effectively explore the environment and converge to desirable behaviors. Reward shaping addresses these challenges by incorporating additional incentives or penalties that provide intermediate feedback, thereby encouraging exploration toward promising states and actions.

# Methodology

Our methodology consists of two components: Progressive Reward Shaping and Value-based Sampling.

## Progressive Reward Shaping (PRS)

We proposed progressive reward shaping (PRS) based on the following key insights: a effective reward function should exhibit (1) richer partial ordering and (2) instructive progression, both of which can significantly improve learning efficiency.

**Richer partial order.** A reward function with richer partial ordering—also described as being more discriminative or dense—can distinguish between rollouts of similar quality rather than assigning identical rewards. For example, the commonly used Reinforcement Learning with Verifiable Rewards (RLVR) (Liu et al. 2025) typically employs a binary reward of 0 and 1. Such sparse binary rewards provide limited learning signals: the model only receives "correct" or "incorrect" feedback, without finer-grained "better" or "worse" signals to guide improvement. Furthermore, for GRPO, this sparsity increases the likelihood of generating zero-advantage samples.

**More instructive.** Inspired by curriculum Learning (Soviany et al. 2022), we design the reward function to start

with simple, fundamental rewards and progressively transition into more challenging criteria. This staged approach encourages the model to first learn basic behaviors before advancing to more complex reasoning patterns, facilitating more stable and efficient learning.

Based on these principles, we present reward design framework in retrieval-augmented question-answering scenarios. Without loss of generality, this reward framework is applicable to diverse agentic RL training scenarios, and we provide a generalized formulation that can be adapted to other domains.

**Short-Form QA** Short-Form QA refers to a type of QA where the answers are typically brief and mostly consist of concrete entities (e.g. time, place and name).

The standard reward function of short-form QA is binary 0-1 reward based on EM (Exact Match) (Jin et al. 2025):

$$r_\phi(x, y) = \mathrm{EM}\left(a_{\text{pred}}, a_{\text{gold}}\right) \tag{7}$$

where $a_{\text{pred}}$ is the extracted final answer from response $y$ and $a_{\text{gold}}$ is the ground truth answer.

For TIR, model $\mathcal{M}$ must generate the correct intermediate tool call step $\tau_k$ of $\mathbb{T}$ before producing the final answer step $\tau_{\text{final}}$. We identify three progressive learning objectives:

1. Complete process: each tool call $tc_k$ in $\tau_k$ can be successfully parsed and executed to return observations $o_k$ and the final step $\tau_{\text{final}}$ can be parsed to extract the final answer $a$.
2. Complete format: the content generated at each step to be correctly placed in the corresponding tags (e.g. `<reasoning></reasoning>`)
3. Correct answer: based on $\bigcup_i o_i$, $\mathcal{M}$ generates the correct answer $a$.

We design corresponding reward components for each learning objective. We will first introduce each reward and then provide the complete PRS formulation.

**Process reward $R_{process}$.** The definition of $R_{process}$ is:

$$R_{\text{process}} = \begin{cases} -1 & \text{other,} \\ 0 & \forall tc_k \in \mathbb{T} \text{ can be parsed but } a \text{ cannot be parsed,} \\ 1 & \forall tc_k, a \in \mathbb{T},\ tc_k \text{ and } a \text{ can be correctly parsed} \end{cases} \tag{8}$$

Process reward evaluates if $\mathcal{M}$ can generate parseable tool calls and final answers.

**Format reward $R_{format}$.** The defination of $R_{format}$ is:

$$R_{format} = \begin{cases} 0 & \text{others} \\ 0.1 & \text{Complete label package} \end{cases} \tag{9}$$

The format reward evaluates whether $\mathcal{M}$ generates outputs that conforms to the required structured format.

**Answer reward $R_a$.** Unlike the binary EM reward in Eq. (7), we design a denser reward based on BLEU (Papineni et al. 2002), named short-form BLEU, which is more suitable for short-form QA.

BLEU is a widely-used metric for machine translation evaluation. It measures the n-gram overlap between a predicted translation and one or more reference texts. The definition of BLEU is

$$\begin{aligned} \mathrm{BLEU} &= \mathrm{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right), \\ \mathrm{BP} &= \begin{cases} 1, & \text{if } c > r, \\ \exp\left(1 - \frac{r}{c}\right), & \text{if } c \le r. \end{cases} \end{aligned} \tag{10}$$

where $N = 4$, $p_n$ is the modified $n$-gram precision, $w_n$ is the weight for each $n$-gram (usually $1/N$), $c$ is the candidate length, $r$ is the reference length, and $\mathrm{BP}$ is the brevity penalty.

However, standard BLEU with $N = 4$ is problematic for short-form QA. When the predicted answer length is less than 4 tokens, even an exact match results in BLEU $< 1$ due to undefined higher-order $n$-gram precisions. This artificially penalizes short answers and biases the model toward generating longer responses.

To address this issue, we propose *short-form BLEU*, which dynamically adjusts the $n$-gram weights $w_n$ according to the prediction length $c$. Specifically, let

$$\max_n = \min(N, c), \tag{11}$$

then we re-define the weights as

$$w_n = \begin{cases} \frac{1}{\max_n}, & n \le \max_n, \\ 0, & n > \max_n \end{cases} \tag{12}$$

so that only valid $n$-grams up to $\max_n$ contribute equally to the final score. This ensures that an exact match between a short predicted answer and the reference can achieve $\mathrm{BLEU} = 1$, avoiding unintended reward degradation.

**Complete Formula of PRS.** The defination of short-formal PRS is

$$PRS_{short} = \begin{cases} R_{process} + R_{format} & \text{if } R_{process} < 1 \\ R_{process} + R_{format} + R_a & \text{others} \end{cases} \tag{13}$$

Note that $R_{process} + R_{format} + R_a \ge R_{process} + R_{format}$, so PRS encourage models to first have the ability to conduct complete interactions, and then improve the quality of final answer.

**Long-Form QA** Different from short-form QA, relying solely on n-gram based metrics as answer reward is insufficient for long-form responses, as models exploit these metrics through reward hacking. To overcome this issue, we add a reward of LLM-as-a-Judge $R_{judge}$ on top of $PRS_{short}$. $R_{judge}$ mainly evaluates whether the model's answers are factually grounded and free from hallucinations. The defination of $PRS_{long}$ is

$$PRS_{long} = \begin{cases} R_{process} + R_{format}, & \text{if } R_{process} < 1, \\ R_{process} + R_{format} + R_{judge}, & \text{if } R_{process} \ge 1, \\ R_{process} + R_{format} + R_{judge} + R_a, & \text{if } R_{process} \ge 1 \text{ and } R_{judge} \ge 1 \end{cases} \tag{14}$$

$PRS_{long}$ encourage models to first generate a reasonable answer without illusions, and then maximize alignment with the reference answer.

