

# Counteracting the Matthew Effect in Self-Improvement of LVLMs through Head-Tail Re-balancing

Xin Guo<sup>1\*</sup>, Zhiheng Xi<sup>1\*</sup>, Yiwen Ding<sup>1</sup>, Yitao Zhai<sup>2</sup>, Xiaowei Shi<sup>2</sup>,  
Xunliang Cai<sup>2</sup>, Tao Gui<sup>3,4†</sup>, Qi Zhang<sup>1</sup>, Xuanjing Huang<sup>1</sup>

<sup>1</sup>College of Computer Science and Artificial Intelligence, Fudan University

<sup>2</sup>Meituan <sup>3</sup>Institute of Trustworthy Embodied AI, Fudan University

<sup>4</sup>Shanghai Key Laboratory of Multimodal Embodied AI

xguo24@m.fudan.edu.cn, tgui@fudan.edu.cn

## Abstract

Self-improvement has emerged as a main-stream paradigm for advancing the reasoning capabilities of large vision–language models (LVLMs), where models explore and learn from successful trajectories iteratively. However, we identify a critical imbalance during this process: the model readily generates high-quality trajectories for simple queries (i.e., head data) but struggles with complex ones (i.e., tail data). This bias drives the optimization to disproportionately prioritize simple reasoning skills, while inhibiting the acquisition of complex capabilities. As iterations progress, this imbalance becomes more acute—a dynamic we term the “Matthew effect”<sup>1</sup>, ultimately stalling performance gains. To mitigate this, we approach head-tail re-balance during the exploration-and-learning process from two perspectives: distribution-reshaping and trajectory-resampling. Extensive experiments on Qwen2-VL-7B-Instruct and InternVL2.5-4B models across visual reasoning tasks demonstrate that our methods consistently improve visual reasoning capabilities, outperforming vanilla self-improvement baselines by an average of 3.86 points.<sup>2</sup>

## 1 Introduction

Large vision language models (LVLMs) have demonstrated impressive reasoning capabilities across complex multimodal tasks (Bai et al., 2025; Zhu et al., 2025). While supervised fine-tuning (SFT) can further improve model performance, its potential is limited by the scarcity of large-scale, high-quality annotated datasets (Zhang et al., 2025b; Peng et al., 2025).

\*Equal contribution.

†Corresponding author.

<sup>1</sup>Matthew effect is a sociological concept originally proposed by Robert K. Merton (Merton, 1968), which can be summarized as “the rich get richer and the poor get poorer”.

<sup>2</sup>Our code and data are available at <https://github.com/XinGuo2002/Head-Tail-Rebalancing>.

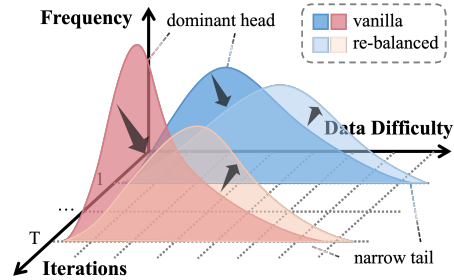


Figure 1: Matthew effect in self-improvement of LVLMs over iterations and our re-balanced solution. **Dark areas** illustrate the imbalanced distribution in vanilla self-improvement, where dominant head and narrow tail become more severe over iterations. **Light areas** depict re-balanced self-improvement—our methods for counteracting Matthew effect by reducing the head and augmenting the tail.

In response, self-improvement has emerged as a promising paradigm, enabling LVLMs to iteratively explore and learn from successful trajectories without reliance on manual annotation (Huang et al., 2023; Deng et al., 2024; Wang et al., 2025a). This approach fosters better alignment of the distributions between self-generated and real-world data (Jiang et al., 2025). However, our preliminary experiments reveal that self-improvement with varying sampling numbers  $K$  encounters severe performance bottleneck—or even degradation—in visual reasoning scenarios (§3.2).

