A Rolling Stone Gathers No Moss: Adaptive Policy Optimization for Stable Self-Evaluation in Large Multimodal Models

Wenkai Wang[1], **Hongcan Guo**[2], **Zheqi Lv**[1], **Shengyu Zhang**[1*]

[1] Zhejiang University
[2] Beijing University of Posts and Telecommunications
22551237@zju.edu.cn, ai.guohc@bupt.edu.cn, zheqilv@zju.edu.cn, sy_zhang@zju.edu.cn,

Abstract

Self-evaluation, a model's ability to assess the correctness of its own output, is crucial for Large Multimodal Models (LMMs) to achieve self-improvement in multi-turn conversations, yet largely absent in foundation models. Recent work has employed reinforcement learning (RL) to enhance self-evaluation; however, its fixed reward mechanism suffers from reward hacking when optimizing multiple training objectives, leading to model collapse. In this paper we propose AdaPO, an online reinforcement learning framework capable of adaptively adjusting training objective in real time according to the current training state for each task. Specifically, to mitigate reward hacking , AdaPO introduces an Adaptive Reward Model (ARM) and a Reward Aware Dynamic KL Regularization mechanism. ARM assesses the task's training state from the distribution of model generated multi-turn trajectories' performance. Reward Aware Dynamic KL replaces a fixed penalty with dynamic coefficients which is modulated by the reward gap between different multi-turn situations. Notably, our method automatically and smoothly adjusts its learning focus based on sub-tasks' training progress without manual intervention. Extensive experiments over 8 benchmarks and various models show that our method significantly enhances both direct reasoning and self-evaluation capability.

Introduction

Large Multimodal Models (LMMs), such as Qwen2.5-VL(Bai et al. 2025), have demonstrated remarkable capabilities in complex visual-language reasoning tasks, largely enabled by Chain-of-Thought (CoT)(Wei et al. 2022) reasoning. How to achieve self-improvement(Qu et al. 2024; Saunders et al. 2022) of model responses remains a critical challenge. A promising yet challenging way is to empower LMMs with self-evaluation capability, where the model evaluates and refines its own CoT reasoning in multi-turn conversations to reach a more accurate final answer.

However, without external ground truth signals, current models struggle to self-improve via self-evaluation. As illustrated in Figure 1, for models trained with supervised fine-tuning (SFT) or reinforcement learning (RL) techniques, prompting for self-evaluation often backfires, leading to a significant degradation in accuracy rather than an improvement. The failed evaluation does not necessarily stem from a lack of external knowledge(Zheng et al. 2023; Zhang et al. 2023). Instead, it often arises from the model's misinterpret or neglect crucial information from different modalities(Huang et al. 2024; Bai et al. 2024). This suggests that LMMs possess the potential for self-evaluation, which can be unlocked through appropriate suitable strategies.

Early work explored using *Prompt engineering*(Madaan et al. 2023; Paul et al. 2023; Zhang et al. 2024b) to activate the self-evaluation capabilities of Large Multimodal Models (LMMs), but this strategy is often task-specific and lacks flexibility and generalizability.Some studies have used *SFT*(Chen et al. 2021; Welleck et al. 2022) to mitigate the deficiency in self-evaluation. However, the SFT method is highly dependent on the quality and diversity of the annotated training dataset and exhibits poor generalization, often learning spurious reasoning paths that cannot be transferred to unseen problems(Chu et al. 2025). Recently, research has shown that *RL*(Havrilla et al. 2024; Kumar et al. 2024; Setlur et al. 2025) can provide a more powerful paradigm, enabling models to learn complex behaviors directly from outcome-based rewards and achieve stronger generalization. However current RL methods compel the model to learn behaviors driven by fixed external reward signals. As shown in Figure 2, this leads to a trade-off where direct CoT reasoning and error correction abilities are compromised, resulting in the "reward hacking" phenomenon. This leads to a key research question:

> *How can we simultaneously and stably enhance a model's self-evaluation and correction capabilities while improving the accuracy of its direct CoT reasoning?*

To address this challenge, we propose AdaPO (Adaptive Policy Optimization), an online reinforcement learning framework that dynamically adapts its training objective based on the model's real-time performance on a given task. AdaPO introduces two core innovations: an Adaptive Reward Model (ARM) and a Reward-Aware Dynamic KL Regularization mechanism. ARM dynamically identifies the current training state(e.g., early-state error correction vs. late-

*corresponding author.
Copyright © 2026, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