The selection of reward weights is based on experience. $R_{process}$, $R_{judge}$ and $R_a$ are equal weights and $R_{format}$ is lower weight cause format not as important as the first three.

**The general form of PRS** PRS is not limited to QA scenarios and can serve as a design framework for various agentic RL training tasks. While $R_{\text{process}}$ and $R_{\text{format}}$ are universal components in TIR training, the framework can be extended with task-specific reward components. Let $\sigma(\cdot)$ be the sigmoid function, then the general form of PRS would be

$$\begin{aligned} R_{\text{PRS}} =& R_1 + \mathbb{I}(R_1 \geq \epsilon_1) \cdot \sigma(R_2) \\ &+ \mathbb{I}(R_1 \geq \epsilon_1, R_2 \geq \epsilon_2)\, \sigma(\cdot R_3) + \cdots . \end{aligned} \tag{15}$$

Where $\epsilon_i$ is the reward threshold to control reaching the next reward step. The sigmoid function $\sigma(\cdot)$ ensures that the cumulative reward at each stage is strictly greater than at the previous stage (i.e., $\sigma(R_i) \in (0, 1)$), thereby encouraging the model to progressively learn increasingly complex behaviors while maintaining stable gradient signals.

## VSPO (Value-based Sampling Policy Gradient)

GRPO suffers from the gradient degradation problem when all rollouts within their group have the same reward. The resulting advantages are zero, leading to zero policy gradients. This has a negative impacts:

- the gradient signal of the batch is dampened, resulting in a decrease in sample efficiency.

To address this issue, we propose Value-based Sampling Policy Optimization (VSPO), an improved variant of GRPO. Unlike the dynamic sampling approach in DAPO (Yu et al. 2025), which may be computationally intensive. VSPO replaces zero advantage samples with selectively sampling in the current batch, maintaining computational efficiency while improving sample utilization and model learning efficiency.

**The introduction of VSPO** To ensure all samples in the batch produce effective gradients, VSPO first filters out the tasks $x$ in the current batch with reward variance $Var\big(\{R(x, y_i)\}_{i=1}^{G}\big)$ less than the threshold $\varepsilon$ (here we set $\varepsilon$ as $1\text{e}{-6}$), then replaces them by resampling from the remaining samples.

Denote current batch as $B$ and the rollout numbers is $n$, $\forall x = (h, q) \in B$, . the rollout group of $x$ is $G_x = \{(y_i, r_i)\}_{i=1}^{n}$, here $y_i$ denotes the rollout and $r_i$ is the corresponding reward. Let $\mu_x$ and $\sigma_x^2$ be the mean and variance of rewards within $G_x$. That is

$$\mu_x = \frac{1}{n} \sum_{i=1}^{n} r_i \tag{16}$$

$$\sigma_x^2 = \frac{1}{n} \sum_{i=1}^{n} (r_i - \mu_x)^2 \tag{17}$$

After filters out the tasks $x$ in the current batch with reward variance $\sigma_x^2$ less than the threshold $\varepsilon$, we have $B = B^{\sigma_x^2 >= \varepsilon} \cup B^{\sigma_x^2 < \varepsilon}$ and $B^{\sigma_x^2 >= \varepsilon} \cap B^{\sigma_x^2 < \varepsilon} = \emptyset$.

A naive approach would be uniform sampling from $B^{\sigma_x^2 >= \varepsilon}$. However, we argue this can be improved. The insight of VSPO is "prioritize samples with higher learning value tasks".

**The learning value of task.** To measure the value of task $x$, we reinterpret the reward mean $\mu_x$ and varience $\sigma_x^2$ of rollouts group $G_x$ in GRPO.

- $\mu_x$ implies the average ability for $\mathcal{M}$ to solve $x$. The higher the $\mu_x$, the better $x$ is solved by $\mathcal{M}$.
- $\sigma_x^2$ reflects the policy's uncertainty in solving $x$ or the diversity of exploration. The higher the $\sigma_x^2$, the more rollouts $G_x$ contains with relatively high and low rewards, that is to say, more gradient signals it contains.

We believe that the more difficult it is for $\mathcal{M}$ to solve and the more gradient signals the task contains have richer learning value. Based on the above insights, we define the value score of task $x$ as:

$$V_x = (R_{max} - \mu_x) \cdot \sigma_x^2, \tag{18}$$

where $R_{max} = \max_{x \in B, y_i \in G_x} R(x, y_i)$ is the maximum reward of current batch $B$. $\mu_x$ and $\sigma_x^2$ is defined in equation 16 and 17.

The term $(R_{\max} - \mu_x)$ captures the difficulty of $x$ and represents the potential improvement gap between the best rollout and average performance, while $\sigma_x^2$ captures the policy's uncertainty. This formulation balances both difficulty and uncertainty, ensuring that the more difficult for $\mathcal{M}$ to solve and the more gradient signals the task contains have higher learning value.

**Value-based sampling** Based on eaquation 18, we normalize the values using a temperature-controlled Softmax:

$$p_{x_i} = \frac{\exp(V_{x_i}/T)}{\sum_k \exp(V_{x_i}/T)}, \quad T > 0, \tag{19}$$

where smaller $T$ increases preference for high value score samples, while a larger $T$ yields more uniform probabilities.

Finally, we sample $\{x_i\}$ from $B^{\sigma^2 >= \varepsilon}$ according to $\{p_{x_i}\}$ to replace the filtered samples.

Handling of some boundary situations, set current batch is $B$:

- If $\forall x \in B$ have $\sigma_x^2 > \epsilon$, which means there is no zero advantage task, then value-based sampling will not be performed. The processing flow is exactly the same as traditional GRPO.
- IF $\forall x \in B$ have $\sigma_x^2 <= \epsilon$, which means every task of $B$ is zero advantage, then value-based sampling will not be performed, too. The processing flow is exactly the same as traditional GRPO.