We delve into this process (§3.3) and uncover the root cause as a significant imbalance in the distribution of self-generated successful trajectories (Dohmatob et al., 2024; Ding et al., 2025). Specifically, simple samples (i.e., head data) overwhelmingly dominate the distribution, while complex samples (i.e., tail data) remain underexplored. As illustrated in Figure 1, this imbalance exacerbates over iterations: the dominant head expands further while the narrow tail is increasingly marginalized, which we term the “Matthew effect” (Merton,

1968). Our analysis further reveals that this effect manifests as a worsening imbalance across difficulty levels (Tong et al., 2024; Xue et al., 2025) and a tendency towards shorter, simpler responses (Wang et al., 2023), ultimately leading to performance degradation.

To counteract this, we propose a methodological framework for head-tail re-balancing (§4)—inspired by Kong et al. (2025) and Xi et al. (2024)—which are categorized into two complementary perspectives: *distribution-reshaping* and *trajectory-resampling*. Regarding *distribution-reshaping*, we introduce **threshold clipping**, which truncates successful trajectories beyond a threshold  $\rho_0$  per query to suppress head dominance, and **repeat-based padding**, which equalizes frequency of all queries through repetition to augment tail data. In terms of *trajectory-resampling*, we enhance diversity through two distinct mechanisms: **adaptive-weighted resampling**, which dynamically adjusts resampling weights based on fail rates (Tong et al., 2024), and **guided resampling**, an efficiency-oriented strategy for tail data that initializes reasoning from varying intermediate steps.

We conduct experiments on visual reasoning tasks using Qwen2-VL-7B-Instruct (Wang et al., 2024a) and InternVL2.5-4B (Zhu et al., 2025) models with sampling configurations of  $K = 8$  and  $K = 16$ , respectively (§5). Empirical results demonstrate that our methods effectively mitigate Matthew effect in visual self-improvement through head-tail re-balancing, yielding significant performance gains over standard baselines.

Our contributions are summarized as follows:

- We systematically identify and analyze the critical performance bottlenecks in the interactive exploration and learning process of LVLMS, revealing the imbalance head-tail distribution and Matthew effect over iterations.
- To address this, we propose a methodological framework for head-tail re-balancing, which integrates four strategies from *distribution-reshaping* and *trajectory-resampling* perspectives.
- We conduct comprehensive experiments across multiple models and tasks, validating that our approach effectively counteracts the Matthew effect and consistently enhances visual reasoning capabilities.

## 2 Related Work

### 2.1 Self-improvement in visual reasoning.

LVLMS have demonstrated remarkable performance across various visual reasoning scenarios (Bai et al., 2025; Zhu et al., 2025), where self-improvement approaches have been widely employed (Yang et al., 2023; Deng et al., 2024; Wang et al., 2025a).

Among these, self-critic (Wang et al., 2025a) and self-correction (Ding and Zhang, 2025; Wu et al., 2025a; Cheng et al., 2025) emerge as prevalent optimization strategies. Prior work typically relies on separate critic models for error detection and correction, which requires substantial additional resources (Sun et al., 2025; Xiong et al., 2025; Zhang et al., 2025a). In contrast, Wang et al. (2025a) introduce a unified model that simultaneously generates responses and performs self-critic to refine. Similarly motivated, Ding and Zhang (2025) propose Sherlock to selectively revise erroneous segments in reasoning trajectories. Moreover, to enhance data efficiency, many studies employ direct preference optimization (DPO) in self-improvement (Deng et al., 2024; Wang et al., 2025a; Tanji and Yamasaki, 2025). However, these works lack a thorough exploration of the iterative process.

In this paper, we focus on the essence of vanilla self-improvement and propose targeted and effective strategies to enhance its performance.

### 2.2 Distribution bias in self-generated data.

Previous work has identified biases sampling and head-tail imbalance in self-improvement sampling process (Shumailov et al., 2024; Dohmatob et al., 2024) and addressed them by increasing the proportion of difficult queries through adaptive-weighted (Tong et al., 2024; Xue et al., 2025; Kong et al., 2025) and guided (Ding et al., 2025) sampling methods. However, these studies primarily focus on text-based reasoning while lacking thorough investigation into visual scenarios. While SynthRL (Wu et al., 2025b) enhances the distribution balance by rewriting queries on visual reasoning tasks, it does not delve into the iterative process.