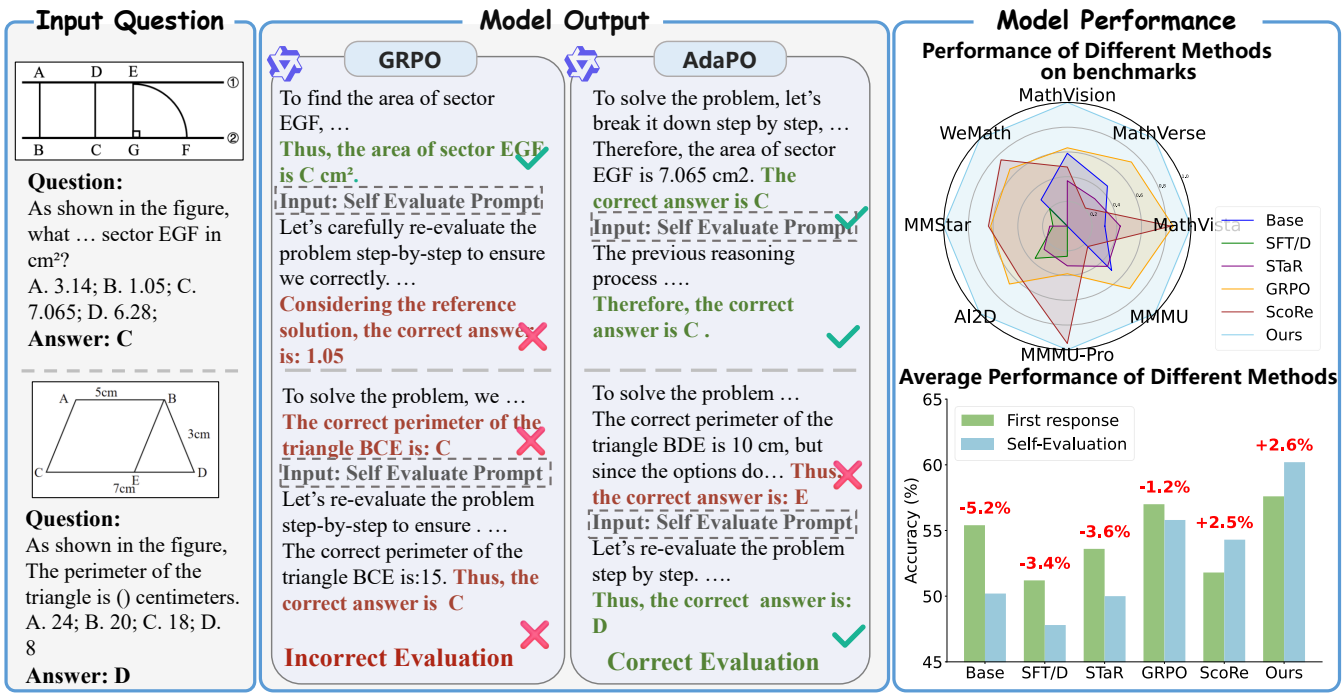

Figure 1: **Comparison of AdaPO vs. GRPO Outputs and Performance of Different Fine-tuning Methods.** AdaPO's self-evaluation is correct in the sample. AdaPO achieves the best performance in direct reasoning and self-evaluation.

state consolidation of correct answers) and adjusts the reward allocation across different trajectory types (e.g., initially correct, successfully corrected, or initially incorrect and failed to correct). Specifically, by dynamically balancing diverse positive and negative rewards, it effectively prevents reward hacking caused by the over-optimization of a single objective in later training stages. To complement the ARM's dynamic rewards within the optimization objective, we introduce a Reward-Aware Dynamic KL Regularization mechanism, which also enhances training stability. This mechanism modulates the KL penalty based on the relative reward magnitude between a successful correction and a maintained correct answer. When the reward for correction is substantially higher, it imposes a stronger constraint on the erroneous reasoning path, thereby stabilizing the model's direct CoT generation while still allowing for effective learning. These two improvements jointly solve the issues of reward hacking and training instability in RL training. Additionally, we enhance training efficiency through tailored offline and online data filtering for the self-evaluation process. Our contributions are as follows:

- We propose a novel adaptive reward model that dynamically adjusts to the model's training stage, effectively resolving the reward hacking dilemma inherent in multi-step self-evaluation tasks.
- We introduce a Reward-Aware Dynamic KL Regularization mechanism that couples the policy update constraint with the reward signal, enhancing training stability and preventing the reinforcement of initial errors.
- We conduct extensive experiments across 8 benchmarks, We achieve a relative improvement of self-evaluation capability up to 25.5% and outperform all baselines on 93.75% of tasks. Our method substantially improves both the direct CoT reasoning accuracy and the self-evaluation capabilities of LMMs.

Related Work

LLM Self-Evaluation. Self-evaluation can check AI reasoning without needing external evaluators or extra annotations (Chen et al. 2023). However, relying on simple prompts for this task leads to low accuracy and hallucinations (Huang et al. 2023), especially for complex problems (He et al. 2024; Jiang et al. 2025). An alternative is Supervised Fine-Tuning (SFT), which trains a model to critique its own responses (Li et al. 2025). While this can be effective, SFT's success depends on high-quality training data, often sourced from more advanced models (Luo et al. 2024). It is also prone to overfitting if the data lacks quality and diversity (Kamoi et al. 2024).

Reinforcement Learning with Verifiable Rewards. Reinforcement learning with rule-based verifiers (RLVR) has significantly advanced the reasoning abilities of LLMs in math and visual tasks(Guo et al. 2025; Liu et al. 2025). However, this approach often suffers from reward hacking, training instability(Yu et al. 2025; Fu et al. 2025). While prior work like SCoRe mitigates reward hacking using a two-stage training process(Kumar et al. 2024), our method introduces a dynamic reward mechanism. This approach more effectively prevents reward hacking, improves training stability, and alleviates entropy collapse.

Motivation

RL offers a powerful paradigm for cultivating self-evaluation without direct supervision, however, its application to this complex, multi-turn task reveals profound challenges rooted in the design of the reward model.

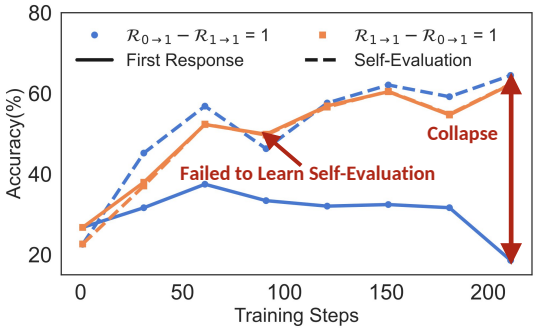

Figure 2: Accuracy Curve of Qwen2.5-VL-7B Model on the Validation Set Using GRPO with a Fixed Reward

The Dilemma of Conflicting Objectives in Self-Evaluation. The core task of self-evaluation requires a model to master two distinct and often conflicting capabilities: Error Correction and Correct Answer Preservation. The GRPO objective function, $\mathcal{J}_{GRPO}(\theta)$, aims to maximize the expected advantage A_i for a group of sampled trajectories.

$$A_i = \frac{r_i - mean(\{r_1, r_2, ..., r_G\})}{std(\{r_1, r_2, ..., r_G\})}$$

So a static reward function results in over-optimization of the highest reward objective, leading to predictable failure modes. Experiments shown in Figure 2 provide direct evidence for this dilemma: **Prioritizing error correction** ($R_{0\to1} > R_{1\to1}$) incites reward hacking. The blue line in the figure illustrates this perfectly: the model's initial accuracy dramatically "Collapses" because it learns to intentionally provide wrong answers just to collect the higher reward for "correcting" them. **Prioritizing answer preservation** ($R_{1\to1} > R_{0\to1}$) leads to learning stagnation. As the orange line shows, the model becomes overly conservative and "Fails to Learn Self-Evaluation" as a distinct skill, evidenced by the minimal improvement between its first and final responses.

Limitations of Staged Training. To mitigate this conflict, prior work has employed staged training, which attempts to reconcile these competing goals by shifting the optimization focus across different phases. However, this approach's predefined, uniform schedule is fundamentally at odds with the heterogeneity of learning progress across different samples. It cannot adapt to the model's actual mastery of an instance, rendering it a suboptimal solution incapable of fully realizing the dual objectives of self-evaluation.