**Value smoothing clipping** Value-based sampling alone can cause training instability. If prompt $x_K$ is sampled $N$ times, its contribution to the objective in Eq. (20) becomes:

$$
\begin{aligned}
\mathcal{J}_{x_K}(\theta) = N \times & \sum_{\{y_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(\cdot|x_K)} \Bigg[ \frac{1}{G} \sum_{i=1}^{G} \frac{1}{|y_i|} \\
& \times \sum_{t=1}^{|y_i|} \min\Big( w_{i,t}(\theta)\, \hat{A}_{i,t}, \\
& \quad \text{clip}\big(w_{i,t}(\theta),\, 1-\epsilon,\, 1+\epsilon\big) \hat{A}_{i,t} \Big) \Bigg] \\
= & \sum_{\{y_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(\cdot|x_K)} \Bigg[ \frac{1}{G} \sum_{i=1}^{G} \frac{1}{|y_i|} \\
& \times \sum_{t=1}^{|y_i|} \min\Big( w_{i,t}(\theta) \cdot N \cdot \hat{A}_{i,t}, \\
& \quad \text{clip}\big(w_{i,t}(\theta),\, 1-\epsilon,\, 1+\epsilon\big) \cdot N \cdot \hat{A}_{i,t} \Big) \Bigg] \quad (20)
\end{aligned}
$$

This shows that sampling $x_K$ $N$ times is equivalent to scaling its advantage $\hat{A}_i$ by $N$, which can destabilize policy gradients.

To stabilize training, we further proposed value smoothing clipping:

$$
A_{\text{new}} = \left( \alpha - \frac{\alpha - 1}{N} \right) \cdot A, \quad (21)
$$

where $\alpha \in [1, +\infty]$ controls the the clipping magnitude (note that $\lim_{N \to +\infty}(\alpha - \frac{\alpha-1}{N}) = \alpha$) When $N = 1$, $A_{\text{new}} = A$; when $N > 1$, $A$ is smoothly reduced to prevent over-dominance of repeated samples in gradient updates.

**A Dynamic Advantage Weight Perspective of VSPO** VSPO can be interpreted as task-value-based reweighting rather than random sampling. Consider the general policy gradient (Sutton et al. 1999) with advantage function $A_t$:

$$
\nabla_\theta \mathcal{J}(\theta) = \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_\theta} \left[ \sum_{t=1}^{T} \nabla_\theta \pi_\theta \left( y_t \mid y_{<t} \right) A_t \right]. \quad (22)
$$

In GRPO, all tokens in the same trajectory share the same advantage, so:

$$
\nabla_\theta \mathcal{J}(\theta) = \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_\theta} \left[ \sum_{t=1}^{T} \nabla_\theta \pi_\theta \left( y_t \mid y_{<t} \right) \right] \cdot A_x. \quad (23)
$$

Let $\omega = \left(\alpha - \frac{\alpha-1}{N}\right)$, then

$$
\nabla_\theta \mathcal{J}(\theta) = \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_\theta} \left[ \sum_{t=1}^{T} \nabla_\theta \pi_\theta \left( y_t \mid y_{<t} \right) \right] \cdot \omega_x \cdot A_x. \quad (24)
$$

Thus, tasks with higher learning value receive larger advantage weights $\omega_x \cdot A_x$, effectively increasing the update step size for these tasks and amplifying policies with greater relative advantages.

# Experimental Setup

To evaluate the effectiveness of PRS and VSPO in training LLM-based TIR agent, we conduct experiments on the following two types of long-horizon reasoning tasks: short-form QA and long-form QA. In these tasks, the agent has access to at least one information retrieval tool to answer the question.

In short-form QA, we use Qwen2.5-3B-Instruct (Yang et al. 2024) and in long-form QA, we use the Qwen3-14B (Team 2025) as the training model. All of them uses full parameter training.

We adopt Trinity-RFT (Pan et al. 2025) as our training framework.

For VSPO, we set $T = 0.1$ and $\alpha = 2$. The rollout num of VSPO, CISPO and GRPO is 5. The learning rate of all algorithms is 1e-7. Training is performed on a single node with 8 H20 GPUs (141GB). To optimize GPU memory usage, we enable gradient checkpointing and use Fully Sharded Data Parallel (FSDP) with CPU offloading.

We also use gradient checkpointing, FSDP offloading, and vLLM-based rollouts with the same hyperparameters as above. The rollout temperature and top-p values are both set to 1.0, and the KL divergence coefficient $\beta$ and clip ratio $\epsilon$ are fixed at 0.001 and 0.2.

## Datasets

**Long-Form QA.** We use proprietary data from our production system. We select conversation records from the online conversation database as RL rollout data. There are three types of queries in total:

- $Q_{simple}$: A plain text query containing a single question.
- $Q_{multim}$: A multimodal query containing both images and text.
- $Q_{multiq}$: A query containing multiple sub-questions.

The number of training and testing sets is 1678 and 334, respectively. We manually annotate reference answers based on our internal knowledge base.

For expert data used in SFT, we use knowledge distillation from the Qwen3-235B-A22B (Team 2025) to construct high-quality trajectory data. Specifically, we collect online service logs from Qwen3-235B-A22B and apply rule-based filtering and LLM-as-a-Judge cleaning to produce expert demonstrations.

**Short-Form QA.** Following (Jin et al. 2025), we use seven benchmark datasets: (1) General Question Answering: NQ (Kwiatkowski et al. 2019), TriviaQA (Joshi et al. 2017), and PopQA (Mallen et al. 2023). (2) Multi-Hop Question Answering: HotpotQA (Yang et al. 2018), 2WikiMultiHopQA (Ho et al. 2020), Musique (Trivedi et al. 2022), and Bamboogle (Press et al. 2023) as Short-Form QA train and test data.

We merge the training sets of NQ and HotpotQA as our training set and evaluation is conducted on the test or validation sets of all seven datasets to assess both in-domain and out-of-domain performance.

| Qwen3-14B | | | | |
|---|---|---|---|---|
| Methods | $Q_{simple}$ | $Q_{multiq}$ | $Q_{multim}$ | Avg. |
| **Reference Answer Pass Rate** | | | | |
| Untrained | 0.5513 | 0.6999 | 0.5000 | 0.5897 |
| GRPO + 0-1 reward | 0.6955 | 0.8017 | 0.6000 | 0.7255 |
| GRPO + $PRS_{long}$ | 0.7144 | 0.8183 | 0.7150 | 0.7300 |
| **Hallucination Pass Rate** | | | | |
| Untrained | 0.7821 | 0.8167 | 0.7143 | 0.7893 |
| GRPO + 0-1 reward | 0.8782 | 0.9333 | 0.8600 | 0.8925 |
| GRPO + $PRS_{long}$ | 0.8982 | 0.9333 | 0.9000 | 0.9077 |

Table 1: Results on the test set of 0-1 reward and PRS for long-form QA.