Instead, we conduct an in-depth investigation into the observable performance and underlying properties of LVLMS during self-improvement, revealing similar distribution bias. To this end, we introduce four re-balancing strategies, effectively mitigating the Matthew effect and achieving significant performance improvements.

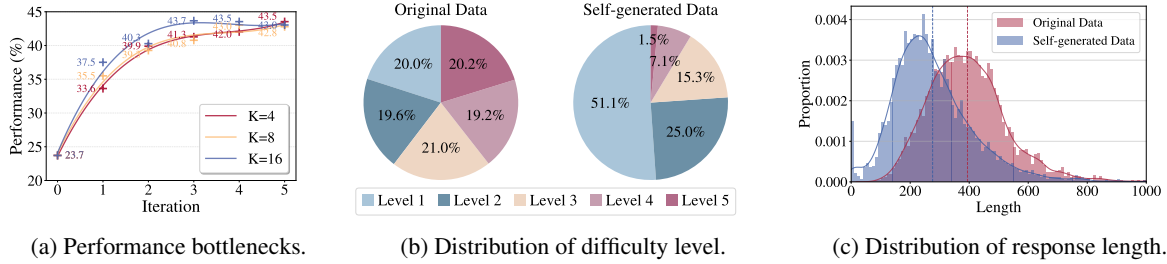


Figure 2: Performance bottlenecks and distribution characteristics in self-improvement. **(a)** Phenomenon of performance bottlenecks under different sampling numbers  $K$ . **(b)** Comparison of difficulty level distributions between original and self-generated data, ranging from level 1 (easiest) to level 5 (most difficult). **(c)** Differences in response length distributions between original and self-generated data, with dashed lines indicating mean values.

### 3 “Matthew Effect” in Self-improvement

While existing research has explored self-generated data, the self-improvement process in visual scenarios remains underexplored. Therefore, we delve into this process to uncover its intrinsic properties.

#### 3.1 Formulating Self-improvement

In this paper, we formulate the vanilla self-improvement as follows. Given a model  $\mathcal{M}_{\text{base}}$ , a training dataset  $\mathcal{D}_{\text{base}} = \{(q_i, a_i)\}_{i=1}^N$  where  $q_i$  represents the query,  $a_i$  denotes the ground-truth answer, and  $N$  is the dataset size, we define the initial model as  $\mathcal{M}_0 = \mathcal{M}_{\text{base}}$  and the number of iterations as  $T$ . At each iteration  $t \in [1, T]$ , the self-improvement sequentially performs three key stages: exploration, filtering, and learning.

**Exploration.** At iteration  $t$ , for each query  $q_i \in \mathcal{D}_{\text{base}}$ , the model  $\mathcal{M}_{t-1}$  generates  $K$  different responses  $\{\hat{r}_{i,k}^{(t)}\}_{k=1}^K$ , where  $\hat{r}_{i,k}^{(t)} \sim \mathcal{M}_{t-1}(\cdot|q_i)$ . Therefore, the sampled dataset is represented as

$$\mathcal{D}_{\text{sample}}^{(t)} = \{(q_i, \hat{r}_{i,k}^{(t)}) \mid 1 \leq i \leq N, 1 \leq k \leq K\}.$$

**Filtering.** To obtain high-quality training data, we define a binary reward function as

$$rf(q_i, a_i, \hat{a}_{i,k}^{(t)}) = \begin{cases} 0, & \text{if } \hat{a}_{i,k}^{(t)} \neq a_i \\ 1, & \text{if } \hat{a}_{i,k}^{(t)} = a_i \end{cases},$$

where  $\hat{a}_{i,k}^{(t)}$  is the answer extracted from response  $\hat{r}_{i,k}^{(t)}$ . Using it, we obtain the filtered dataset:

$$\mathcal{D}_{\text{filter}}^{(t)} = \{(q_i, \hat{r}_{i,k}^{(t)}) \in \mathcal{D}_{\text{sample}}^{(t)} \mid rf(q_i, a_i, \hat{a}_{i,k}^{(t)}) = 1\}.$$