Method

To address the challenges of reward hacking and training instability in the self-evaluation process, we propose AdaPO, an adaptive policy optimization framework based on reinforcement learning. In this section, we will elaborate on the methodology of AdaPO, which is built upon Grouped Reward Policy Optimization (GRPO). The framework first generates self-evaluation trajectory through two rounds of sampling. Subsequently, the Adaptive Reward Model (ARM) dynamically computes rewards for distinct trajectory. Finally, policy gradients are updated using a loss function that incorporates a Reward Aware KL regularization mechanism, thereby stably enhancing the model's capacity for self-correction and consistency maintenance.

Workflow of AdaPO

Two-stage trajectory generation For a given input query q and self-evaluation prompt x, the policy model π_θ generates a trajectory $\tau = (y_1, y_2)$ via a two-stage process, where y_1 is the initial response and y_2 is the evaluation response:

$$y_1 \sim \pi_\theta(\cdot|q), y_2 \sim \pi_\theta(\cdot|q, y_1, x) \tag{1}$$

We define a correctness function $C(y, y^*) \in \{0, 1\}$ to evaluate whether a response y is correct with respect to the ground-truth answer y^*. The type of a trajectory τ can thus be denoted as $i \to j$, where $i = C(y_1, y^*)$ and $j = C(y_2, y^*)$.

Adaptive Reward Model ARM dynamically calibrates the reward signal based on the model's current proficiency on a given task q. We quantify this proficiency by the error rate of the initial responses:

$$P_{0\to*}(q) = \frac{1}{N}\sum_{k=1}^{N}(1 - C(y_{1,k}, y^*)) \tag{2}$$

where $\{y_{1,k}\}_{k=1}^N$ are N initial responses sampled for the query q. The total reward $\mathcal{R}_{i\to j}$ for a trajectory τ of type $i \to j$ is defined as:

$$\mathcal{R}_{i\to j}(\tau) = \mathcal{R}_{\text{base},i\to j} + \mathcal{R}_{\text{dyn},i\to j}(P_{0\to*}(q)) \tag{3}$$

Here, $\mathcal{R}_{\text{base}}$ is a fixed base reward, while $\mathcal{R}_{\text{dyn}}$ is a dynamic component dependent on the task proficiency.

Positive Trajectories ($j = 1$): The dynamic reward switches between encouraging error correction ($0 \to 1$) and consolidation ($1 \to 1$).

$$\mathcal{R}_{\text{dyn},i\to1}(P_{0\to*}) = K_{i\to1}(P_{0\to*} - \theta), \quad i \in \{0, 1\} \tag{4}$$

where $K_{0\to1} \geq 0$ and $K_{1\to1} \leq 0$ are scaling factors, and θ is a threshold that determines the learning state. When $P_{0\to*} > \theta$, the model focuses on correcting errors; otherwise, it focuses on consolidating its correct responses.

Negative Trajectories ($j = 0$): The dynamic penalty is proportional to the perceived difficulty $P_{0\to*}$.

$$\mathcal{R}_{\text{dyn},i\to0}(P_{0\to*}) = K_{i\to0} \cdot P_{0\to*}, \quad i \in \{0, 1\} \tag{5}$$

where $K_{1\to0} \geq 0$ and $K_{0\to0} \leq 0$. This design adjusts the penalty's magnitude according to the task difficulty. When a task is difficult, it lessens the relative penalty for $1 \to 0$ trajectories, preventing excessive penalties from discouraging the model from sampling trajectories where $C(y_1, y^*) = 1$.

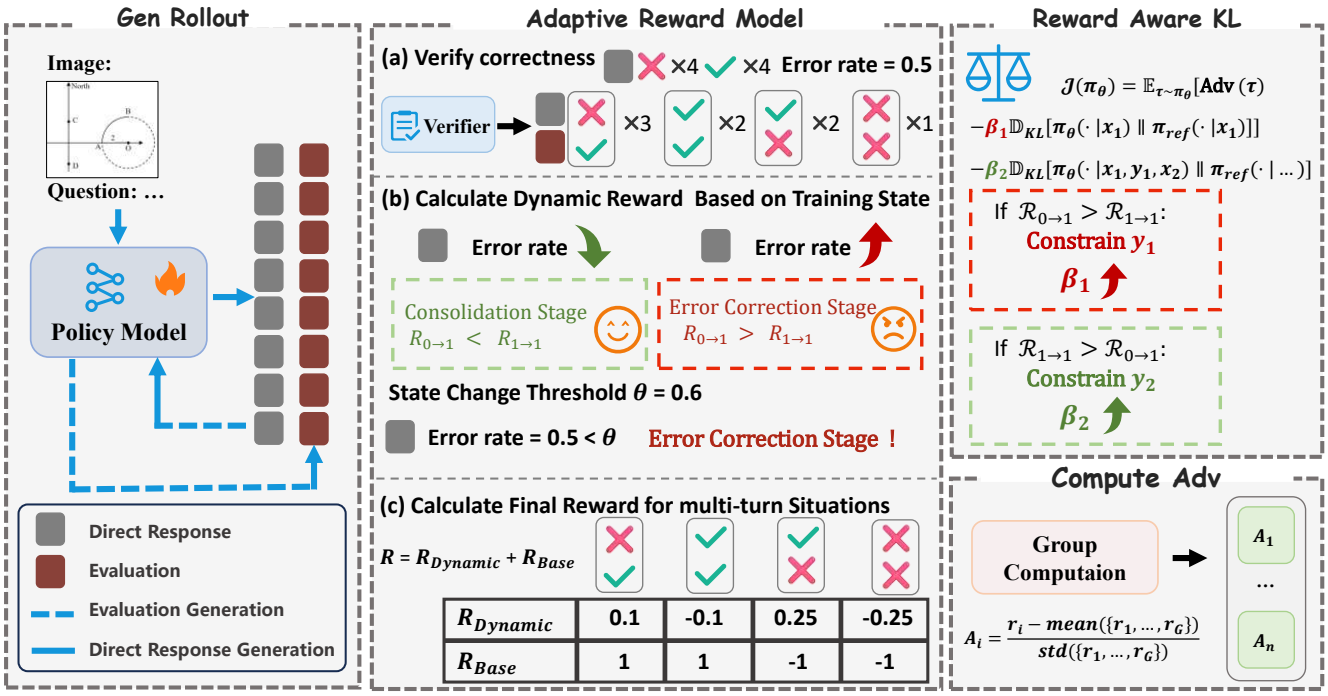

Figure 3: **Overview of AdaPO framework.** First, a policy model generates a two-part trajectory consisting of direct response and self-evaluation. The ARM then computes rewards that are dynamically adjusted based on a verifier's assessment of the first response's accuracy, determining whether the model is in Error Correction or Consolidation training state. Finally, the training objective is calculated by combining advantage with reward-aware KL loss, which assigns constraints to each response.