### Evaluation Metrics

**Long-Form QA.** We use GPT-4.1 as an LLM judge to determine whether the responses contains hallucinations and whether it matches the annotated reference answers. The evaluation prompts are provided in Appendix .

**Short-Form QA.** We use `Exact Match` (EM) as the evaluation metric, computed against the golden answers provided by each benchmark dataset.

### Baselines

To verify the effectiveness of PRS, we compare it with rule-based 0-1 reward to demonstrate the effectiveness of PRS. All reward functions use GRPO as the RL algorithm. For long-form QA, the 0-1 reward use Qwen3-235B-A22B-Instruct-2507 (Team 2025) as the verifier, determine whether the model output passes based on the annotated reference answers, if passed, the reward is 1, otherwise it is 0. For short-form QA, using exact match as rule-based 0-1 reward.

To verify the effectiveness of VSPO, we compare it with GRPO, PPO (Schulman, Chen, and Abbeel 2017) and SFT-only. All RL algorithms use the same PRS reward.

## Main Results

### The main results of PRS

The main results comparing PRS with 0-1 reward are presented in Table 1 for long-form QA and Table 2 for short-form QA. Under the same RL algorithm, we can found that compared to 0-1 reward, PRS achieved better results. In long-form OA, $PRS_{long}$ has increased reference answer pass rate by $0.62\%$ and hallucination Pass Rate by $1.7\%$ compared to 0-1 reward. In short-form OA, $PRS_{short}$ has increased exact match score by $2.74\%$ compared to 0-1 reward.

### The main results of VSPO

The main results comparing VSPO with baseline methods across are presented in Table 3 for long-form QA and Table 4 for short-form QA. Under the same reward function, we can found that compared to other RL algorithm, VSPO achieved better results. In long-form QA, VSPO has increased reference answer pass rate by $5.34\%$ compared to GRPO. In short-form OA, $PRS_{short}$ has increased exact match score by $3.16\%$ compared to GRPO.

## Analysis

We analyze some issues based on long-form QA and short-form QA.

### VSPO v.s. Other RL Algorithms

The training reward and test set metric of both RL algorithm are show in Firure 2.

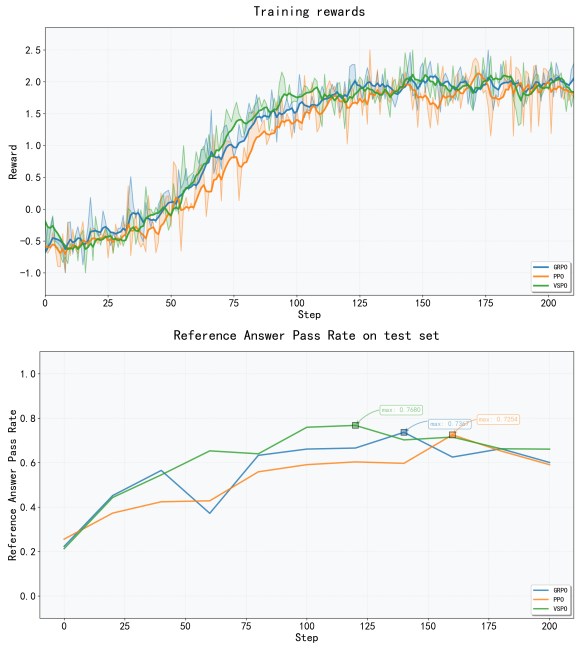



Figure 2: Reward curves on the training and test set metric(Reference Answer Pass Rate) for RL algorithm in long-form QA.

VSPO achieved the best performance based on rewards on the train set and reference answer pass rate on the test sets. As the number of training steps increases, PPO gradually converges to the same level as VSPO and GRPO on training set.

Among all RL algorithms, VSPO was the first to receive the best test metric performance, this indicates that filling in tasks that cannot be helpful for learning through value-based sampling makes the model more focused on high-value tasks, which helps to increase convergence speed and achieve better performance.

### The Effect of $T$ and $\alpha$ of VSPO

We analyze the effect of $T$ and $\alpha$ of VSPO. Based on equation 19, We can analyze that $T$ controlling the sharpness of value distribution. If $T$ tends towards 0, the value-based sampling will be concentrated in high learning value tasks. If $T$ tends towards positive infinity, the value-based sampling

| Qwen2.5-3B-Instruct | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Methods | NQ | TriviaQA | PopQA | HotpotQA | 2wiki | Musique | Bamboogle | Avg. |
| **Exact Match (EM)** | | | | | | | | |
| Untrained | 0.0415 | 0.1751 | 0.0849 | 0.0622 | 0.0891 | 0.0082 | 0.1667 | 0.0960 |
| GRPO + 0-1 reward | 0.3213 | 0.5022 | 0.3457 | 0.3000 | 0.2586 | 0.1120 | 0.0833 | 0.3390 |
| GRPO + $PRS_{short}$ | 0.3518 | 0.4987 | 0.3562 | 0.3100 | 0.2600 | 0.1245 | 0.3333 | 0.3483 |

Table 2: Results on the test set of 0-1 reward and PRS for short-form QA.

| Qwen3-14B | | | | |
|---|---|---|---|---|
| Methods | $Q_{simple}$ | $Q_{multiq}$ | $Q_{multim}$ | Avg. |
| **Reference Answer Pass Rate** | | | | |
| Untrained | 0.5513 | 0.6999 | 0.5000 | 0.5897 |
| SFT | 0.6405 | 0.7500 | 0.6000 | 0.6687 |
| GRPO + $PRS_{long}$ | 0.7144 | 0.8183 | 0.7150 | 0.7300 |
| PPO + $PRS_{long}$ | 0.700 | 0.8213 | 0.7020 | 0.7327 |
| VSPO + $PRS_{long}$ | 0.7147 | 0.9000 | 0.8557 | 0.7690 |
| **Hallucination Pass Rate** | | | | |
| Untrained | 0.7821 | 0.8167 | 0.7143 | 0.7893 |
| SFT | 0.8079 | 0.8167 | 0.7556 | 0.8086 |
| GRPO + $PRS_{long}$ | 0.8982 | 0.9333 | 0.9000 | 0.9077 |
| PPO + $PRS_{long}$ | 0.8282 | 0.8633 | 0.9000 | 0.8399 |
| VSPO + $PRS_{long}$ | 0.8662 | 0.9000 | 0.8571 | 0.8750 |

Table 3: Results on the test set of RL algorithm for long-form QA.

will become flat and uniform and weaken the gap in learning value of tasks.