**Learning.** At iteration  $t$ , the model  $\mathcal{M}_t$  is trained by fine-tuning the base model  $\mathcal{M}_{\text{base}}$  on the filtered

dataset  $\mathcal{D}_{\text{train}}^{(t)} = \mathcal{D}_{\text{filter}}^{(t)}$ . The objective is to minimize the negative log-likelihood:

$$\mathcal{L}_{\text{SFT}}^{(t)} = -\mathbb{E}_{(q,r) \sim \mathcal{D}_{\text{train}}^{(t)}} \sum_{j=1}^{|r|} \log P(r_j | q, r_{<j}; \mathcal{M}_t).$$

Through  $T$  iterations of this process, the model continuously generates higher-quality responses, which in turn enhances its performance in the next iterations, thereby achieving self-improvement.

#### 3.2 Plateaus in Varying Sampling Numbers

First, to reveal the impact of sampling number  $K$ , we conduct experiments over five self-improvement iterations with varying value of  $K$ . Figure 2a uncovers two key observations: (1) Different  $K$  exhibits similar performance trends during self-improvement: notable improvements in early iterations, rapid convergence to performance bottlenecks, and even declining trends (e.g.,  $K = 16$ ) in later iterations; (2) Sampling number plays a dominant role in the first iteration, where higher sampling number leads to better performance; while in later iterations, sampling number has virtually no impact on performance. For instance, at iteration 1, the  $K = 16$  model outperforms the  $K = 4$  model by 3.89 points on the test set. While at iteration 5, the  $K = 16$  model even performs 0.39 points worse than the  $K = 4$  model.

#### 3.3 Imbalance in Self-improvement

To figure out the bottlenecks of self-improvement, we then analyze the distribution of self-generated data (Tong et al., 2024; Ding et al., 2025).

**Differences between original and self-generated data.** To characterize the properties of self-generated data, we first compare it with the original data from two dimensions: *difficulty* and *length*.

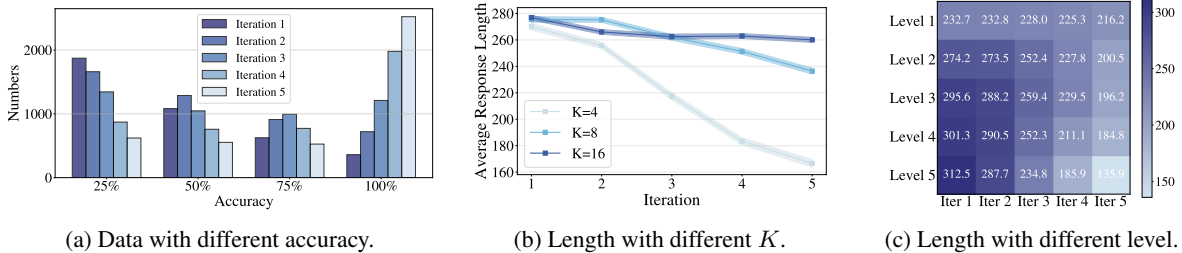


Figure 3: Matthew effect over iterations. (a) Matthew effect in the distribution of data in  $\mathcal{D}_{\text{filter}}$  with different accuracy under the setting of  $K = 4$ . (b) Trends in average response length (i.e., number of tokens) across iterations under different sampling numbers  $K$ . (c) Matthew effect in average response length for data of different difficulty levels across iterations under the setting of  $K = 8$ , where difficulty increases progressively from level 1 to level 5.

- **Imbalanced difficulty distribution.** As shown in Figure 2b, the difficulty distribution (see Appendix A for difficulty level categorization) differs markedly between the original and self-generated data, revealing a “dominant head and narrow tail” pattern. While original data exhibits balanced difficulty distribution, self-generated data suffers from severe imbalance, with easy samples (level 1) comprising 51.1% of the total and difficult samples (level 5) nearly absent.
- **Shorter average response length.** As revealed in Figure 2c, self-generated responses are significantly shorter than original ones (averaging 277 tokens vs. 395 tokens), with some extremely short responses (<10 tokens) lacking CoT reasoning. This suggests that self-generated data is prone to producing abbreviated reasoning processes, including instances that deliver conclusions directly even under CoT prompting.