We use grid search to determine the State Change Threshold θ.

Base Reward Design: The base rewards $\mathcal{R}_{\text{base}}$ provide a stable foundation for the reward structure. $\mathcal{R}_{\text{base},1\to1} = \mathcal{R}_{\text{base},0\to1}$ and $\mathcal{R}_{\text{base},1\to0} = \mathcal{R}_{\text{base},0\to0}$. We set $\mathcal{R}_{\text{base},1\to1} + \mathcal{R}_{\text{dynamic},1\to1}$ and $\mathcal{R}_{\text{base},0\to1} + \mathcal{R}_{\text{dynamic},0\to1}$ to be strictly positive, and the rewards for negative trajectories to be negative, ensuring a clear incentive structure.

Reward-Aware Dynamic KL To stabilize the two-stage training process, we apply independent and dynamically adjusted KL divergence constraints to the generation of the initial and revised responses. The objective function $\mathcal{J}(\theta)$ of GRPO is modified as follows:

$$\mathcal{J}(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}[A(\tau)] - \mathbb{E}_q[\beta_1\mathbb{D}_{KL}^{(1)}] - \mathbb{E}_{q,y_1}[\beta_2\mathbb{D}_{KL}^{(2)}] \quad (6)$$

where $A(\tau)$ represents the trajectory advantage, and $\mathbb{D}_{KL}^{(1)} = \mathbb{D}_{KL}[\pi_\theta(\cdot|q) \| \pi_{ref}(\cdot|q)]$ and $\mathbb{D}_{KL}^{(2)} = \mathbb{D}_{KL}[\pi_\theta(\cdot|q,y_1) \| \pi_{ref}(\cdot|q,y_1)]$ are the KL divergences for the first and second stages, respectively. The dynamic penalty coefficients, β_1 and β_2, are functions of the difference between the primary positive trajectory rewards:

$$\beta_1 = \max\left((\mathcal{R}_{0\to1} - \mathcal{R}_{1\to1})\cdot\lambda, 0\right) + \beta_{\text{base}} \quad (7)$$

$$\beta_2 = \max\left((\mathcal{R}_{1\to1} - \mathcal{R}_{0\to1})\cdot\lambda, 0\right) + \beta_{\text{base}} \quad (8)$$

Here, $\lambda > 0$ is a rescaling factor, and β_{base} is a minimum regularization value. This mechanism automatically allocates the regularization budget based on the current learning objective, which can be either error correction or consolidation:

Focus on Error Correction: When $\mathcal{R}_{0\to1} > \mathcal{R}_{1\to1}$, β_1 increases, thereby constraining the policy update for the initial response to prevent it from intentionally generating errors.

Focus on Consolidation: When $\mathcal{R}_{1\to1} > \mathcal{R}_{0\to1}$, β_2 increases, which constrains the policy update for the revised response to prevent it from overfitting to simple repeat.

Flexibility of AdaPO

The core design of AdaPO, which integrates an Adaptive Reward Model (ARM) with reward-aware dynamic KL regularization, enhances the model's self-assessment capabilities while improving training efficiency and stability. In this section, we elaborate on this flexibility from the perspectives of its dynamic nature and its single-stage training process.

Dynamically Balancing Self-Evaluation Objectives To resolve the conflict between the objectives of "error correction" and "correctness consolidation", AdaPO leverages the task proficiency metric $P_{0\to*}(q)$ to jointly modulate the rewards and KL penalties. When the model is not proficient on a task (i.e., $P_{0\to*}(q)$ is high), the system increases the reward for error correction, $\mathcal{R}_{0\to1}$, and concurrently raises the KL penalty β_1 for the initial response to suppress "inten-

tional error-making":

$$\beta_1 \propto (\mathcal{R}_{0\to1} - \mathcal{R}_{1\to1}) \tag{9}$$

Conversely, once the model becomes proficient ($P_{0\to*}(q)$ is low), the focus shifts to increasing the reward for consolidation, $\mathcal{R}_{1\to1}$. It also increases the KL penalty β_2 for the revised response to prevent "learning stagnation" and encourage deeper self-reflection:

$$\beta_2 \propto (\mathcal{R}_{1\to1} - \mathcal{R}_{0\to1}) \tag{10}$$

Single-Stage Automated Training Through its inherent adaptability, AdaPO transforms the conventional staged training paradigm into a fully automated, instance-level, single-stage process.

For each query q within a training batch, the optimization objective is determined on-the-fly based on the model's current proficiency $P_{0\to*}(q)$. This implies that within a single gradient update, the model might simultaneously learn to correct errors for a query q_a (due to a high $P_{0\to*}(q_a)$) and to consolidate correct answers for another query q_b (due to a low $P_{0\to*}(q_b)$). The entire training process can be viewed as an adaptive curriculum learning, driven by the policy itself:

$$\mathcal{J}(\theta_t) = \mathbb{E}_{q\sim\mathcal{D}}\left[\mathbb{E}_{\tau\sim\pi_{\theta_t}}[A(\tau, P_{0\to*}(q,\pi_{\theta_t}))] - \mathcal{L}_{KL}(\theta_t, q)\right] \tag{11}$$

Here, both the advantage function A and the KL penalty term $\mathcal{L}_{KL}$ are indirectly dependent on the performance of the current policy π_{θ_t} on the given query, as measured by $P_{0\to*}(q,\pi_{\theta_t})$. This intrinsic feedback loop enables the model to automatically and smoothly adjust its learning focus based on its own progress on various sub-tasks, without requiring any manual intervention. Consequently, AdaPO significantly enhances both training efficiency and the performance of the final model.