$\alpha$ controlls the maximum amplitude of sample update. The larger the value of $\alpha$, the more radical updates are allowed, while the smaller the value, the more conservative updates are allowed.

The results of different $\alpha$ and $T$ are shown in Table 5 and the reference answer pass rate curve of test set is shown in Figure 3.

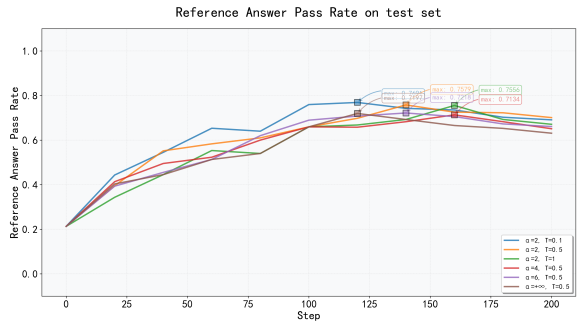



Figure 3: Reference Answer Pass Rate of test set of different $\alpha$ and $T$.

From Table 5 and Figure 3, We can analyze that, in VSPO:

- When $\alpha$ is fixed, a lower $T$ will bring better results and as $T$ decreases, the best result will be achieved earlier. This is because a lower $T$ allows value-based sampling to be repeated more times on more valuable tasks, which allows more valuable tasks to have a relatively larger gradient modulus to update themselves.
- When $T$ is fixed, raising $\alpha$ did not bring any improvement in effectiveness and $\alpha = +\infty$ (means no value smoothing clipping, corresponding to no clip) lead to a decrease in the effect. This is because the main function of $\alpha$ is to limit the changes in advantages in a smooth manner. If the magnitude of $\alpha$'s changes is too large, it may lead to unstable updates and reduce the effectiveness of training.

## The Effect of PRS

Not limited to the metric results shown in Table 1 and Table 2 on the test set, we will further analyze the effect of PRS.

**PRS can reduce the proportion of zero advantage tasks in batch.** Figure 4 shows the proportion of zero advantage tasks at each training step when using 0-1 reward and $PRS_{long}$ (The RL algorithm is GRPO).

We can see that, when using 0-1 reward, about 60% of tasks for batch in each step do not bring any learning signals. For PRS, based on the summation of various components, it can generate rich partial order, so the proportion of zero advantage tasks is greatly reduced.

Figure 4: Proportion of zero advantage tasks when training when using 0-1 reward and PRS.

**PRS can help policy explore strategies more efficiently.** Figure 5 shows the entropy loss of VSPO when using 0-1 reward and PRS as training reward. Entropy loss reflects

| Qwen2.5-3B-Instruct | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Methods | NQ | TriviaQA | PopQA | HotpotQA | 2wiki | Musique | Bamboogle | Avg. |
| **Exact Match (EM)** | | | | | | | | |
| Untrained | 0.0415 | 0.1751 | 0.0849 | 0.0622 | 0.0891 | 0.0082 | 0.1667 | 0.0960 |
| SFT | 0.2490 | 0.2920 | 0.1040 | 0.1860 | 0.2480 | 0.0440 | 0.1120 | 0.1760 |
| GRPO + $PRS_{short}$ | 0.3518 | 0.4987 | 0.3562 | 0.3100 | 0.2600 | 0.1245 | 0.3333 | 0.3483 |
| PPO + $PRS_{short}$ | 0.3210 | 0.4887 | 0.3562 | 0.3200 | 0.2650 | 0.1200 | 0.3325 | 0.3333 |
| VSPO + $PRS_{short}$ | 0.3600 | 0.5000 | 0.3634 | 0.3100 | 0.2650 | 0.1367 | 0.3489 | 0.3593 |

Table 4: Results on the test set of RL algorithm for short-form QA.

| Qwen3-14B | | | | |
|---|---|---|---|---|
| Methods | $Q_{simple}$ | $Q_{multiq}$ | $Q_{multim}$ | Avg. |
| **Reference Answer Pass Rate** | | | | |
| Untrained | 0.5513 | 0.6999 | 0.5000 | 0.5897 |
| $\alpha = 2, T = 1$ | 0.7019 | 0.8833 | 0.8571 | 0.7556 |
| $\alpha = 2, T = 0.5$ | 0.7276 | 0.8250 | 0.8571 | 0.7579 |
| $\alpha = 2, T = 0.1$ | 0.7147 | 0.9000 | 0.8557 | 0.7690 |
| $\alpha = 4, T = 0.5$ | 0.6923 | 0.875 | 0.8571 | 0.7134 |
| $\alpha = 6, T = 0.5$ | 0.7051 | 0.8750 | 0.8571 | 0.7218 |
| no clip, $T = 0.5$ | 0.6955 | 0.8917 | 0.996 | 0.7197 |
| **Hallucination Pass Rate** | | | | |
| Untrained | 0.7821 | 0.8167 | 0.7143 | 0.7893 |
| $\alpha = 2, T = 1$ | 0.8782 | 0.8667 | 0.8658 | 0.8747 |
| $\alpha = 2, T = 0.5$ | 0.8846 | 0.8167 | 0.8571 | 0.8655 |
| $\alpha = 2, T = 0.1$ | 0.8662 | 0.9000 | 0.8571 | 0.8750 |
| $\alpha = 4, T = 0.5$ | 0.8782 | 0.9167 | 0.7143 | 0.8494 |
| $\alpha = 6, T = 0.5$ | 0.8526 | 0.8667 | 0.4286 | 0.8177 |
| no clip, $T = 0.5$ | 0.8526 | 0.8333 | 0.8714 | 0.8075 |

Table 5: Results on the test set of different $\alpha$ and $T$ of VSPO for long-form QA.

the uncertainty of policy model in selecting the current action. We observe that PRS reduces uncertainty to a lower level earlier than the 0-1 reward and achieves better test set performance. This is not entropy collapse (which would manifest as degraded test metrics), but rather efficient convergence. Therefore, we infer that PRS, as a curriculum-learning-inspired reward design, effectively guides the policy's exploration and enables faster convergence to high-quality strategies. In contrast, the 0-1 reward leads to inefficient exploration under sustained high uncertainty.

**PRS has better improvement on weaker models** Adding Qwen3-8B to long-form QA and Qwen2.5-7B-Instruct to short-form QA, the results comparing PRS with 0-1 reward are presented in Table 6 for long-form QA and Table 7 for short-form QA.

In long-form OA, $PRS_{long}$ has increased reference answer pass rate by $1.27\%$ ($0.62\%$ for Qwen3-14b) and hallu-

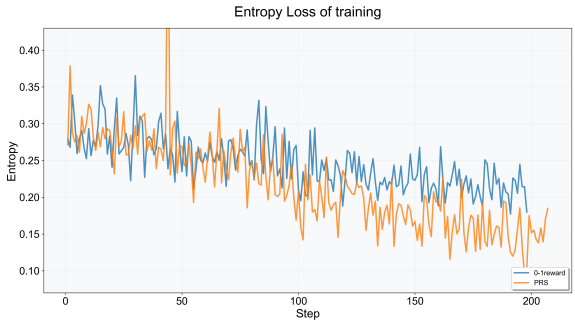



Figure 5: Entropy loss of training using 0-1 reward and PRS.

| Qwen3-8B | | | | |
|---|---|---|---|---|
| Methods | $Q_{simple}$ | $Q_{multiq}$ | $Q_{multim}$ | Avg. |
| **Reference Answer Pass Rate** | | | | |
| Untrained | 0.5353 | 0.1083 | 0.3571 | 0.4148 |
| GRPO + 0-1 reward | 0.6731 | 0.1000 | 0.6429 | 0.5180 |
| GRPO + $PRS_{long}$ | 0.6506 | 0.1917 | 0.5714 | 0.5246 |
| **Hallucination Pass Rate** | | | | |
| Untrained | 0.7372 | 0.1000 | 0.7143 | 0.5650 |
| GRPO + 0-1 reward | 0.8974 | 0.1167 | 0.8971 | 0.6873 |
| GRPO + PRS | 0.8654 | 0.7667 | 0.8571 | 0.8386 |

Table 6: Results on the test set of 0-1 reward and $PRS_{long}$ for long-form QA.

cination Pass Rate by $22.01\%$ ($1.7\%$ for Qwen3-14b) compared to 0-1 reward. In short-form QA, $PRS_{short}$ has increased exact match score by $2.74\%$ compared to 0-1 reward ($1.08\%$ for Qwen2.5-7b-Instruct). It can be seen that PRS brings more improvements to weaker models. We believe that the reason is that weak models have weaker exploration ability in the process of RL compared to strong models, so the lack of a guiding reward function (such as 0-1 reward) is a difficult learning signal for weak models. On the contrary, PRS based on curriculum learning gradually decomposes complex tasks through reward settings, which helps weak models gradually explore and improve strategy performance.

| Qwen2.5-7B-Instruct | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Methods | NQ | TriviaQA | PopQA | HotpotQA | 2wiki | Musique | Bamboogle | Avg. |
| **Exact Match (EM)** | | | | | | | | |
| Untrained | 0.2770 | 0.4872 | 0.3289 | 0.2797 | 0.2411 | 0.1037 | 0.2320 | 0.3210 |
| GRPO + 0-1 reward | 0.3500 | 0.5438 | 0.3754 | 0.3600 | 0.3095 | 0.1286 | 0.2996 | 0.3855 |
| GRPO + $PRS_{short}$ | 0.3573 | 0.5623 | 0.3815 | 0.3757 | 0.3174 | 0.0996 | 0.3333 | 0.3897 |

Table 7: Results on the test set of 0-1 reward and $PRS_{short}$ for short-form QA.

### Ablation Analysis of Value-based Sampling and Value Smoothing Clipping

To evaluate the effectiveness of value-based sampling and value smoothing clipping of VSPO, we conduct experiments using no sample without clip (equivalent to GRPO), random sample without clip, value-based sample without clip, random sample with clip and value-based sample with clip (equivalent to VSPO), respectively. The result on the test set is shown in Table 8. Table 8 demonstrates that removing either value-based sampling or value smoothing clipping significantly degrades performance, confirming the necessity of both components. Notably, any sampling strategy without clipping leads to a decrease in performance, while value-based sampling with clipping achieves the best performance.

| Qwen3-14B | | | | |
|---|---|---|---|---|
| Methods | $Q_{simple}$ | $Q_{multiq}$ | $Q_{multim}$ | Avg. |
| **Reference Answer Pass Rate** | | | | |
| w/o sample w/o clip (GRPO) | 0.7144 | 0.8183 | 0.7150 | 0.7300 |
| random sample w/o clip | 0.4120 | 0.1600 | 0.5400 | 0.3100 |
| value-based sample w/o clip | 0.6955 | 0.8917 | 0.9960 | 0.7197 |
| random sample w clip | 0.6322 | 0.5790 | 0.7150 | 0.6335 |
| value-based sample w clip (VSPO) | 0.7147 | 0.9000 | 0.8557 | 0.7690 |
| **Hallucination Pass Rate** | | | | |
| w/o sample w/o clip (GRPO) | 0.8982 | 0.9333 | 0.9000 | 0.9077 |
| random sample w/o clip | 0.5368 | 0.2300 | 0.5734 | 0.4149 |
| value-based sample w/o clip | 0.8526 | 0.8333 | 0.8714 | 0.8075 |
| random sample w clip | 0.7950 | 0.6333 | 0.8876 | 0.8256 |
| value-based sample w clip (VSPO) | 0.8662 | 0.9000 | 0.8571 | 0.8750 |

Table 8: Results on the test set of different ablation settings for long-form QA.

We further analyze the impact of value-based sampling and value smoothing clipping through KL divergence on train set and reference answer pass rate on test set. Figure 6 presents these metrics across different configurations. Two key observations emerge:

(1) **Value smoothing clipping stabilizes training.** When using random sampling, without clipping causes random sampling abnormally high KL divergence and low metric on test set, which leading to extremely unstable training. With clip can change the low performance of random sampling to some extent. When using value-based sampling, without clipping will have a larger KL loss and lower reference answer pass rate on test set than with clip. The above information can verify the effectiveness of value smoothing clipping. By limiting the changes in advantages in a smooth manner, value smoothing clipping stabilizes training while improving sampling performance.

(2) **Value-based sampling is necessary for performance gains.** When using value smoothing clipping, value-based sampling have a higher reference answer pass rate on test set on random sampling. When not using value smoothing clipping, random sampling have abnormally high KL divergence and lower metric on test set than value-based sampling. This confirming the necessity of intelligent sample selection based on learning value.