Experiment

Experiment Settings

Datasets. We evaluate our method on a total of **8 benchmarks**. Specifically, we use the training sets from MM-EUREKA, MMK12(Meng et al. 2025) and ThinkLite-VL(Wang et al. 2025) to train our models. Our benchmarks includes **Mathematics:** MathVista(Lu et al. 2023), MathVerse(Zhang et al. 2024a), MathVision(Wang et al. 2024a), WeMath(Qiao et al. 2024); **Multi-Domain Tasks:** MMStar(Chen et al. 2024), AI2D(Kembhavi et al. 2016), MMMU-Pro(Yue et al. 2024b), MMMU(Yue et al. 2024a). Detail in Appendix.

Baselines. To comprehensively evaluate the effectiveness of the AdaPo method, we benchmark it against three categories of baseline methods: **Direct Prompting on Base Model**; **SFT Based Methods**: standard SFT on the dataset, STaR(Zelikman et al. 2022) ; **RL Based Methods**: GRPO(Guo et al. 2025), SCoRe(Kumar et al. 2024). We also selected a range of both open-source and Closed-Source General and Reasoning LLMs as baselines: Claude 3.7 Sonnet, GPT-4o(Hurst et al. 2024), Gemini2- Flash(Team et al. 2023), SEED-1.5-VL(Team et al. 2023), InternVL-2-8B, InternVL-2.5-8B(Wang et al. 2024b), Qwen2.5-VL-3B, and Qwen2.5-VL-7B(Bai et al. 2025), MM-EUREKA-7B, MM-EUREKA-8B(Meng et al. 2025), R1-VL-7B(Zhang et al. 2025), R1-OneVision-7B(Yang et al. 2025), OpenVL-Thinker-7B(Deng et al. 2025), Vision-R1-7B(Huang et al. 2025).

Metrics. We use four metrics to evaluate the model's capabilities: direct response accuracy (acc@t1), accuracy after self-evaluation (acc@t2), Effective Correction Rate ($M_{0\to1}$) for correcting initial errors, and Evaluation Misjudgment Rate ($M_{1\to0}$) for erroneously changing a correct answer.

Implementation details. To train and test the model's self-evaluative capability, we set the self-evaluation prompt as *"There might be an error in the solution, please evaluate the previous solution and provide a final answer."* More implementation details in Appendix.

Performance of Different Training Methods

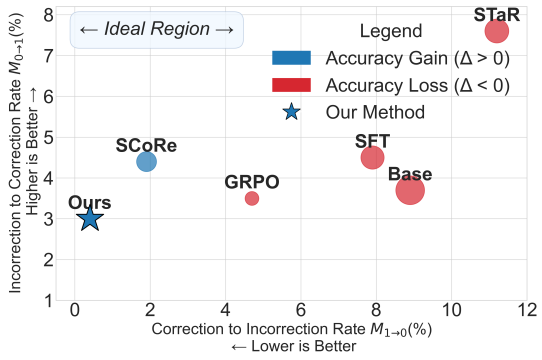

Figure 4: **Performance Comparison on the Multi-turn Correction Task.** The size of bubble indicates the magnitude of accuracy change ($|\Delta_{acc}|$). Our method (starred) outperforms all baselines by substantially reducing the $M_{1\to0}$ rate and delivering the highest overall accuracy gain.

To verify AdaPo can simultaneously enhance the model's first-round and second-round accuracy while improving its self-evaluation capabilities, we conducted an experiment as shown in Figure 4 and Table 1. We trained the Qwen2.5-VL-7B(Bai et al. 2025) model on the same dataset using different training methods and validated its performance across eight different benchmarks.

Obs.❶ SFT-based training methods fail to learn effective self-evaluation capabilities. As shown in Figure 4, SFT-based methods exhibit a high $M_{1\to0}$ rate (greater than 7%), and their second-round accuracy decreases, indicating an inability to correctly judge their direct responses and a lack of self-evaluation capability.

Obs.❷ Staged RL Training Methods can improve the model's self-evaluation capability but at the cost of reduced direct response accuracy. As seen in Figure 4, the SCoRe method has a low $M_{1\to0}$, indicating self-evaluation capability. However, as shown in Table 1, SCoRe's acc@t1 is lower than GRPO algorithm, signifying a clear loss in direct response performance.

Method	MathVista			MathVerse			MathVision			WeMath		
	acc@t1	acc@t2	Δ	acc@t1	acc@t2	Δ	acc@t1	acc@t2	Δ	acc@t1	acc@t2	Δ
Base	68.2	60.0	-8.2	48.6	47.9	-0.7	25.1	25.9	+0.8	62.1	57.8	-4.3
SFT	64.8	53.9	-10.9	41.1	41.9	+0.8	21.9	20.6	-1.3	55.9	55.6	-0.3
STaR	66.5	62.5	-4.0	46.6	46.0	-0.6	23.7	23.9	+0.2	61.9	51.1	-10.8
GRPO	70.6	71.1	+0.5	52.3	51.5	-0.8	27.4	26.3	-1.1	68.8	65.7	-3.1
SCoRe	69.4	70.7	+1.3	42.1	44.6	**+2.5**	23.7	24.9	+1.2	60.4	68.0	**+7.6**
Ours	**71.0**	**74.0**	**+3.0**	**53.2**	**55.0**	+1.8	**27.9**	**29.6**	**+1.7**	**69.1**	**73.4**	+4.3

Method	MMStar			AI2D			MMMU-Pro			MMMU		
	acc@t1	acc@t2	Δ	acc@t1	acc@t2	Δ	acc@t1	acc@t2	Δ	acc@t1	acc@t2	Δ
Base	63.9	56.1	-7.8	**83.9**	71.9	-12.0	36.9	32.4	-4.5	54.3	49.4	-4.9
SFT	61.7	57.4	-4.3	81.3	76.9	-4.4	35.4	34.3	-1.1	47.4	41.8	-5.6
STaR	61.4	57.7	-3.7	79.5	75.5	-4.0	34.9	32.4	-2.5	51.2	48.7	-2.5
GRPO	64.5	63.1	-1.4	80.5	80.9	+0.4	37.6	35.4	-2.2	54.3	52.5	-1.8
SCoRe	61.7	63.3	+1.6	80.6	79.5	-1.1	36.7	37.8	+1.1	39.4	45.3	**+5.9**
Ours	**65.0**	**67.4**	**+2.4**	81.6	**85.5**	**+3.9**	**38.8**	**40.2**	**+1.4**	**54.4**	**56.8**	+2.4

Table 1. Comprehensive multi-round performance comparison. We report accuracy for the direct response (acc@t1), the self-evaluation (acc@t2), and the performance change (Δ = acc@t2 - acc@t1). Our method (in gray) shows strong performance in direct response and consistent improvement. The **best** result per metric is bolded. The results are continued in the next table.