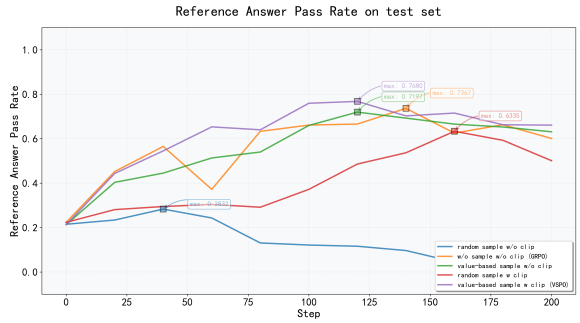
## Conclusions

In this work, we formalized the Tool-Integrated Reasoning (TIR) problem and proposed two complementary techniques to improve Agentic RL in complex reasoning tasks: *Progressive Reward Shaping* (PRS) and *Value-based Sampling Policy Optimization* (VSPO).

PRS introduces a curriculum-based reward design that provides dense, stage-wise learning signals, enabling models to first master essential capabilities—such as generating parseable outputs and properly formatted responses—before optimizing for challenging objectives like answer quality and factual grounding. We instantiated PRS in both short-form and long-form QA, with the short-form version incorporating a length-aware BLEU to avoid unfair penalization of correct short answers, and the long-form version incorporating an LLM-as-a-Judge signal to address reward hacking in complex outputs.

To address gradient degradation in GRPO when rollouts have identical rewards, we developed VSPO, which uses value-based sample to prioritize updates on tasks with higher learning value and applies value smoothing clipping to stabilize policy gradients. This combination ensures that policy updates focus on the most informative samples while preventing destabilization from repeated high-value prompts.

(a) KL Loss




(b) Reference Answer Pass Rate on test set

Figure 6: The KL Loss and Reference Answer Pass Rate on test set.

Extensive experiments on multiple short-form and long-form QA benchmarks demonstrate that PRS consistently outperforms traditional binary rewards, and VSPO achieves more stable training, better sample efficiency and superior performance across policy optimization algorithms. Overall, this work provides a generalizable reward shaping and policy optimization in Agentic RL with TIR, applicable beyond QA tasks to any multi-step reasoning scenarios.

## References


Chen, A.; Li, A.; Gong, B.; Jiang, B.; Fei, B.; Yang, B.; Shan, B.; Yu, C.; Wang, C.; Zhu, C.; et al. 2025. MiniMax-M1: Scaling Test-Time Compute Efficiently with Lightning Attention. *arXiv preprint arXiv:2506.13585*.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Guo, T.; Chen, X.; Wang, Y.; Chang, R.; Pei, S.; Chawla, N. V.; Wiest, O.; and Zhang, X. 2024. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*.

Ho, X.; Nguyen, A.-K. D.; Sugawara, S.; and Aizawa, A. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*.

Huang, X.; Liu, W.; Chen, X.; Wang, X.; Wang, H.; Lian, D.; Wang, Y.; Tang, R.; and Chen, E. 2024. Understanding the planning of LLM agents: A survey. *arXiv preprint arXiv:2402.02716*.

Ibrahim, S.; Mostafa, M.; Jnadi, A.; Salloum, H.; and Osinenko, P. 2024. Comprehensive overview of reward engineering and shaping in advancing reinforcement learning applications. *IEEE Access*.

Jin, B.; Zeng, H.; Yue, Z.; Yoon, J.; Arik, S.; Wang, D.; Zamani, H.; and Han, J. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Joshi, M.; Choi, E.; Weld, D. S.; and Zettlemoyer, L. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7: 453–466.

Lee, M.-C.; Zhu, Q.; Mavromatis, C.; Han, Z.; Adeshina, S.; Ioannidis, V. N.; Rangwala, H.; and Faloutsos, C. ???? Agent-G: An Agentic Framework for Graph Retrieval Augmented Generation.

Lin, H.; and Xu, Z. 2025. Understanding tool-integrated reasoning. *arXiv preprint arXiv:2508.19201*.

Liu, K.; Yang, D.; Qian, Z.; Yin, W.; Wang, Y.; Li, H.; Liu, J.; Zhai, P.; Liu, Y.; and Zhang, L. 2025. Reinforcement learning meets large language models: A survey of advancements and applications across the llm lifecycle. *arXiv preprint arXiv:2509.16679*.

Madaan, A.; Tandon, N.; Gupta, P.; Hallinan, S.; Gao, L.; Wiegreffe, S.; Alon, U.; Dziri, N.; Prabhumoye, S.; Yang, Y.; et al. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36: 46534–46594.

Mallen, A.; Asai, A.; Zhong, V.; Das, R.; Khashabi, D.; and Hajishirzi, H. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 9802–9822.

Pan, X.; Chen, Y.; Chen, Y.; Sun, Y.; Chen, D.; Zhang, W.; Xie, Y.; Huang, Y.; Zhang, Y.; Gao, D.; et al. 2025. Trinity-rft: A general-purpose and unified framework for reinforcement fine-tuning of large language models. *arXiv preprint arXiv:2505.17826*.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 311–318.

Press, O.; Zhang, M.; Min, S.; Schmidt, L.; Smith, N. A.; and Lewis, M. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 5687–5711.

Qian, C.; Acikgoz, E. C.; He, Q.; Wang, H.; Chen, X.; Hakkani-Tür, D.; Tur, G.; and Ji, H. 2025. Toolrl: Reward is all tool learning needs. *arXiv preprint arXiv:2504.13958*.

Ravuru, C.; Sakhinana, S. S.; and Runkana, V. 2024. Agentic retrieval-augmented generation for time series analysis. *arXiv preprint arXiv:2408.14484*.

Schick, T.; Dwivedi-Yu, J.; Dessì, R.; Raileanu, R.; Lomeli, M.; Hambro, E.; Zettlemoyer, L.; Cancedda, N.; and Scialom, T. 2023. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36: 68539–68551.

Schulman, J.; Chen, X.; and Abbeel, P. 2017. Equivalence between policy gradients and soft q-learning. *arXiv preprint arXiv:1704.06440*.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y.; Wu, Y.; et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Singh, A.; Ehtesham, A.; Mahmud, S.; and Kim, J.-H. 2024. Revolutionizing mental health care through langchain: A journey with a large language model. In *2024 IEEE 14th Annual Computing and Communication Workshop and Conference (CCWC)*, 0073–0078. IEEE.

Soviany, P.; Ionescu, R. T.; Rota, P.; and Sebe, N. 2022. Curriculum learning: A survey. *International Journal of Computer Vision*, 130(6): 1526–1565.

Sutton, R. S.; McAllester, D.; Singh, S.; and Mansour, Y. 1999. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12.