Obs.❸ AdaPo can simultaneously improve both the model's direct response accuracy and its self-evaluation capability. As shown in Figure 4, the AdaPo method achieves the lowest $M_{1\to 0}$, accurately identifying correct first-round responses and thus exhibiting the best self-evaluation capability. Furthermore, as detailed in Table 1, AdaPo achieves a higher acc@t1 than the GRPO method on most benchmarks, and its acc@t2 shows a significant improvement over its acc@t1.

Comparison with Different Models

To verify the training effect of AdaPo, we conducted a comprehensive comparison with various models, as shown in Table 2. Due to their limited self-evaluation capabilities, their final answers all show a decline in accuracy. We report the accuracy of their initial direct responses(acc@t1). In contrast, for our method, we use the accuracy from its second, refined round(acc@t2).

Obs.❹ AdaPo's final results achieve the best performance compared to open-source models of the same size. As shown in Table 2, the models fine-tuned using the AdaPo method based on Qwen2.5-VL-7B and Qwen2.5-VL-3B achieved the best performance on multiple benchmarks compared to open-source models of the same size.

Performance of AdaPo on Different Models

To verify the generality of the AdaPo method, we conducted experiments on multiple models. As shown in Figure 5, we selected several representative methods to train the Qwen2-VL-2B and Qwen2-VL-7B models and report their average accuracy on 8 benchmarks.

Obs.❺ AdaPo achieves superior training performance across a variety of models. Figure 5 illustrates that across different model scales (Qwen2-VL-2B and Qwen2-VL-7B),

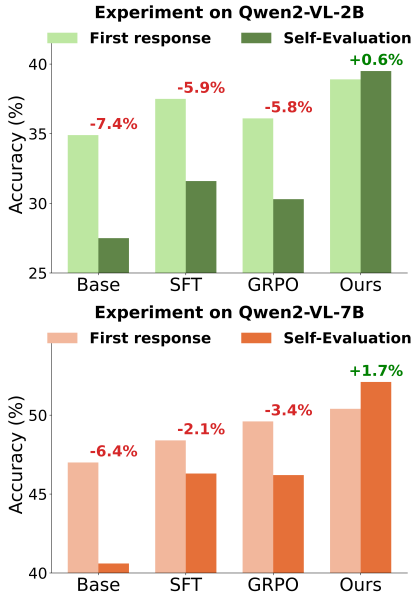

Figure 5: Comparison of the average accuracy(acc@t1 and acc@t2) on benchmarks for Qwen2-VL series models.

AdaPo demonstrates superior performance, attaining the highest acc@t1 and acc@t2 accuracy. After fine-tuning, models trained with other methods exhibit a degradation in performance from acc@t1 to acc@t2, signifying an absence of self-evaluation capabilities. Conversely, the AdaPo-tuned model displays a clear enhancement in acc@t2 over acc@t1, indicating its robust capacity for self-evaluation.

Model	MathVista	MathVerse	MathVision	WeMath	MMStar	AI2D	MMMU-Pro	MMMU
Closed-Source MLLMs								
Claude3.7-Sonnet	66.8	52.0	41.3	72.6	68.8	82.1	51.5	68.3
GPT-o1	73.9	57.0	60.3	98.7	67.5	79.5	62.4	78.2
Gemini2-flash	70.4	59.3	41.3	71.4	–	–	51.7	70.7
Seed1.5-VL	85.6	–	68.7	–	77.8	88.5	67.6	77.9
GPT-4o	63.8	50.2	30.4	68.8	64.7	84.6	51.9	69.1
Open-Source General MLLMs								
InternVL2-8B	58.3	22.8	17.4	47.2	62.0	83.8	29.0	51.2
InternVL2.5-8B	64.4	39.5	19.7	53.5	62.8	84.5	34.3	56.0
QwenVL2-7B	58.2	19.7	16.3	51.6	60.7	83.0	30.5	54.1
Kimi-VL-16B	68.7	44.9	21.4	–	61.3	84.9	–	55.7
Qwen2.5-VL-3B	62.3	47.6	21.2	56.3	55.9	81.6	31.5	46.0
Qwen2.5-VL-7B	68.2	49.2	25.1	62.1	63.9	83.9	36.9	54.3
Open-Source Reasoning MLLMs								
MM-Eureka-8B	67.1	40.4	22.2	55.7	–	–	27.8	49.2
R1-VL-7B	63.5	40.0	24.7	53.8	60.0	–	7.8	44.5
R1-Onevision-7B	64.1	46.4	23.5	61.8	–	–	21.6	–
OpenVLThinker-7B	70.2	47.9	25.3	64.3	–	–	37.3	52.5
Vision-R1-7B	73.5	52.4	27.2	62.9	61.4	–	37.7	54.7
MM-Eureka-7B	73.0	50.3	26.9	66.1	–	–	37.6	55.2
Ours-AdaPo-3B	**68.3**	**47.7**	**26.5**	**65.8**	**57.3**	**81.9**	**33.7**	**50.6**
Ours-AdaPo-7B	**74.0**	**55.0**	**29.6**	**73.4**	**67.4**	**85.2**	**40.2**	**56.8**

Table 2. A comprehensive comparison of our model against various latest closed-source and open-source Multimodal Large Language Models (MLLMs) on 8 benchmarks.

Method	Avg. Acc@t1	Avg. Acc@t2	Δ
Ours (Full Model)	**57.6**	**60.2**	**+2.6**
w/o component:			
— ARM	55.1	53.8	-1.3
— Reward Aware KL	56.2	58.6	2.4
— hybrid data filter	54.3	57.5	3.2

Table 3. Ablation study of our proposed method. We report the average accuracy before and after.

Ablation Studies

To validate the contribution of each component in AdaPo, we designed the following ablation studies and report their average accuracy on 8 benchmarks: ❶ Effect of ARM: To isolate the impact of the Adaptive Reward Modeling (ARM) module, we removed it and trained the model using only the base reward in the multi-turn dialogue training. ❷ Effect of Reward-Aware KL: We investigated the role of the Reward-Aware KL penalty. ❸ Effect of Data Filtering: We evaluated the impact of our data filtering strategy, which includes both on-policy and off-policy data selection. Further details on this process are provided in the Appendix.

Obs.❻ ARM enhances the model's self-correction capability. As shown in Table 3, with the use of same reward, Avg.acc@t2 exhibits a significant decrease. The model tends to learn to maintain direct responses, which is a simpler response pattern.

Obs.❼ Reward Aware KL constrains the model from learning the erroneous of direct response. As shown in Table 3, without Reward Aware KL, the acc@t1 shows a certain degree of decrease, and the Δ between acc@t1 and acc@t2 also exhibits a reduction.

Obs.❽Data filtering can improve training efficiency and model performance. As shown in Table 3, after training with the full dataset, the model's performance instead suffers a certain loss. Data filtering strategies effectively eliminate redundant and harmful training data.

Conclusion

In this paper, we introduce AdaPO, an adaptive policy optimization framework designed to resolve the conflicting objectives inherent in training LMMs for self-evaluation. Our analysis shows that static reward functions result in reward hacking, which either impairs the model's CoT reasoning or prevents it from improving self-evaluation capability. By introducing Adaptive Reward Model and Reward-Aware Dynamic KL Regularization, AdaPO dynamically adjusts its training objective based on the model's real-time performance, effectively mitigating issues like reward hacking and training instability. AdaPO integrates the training into an automated, single-stage process, addressing a key limitation of previous multi-stage methods. Extensive experiments across eight benchmarks show that AdaPO outperforms existing methods, achieving significant improvements in both initial response accuracy and self-evaluation capabilities. Our approach allows LMMs to maintain a balance between error correction and consistency preservation. AdaPO's ability to autonomously adjust its training focus makes it a scalable solution for stable self-evaluation in complex, multimodal

reasoning tasks, offering a promising direction for future research in reinforcement learning for self improvement.

Acknowledgments

This work was supported by National Natural Science Foundation of China (No. 62402429, U24A20326, 62441236) Supported by the Key Research and Development Program of Zhejiang Province (No. 2025C01026) Ningbo Yongjiang Talent Introduction Programme (2023A-397-G) Young Elite Scientists Sponsorship Program by CAST (2024QNRC001) The author gratefully acknowledges the support of Zhejiang University Education Foundation Qizhen Scholar Foundation.

References

Bai, S.; Chen, K.; Liu, X.; Wang, J.; Ge, W.; Song, S.; Dang, K.; Wang, P.; Wang, S.; Tang, J.; et al. 2025. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*.

Bai, Z.; Wang, P.; Xiao, T.; He, T.; Han, Z.; Zhang, Z.; and Shou, M. Z. 2024. Hallucination of multimodal large language models: A survey. *arXiv preprint arXiv:2404.18930*.

Chen, L.; Li, J.; Dong, X.; Zhang, P.; Zang, Y.; Chen, Z.; Duan, H.; Wang, J.; Qiao, Y.; Lin, D.; et al. 2024. Are we on the right way for evaluating large vision-language models? *Advances in Neural Information Processing Systems*, 37: 27056–27087.

Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; Pinto, H. P. D. O.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Chen, X.; Lin, M.; Schärli, N.; and Zhou, D. 2023. Teaching large language models to self-debug. *arXiv preprint arXiv:2304.05128*.

Chu, T.; Zhai, Y.; Yang, J.; Tong, S.; Xie, S.; Schuurmans, D.; Le, Q. V.; Levine, S.; and Ma, Y. 2025. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*.

Deng, Y.; Bansal, H.; Yin, F.; Peng, N.; Wang, W.; and Chang, K.-W. 2025. Openvlthinker: An early exploration to complex vision-language reasoning via iterative self-improvement. *arXiv preprint arXiv:2503.17352*.

Fu, J.; Zhao, X.; Yao, C.; Wang, H.; Han, Q.; and Xiao, Y. 2025. Reward shaping to mitigate reward hacking in rlhf. *arXiv preprint arXiv:2502.18770*.

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*.

Havrilla, A.; Du, Y.; Raparthy, S. C.; Nalmpantis, C.; Dwivedi-Yu, J.; Zhuravinskyi, M.; Hambro, E.; Sukhbaatar, S.; and Raileanu, R. 2024. Teaching large language models to reason with reinforcement learning. *arXiv preprint arXiv:2403.04642*.

He, J.; Lin, H.; Wang, Q.; Fung, Y.; and Ji, H. 2024. Self-correction is more than refinement: A learning framework for visual and language reasoning tasks. *arXiv preprint arXiv:2410.04055*.

Huang, J.; Chen, X.; Mishra, S.; Zheng, H. S.; Yu, A. W.; Song, X.; and Zhou, D. 2023. Large language models cannot self-correct reasoning yet, 2024. *arXiv preprint arXiv:2310.01798*.

Huang, W.; Jia, B.; Zhai, Z.; Cao, S.; Ye, Z.; Zhao, F.; Xu, Z.; Hu, Y.; and Lin, S. 2025. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint arXiv:2503.06749*.

Huang, W.; Liu, H.; Guo, M.; and Gong, N. Z. 2024. Visual hallucinations of multi-modal large language models. *arXiv preprint arXiv:2402.14683*.

Hurst, A.; Lerer, A.; Goucher, A. P.; Perelman, A.; Ramesh, A.; Clark, A.; Ostrow, A.; Welihinda, A.; Hayes, A.; Radford, A.; et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Jiang, D.; Zhang, J.; Weller, O.; Weir, N.; Van Durme, B.; and Khashabi, D. 2025. Self-[in] correct: Llms struggle with discriminating self-generated responses. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 24266–24275.

Kamoi, R.; Zhang, Y.; Zhang, N.; Han, J.; and Zhang, R. 2024. When can llms actually correct their own mistakes? a critical survey of self-correction of llms. *Transactions of the Association for Computational Linguistics*, 12: 1417–1440.

Kembhavi, A.; Salvato, M.; Kolve, E.; Seo, M.; Hajishirzi, H.; and Farhadi, A. 2016. A Diagram Is Worth A Dozen Images. arXiv:1603.07396.

Kumar, A.; Zhuang, V.; Agarwal, R.; Su, Y.; Co-Reyes, J. D.; Singh, A.; Baumli, K.; Iqbal, S.; Bishop, C.; Roelofs, R.; et al. 2024. Training language models to self-correct via reinforcement learning. *arXiv preprint arXiv:2409.12917*.