Team, Q. 2025. Qwen3 Technical Report. arXiv:2505.09388.

Trivedi, H.; Balasubramanian, N.; Khot, T.; and Sabharwal, A. 2022. MuSiQue: Multihop Questions via Single-hop Question Composition. *Transactions of the Association for Computational Linguistics*, 10: 539–554.

Wei, Y.; Yu, X.; Weng, Y.; Pan, T.; Li, A.; and Du, L. 2025. Autotir: Autonomous tools integrated reasoning via reinforcement learning. *arXiv preprint arXiv:2507.21836*.

Wu, M.; Yang, J.; Jiang, J.; Li, M.; Yan, K.; Yu, H.; Zhang, M.; Zhai, C.; and Nahrstedt, K. 2025. VTool-R1: VLMs Learn to Think with Images via Reinforcement Learning on Multimodal Tool Use. *arXiv preprint arXiv:2505.19255*.

Yang, A.; Yang, B.; Hui, B.; Zheng, B.; Yu, B.; Zhou, C.; Li, C.; Li, C.; Liu, D.; Huang, F.; Dong, G.; Wei, H.; Lin, H.; Tang, J.; Wang, J.; Yang, J.; Tu, J.; Zhang, J.; Ma, J.; Xu, J.; Zhou, J.; Bai, J.; He, J.; Lin, J.; Dang, K.; Lu, K.; Chen, K.; Yang, K.; Li, M.; Xue, M.; Ni, N.; Zhang, P.; Wang, P.; Peng, R.; Men, R.; Gao, R.; Lin, R.; Wang, S.; Bai, S.; Tan, S.; Zhu, T.; Li, T.; Liu, T.; Ge, W.; Deng, X.; Zhou, X.; Ren, X.; Zhang, X.; Wei, X.; Ren, X.; Fan, Y.; Yao, Y.; Zhang, Y.; Wan, Y.; Chu, Y.; Liu, Y.; Cui, Z.; Zhang, Z.; and Fan, Z. 2024. Qwen2 Technical Report. *arXiv preprint arXiv:2407.10671*.

Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W.; Salakhutdinov, R.; and Manning, C. D. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, 2369–2380.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K. R.; and Cao, Y. 2022. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.

Yu, Q.; Zhang, Z.; Zhu, R.; Yuan, Y.; Zuo, X.; Yue, Y.; Dai, W.; Fan, T.; Liu, G.; Liu, L.; et al. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

Yue, Y.; Yuan, Y.; Yu, Q.; Zuo, X.; Zhu, R.; Xu, W.; Chen, J.; Wang, C.; Fan, T.; Du, Z.; et al. 2025. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*.

Zhang, G.; Geng, H.; Yu, X.; Yin, Z.; Zhang, Z.; Tan, Z.; Zhou, H.; Li, Z.; Xue, X.; Li, Y.; et al. 2025. The landscape of agentic reinforcement learning for llms: A survey. *arXiv preprint arXiv:2509.02547*.

Zhao, W. X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2).

## Prompt Template for QA Agent

```
Answer the given question. You must conduct reasoning
      inside <reasoning> and </reasoning> first every
      time you get new information. After reasoning,
     if you find you lack some knowledge, you can
     call wiki_search, and it will return the top
     searched results. You can search as many times
     as you want. If you find no further external
     knowledge needed, you can directly provide the
     answer inside <answer> and </answer> without
     detailed illustrations. For example, <answer>
     xxx </answer>.
Question: {query}
```

## Prompt Template for LLM-as-aJudge Evaluation

```
Role and Background
You are an experienced AI capability evaluation
     expert, with rigorous logic and keen observation
      skills, highly adept at assessing whether AI
     responses meet the requirements.
Now, an online travel platform has launched a Q&A
     Agent that provides answers to merchants'
     questions.

Your Task
You will receive the following information:
[Merchant's Question]: The question raised by the
     merchant.
[Q&A Agent's Response]: The answer provided by the Q&
     A Agent.
[Reference Answer]: The standard answer corresponding
      to the merchant's question.
You need to, strictly based on the [Evaluation
     Criteria], give an evaluation of the [Q&A Agent'
     s Response].
You must strictly follow the internal step-by-step
     thinking process, comprehensively analyze and
     evaluate the [Q&A Agent's Response] based on the
      provided relevant information, and output the
     evaluation result in the specified format.

Your Internal Thinking Process (Please carry out
     internally, do not output)
Follow the steps below in your mind, analyzing step
     by step:

Analyze [Evaluation Criteria]
Carefully read the Evaluation Criteria, understand
     the core evaluation question and the
     requirements of each criterion.

Analyze [Merchant's Question]
Analyze and understand the Merchant's Question. Some
     questions may contain multiple sub-questions,
     others may include images (in URL form). If
     there are multiple questions, you need to
     understand the meaning of each.

Analyze [Q&A Agent's Response]
Analyze and understand the Q&A Agent's Response. If
     the Merchant's Question contains multiple sub-
     questions, the Q&A Agent's Response may contain
     multiple answers. You need to understand each
     answer, extract the information inside,
     determine which question it answers, and check
     if the model responded to all the questions.

Analyze [Reference Answer]
Analyze and understand the Reference Answer. If the
     Merchant's Question contains multiple sub-
     questions, the Reference Answer will have
     answers to each question, and sometimes the
     order may differ from the order of questions in
     the Merchant's Question. You need to first
     understand each answer, extract the information
     inside, match it with the corresponding question
     , and know the reference answer for each.
```

```
Perform Evaluation
Based on the above analysis results, use the [
     Evaluation Criteria] and [Reference Answer] to
     analyze and evaluate the [Q&A Agent’s Response].

Evaluation Criteria
Core evaluation question: Does the [Q&A Agent’s
     Response] match the [Reference Answer]?
Evaluation criteria:

Match: The Agent’s response must be semantically
     consistent with the Reference Answer. If the
     Merchant’s Question contains multiple sub-
     questions, the Q&A Agent’s Response must include
      an answer to each question, and each must be
     semantically consistent with the corresponding
     Reference Answer.
Mismatch: If the above "Match" criterion is not met,
     then it is judged as "Mismatch". That is, the Q&
     A Agent’s Response has significant deviations
     from the Reference Answer. For example, when the
      merchant asks multiple questions, the Q&A Agent
     ’s Response only answers some of them, etc.
Output Format Requirement
You must strictly follow the format below when
     outputting, and directly provide the evaluation
     rating without any extra explanation or comments
     .
<Evaluation Rating>:
```