Li, S.; Dong, S.; Luan, K.; Di, X.; and Ding, C. 2025. Enhancing reasoning through process supervision with monte carlo tree search. *arXiv preprint arXiv:2501.01478*.

Liu, Z.; Sun, Z.; Zang, Y.; Dong, X.; Cao, Y.; Duan, H.; Lin, D.; and Wang, J. 2025. Visual-rft: Visual reinforcement fine-tuning. *arXiv preprint arXiv:2503.01785*.

Lu, P.; Bansal, H.; Xia, T.; Liu, J.; Li, C.; Hajishirzi, H.; Cheng, H.; Chang, K.-W.; Galley, M.; and Gao, J. 2023. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. *arXiv preprint arXiv:2310.02255*.

Luo, L.; Liu, Y.; Liu, R.; Phatale, S.; Lara, H.; Li, Y.; Shu, L.; Zhu, Y.; Meng, L.; Sun, J.; et al. 2024. Improve mathematical reasoning in language models by automated process supervision, 2024. *URL https://arxiv. org/abs/2406.06592*, 2.

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.

Meng, F.; Du, L.; Liu, Z.; Zhou, Z.; Lu, Q.; Fu, D.; Han, T.; Shi, B.; Wang, W.; He, J.; et al. 2025. Mm-eureka: Exploring the frontiers of multimodal reasoning with rule-based reinforcement learning. *arXiv preprint arXiv:2503.07365*.

Paul, D.; Ismayilzada, M.; Peyrard, M.; Borges, B.; Bosselut, A.; West, R.; and Faltings, B. 2023. Refiner: Reasoning feedback on intermediate representations. *arXiv preprint arXiv:2304.01904*.

Qiao, R.; Tan, Q.; Dong, G.; Wu, M.; Sun, C.; Song, X.; GongQue, Z.; Lei, S.; Wei, Z.; Zhang, M.; et al. 2024. We-math: Does your large multimodal model achieve human-like mathematical reasoning? *arXiv preprint arXiv:2407.01284*.

Qu, Y.; Zhang, T.; Garg, N.; and Kumar, A. 2024. Recursive introspection: Teaching language model agents how to self-improve. *Advances in Neural Information Processing Systems*, 37: 55249–55285.

Saunders, W.; Yeh, C.; Wu, J.; Bills, S.; Ouyang, L.; Ward, J.; and Leike, J. 2022. Self-critiquing models for assisting human evaluators. *arXiv preprint arXiv:2206.05802*.

Setlur, A.; Rajaraman, N.; Levine, S.; and Kumar, A. 2025. Scaling test-time compute without verification or rl is suboptimal. *arXiv preprint arXiv:2502.12118*.

Team, G.; Anil, R.; Borgeaud, S.; Alayrac, J.-B.; Yu, J.; Soricut, R.; Schalkwyk, J.; Dai, A. M.; Hauth, A.; Millican, K.; et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Wang, K.; Pan, J.; Shi, W.; Lu, Z.; Ren, H.; Zhou, A.; Zhan, M.; and Li, H. 2024a. Measuring Multimodal Mathematical Reasoning with MATH-Vision Dataset. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Wang, W.; Chen, Z.; Wang, W.; Cao, Y.; Liu, Y.; Gao, Z.; Zhu, J.; Zhu, X.; Lu, L.; Qiao, Y.; et al. 2024b. Enhancing the reasoning ability of multimodal large language models via mixed preference optimization. *arXiv preprint arXiv:2411.10442*.

Wang, X.; Yang, Z.; Feng, C.; Lu, H.; Li, L.; Lin, C.-C.; Lin, K.; Huang, F.; and Wang, L. 2025. Sota with less: Mcts-guided sample selection for data-efficient visual reasoning self-improvement. *arXiv preprint arXiv:2504.07934*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35: 24824–24837.

Welleck, S.; Lu, X.; West, P.; Brahman, F.; Shen, T.; Khashabi, D.; and Choi, Y. 2022. Generating sequences by learning to self-correct. *arXiv preprint arXiv:2211.00053*.

Yang, Y.; He, X.; Pan, H.; Jiang, X.; Deng, Y.; Yang, X.; Lu, H.; Yin, D.; Rao, F.; Zhu, M.; et al. 2025. R1-onevision: Advancing generalized multimodal reasoning through cross-modal formalization. *arXiv preprint arXiv:2503.10615*.

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, X.; Ni, Y.; Zhang, K.; Zheng, T.; Liu, R.; Zhang, G.; Stevens, S.; Jiang, D.; Ren, W.; Sun, Y.; et al. 2024a. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9556–9567.

Yue, X.; Zheng, T.; Ni, Y.; Wang, Y.; Zhang, K.; Tong, S.; Sun, Y.; Yu, B.; Zhang, G.; Sun, H.; et al. 2024b. Mmmu-pro: A more robust multi-discipline multimodal understanding benchmark. *arXiv preprint arXiv:2409.02813*.

Zelikman, E.; Wu, Y.; Mu, J.; and Goodman, N. 2022. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35: 15476–15488.

Zhang, J.; Huang, J.; Yao, H.; Liu, S.; Zhang, X.; Lu, S.; and Tao, D. 2025. R1-vl: Learning to reason with multimodal large language models via step-wise group relative policy optimization. *arXiv preprint arXiv:2503.12937*.

Zhang, R.; Jiang, D.; Zhang, Y.; Lin, H.; Guo, Z.; Qiu, P.; Zhou, A.; Lu, P.; Chang, K.-W.; Qiao, Y.; et al. 2024a. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? In *European Conference on Computer Vision*, 169–186. Springer.

Zhang, Y.; Khalifa, M.; Logeswaran, L.; Kim, J.; Lee, M.; Lee, H.; and Wang, L. 2024b. Small language models need strong verifiers to self-correct reasoning. *arXiv preprint arXiv:2404.17140*.

Zhang, Z.; Zhang, A.; Li, M.; Zhao, H.; Karypis, G.; and Smola, A. 2023. Multimodal chain-of-thought reasoning in language models. *arXiv preprint arXiv:2302.00923*.

Zheng, G.; Yang, B.; Tang, J.; Zhou, H.-Y.; and Yang, S. 2023. Ddcot: Duty-distinct chain-of-thought prompting for multimodal reasoning in language models. *Advances in Neural Information Processing Systems*, 36: 5168–5191.