**Matthew effect over iterations.** Further, we investigate distribution shifts during the iterative process. To begin with, we analyze how queries with different accuracy are distributed in self-generated data. Results in Figure 3a indicate that throughout iterations, well-mastered data (with 100% accuracy) occupies an increasing proportion, while poorly-performed data (with 25% accuracy) is gradually squeezed out of the training dataset  $\mathcal{D}_{\text{train}}$ —a phenomenon we term the “Matthew effect”. Given that high-accuracy queries account for more samples, the diminishing proportion of difficult tail becomes even more severe, limiting the self-improvement performance ceilings.

Next, we analyze response length changes during the iterative process. As shown in Figure 3b, average response length consistently declines across iterations, while higher sampling number mitigates this degradation. To delve deeper, we compare

average response length across various difficulty levels over iterations. Figure 3c reveals three key observations: (1) Simple data (level 1) maintains relatively stable response length during iterations, showing minimal degradation. (2) Difficult data (level 5) suffers the most severe length degradation, with a dramatic reduction of 56.5%. (3) At iteration 1, the higher the difficulty level, the longer the response length; however, this trend completely reverses by the fifth iteration, with difficult data generating the shortest responses of merely 136 tokens. These findings suggest that Matthew effect manifests in response length as well—simple data requiring shorter responses maintains appropriate length, whereas difficult data requiring longer responses undergoes significant degradation. For a more detailed analysis, please refer to Appendix B.

## 4 Methodology

To alleviate the phenomenon of Matthew effect, we propose four effective re-balanced strategies from two perspectives: distribution-reshaping and trajectory-resampling.

Regarding distribution-reshaping, motivated by under-sampling and over-sampling in machine learning (Mohammed et al., 2020), we propose two intuitive strategies. On one hand, we reduce the quantity of head to increase the proportion of tail samples, introducing the **Threshold Clipping (TC)** strategy. On the other hand, we directly augment the number of tail through repetition, proposing the **Repeated-based Padding (RP)** strategy.

From trajectory-resampling perspective, we also propose two strategies. The first is **Adaptive-weighted Resampling (AR)**, which dynamically adjusts resampling weights based on fail rate. The second is **Guided Resampling (GR)**, which initializes model exploration from varying intermediate reasoning steps (Xi et al., 2024; Ding et al., 2025).

## 4.1 Distribution Reshaping

**Threshold clipping.** Adopting the philosophy of “less is more”, we propose threshold clipping, which increases the proportion of tail by reducing the number of head.

Specifically, threshold clipping sets a threshold  $\rho_0$  and randomly truncates responses to limit each query to at most  $\rho_0$  correct responses. Formally, at iteration  $t$ , instead of using the filtered dataset directly, we train the base model  $\mathcal{M}_{\text{base}}$  on the following dataset:

$$\mathcal{D}_{\text{train-TC}}^{(t)} = \{(q_i, \hat{r}_{i,k}^{(t)}) \in \mathcal{D}_{\text{filter}}^{(t)} \mid k \leq \rho_0\}.$$

**Repeat-based padding.** Moreover, we consider increasing the quantity of tail directly and introduce repeat-based padding to enforce balanced data distribution.

Specifically, repeat-based padding ensures that all queries appear with equal frequency  $K$  in the next training dataset. To achieve this, queries with insufficient correct samples are padded through repetition. Formally, at iteration  $t$ , the training dataset is expressed as follows:

$$\mathcal{D}_{\text{train-RP}}^{(t)} = \{(q_i, \hat{r}_{i, k \bmod n_i}^{(t)}) \in \mathcal{D}_{\text{filter}}^{(t)} \mid 1 \leq k \leq K\},$$

where  $n_i$  is the number of  $\hat{r}_i$ , that is, the number of correct responses out of  $K$  sampling for  $q_i$ .

## 4.2 Trajectory Resampling

**Adaptive-weighted resampling.** However, merely reshaping data distribution yields limited benefits. As excessive duplication may reduce data diversity and trigger overfitting, we additionally employ resampling strategies to enhance the proportion of tail. Drawing inspiration from the “pass rate” (Team et al., 2025) and “fail rate” (Tong et al., 2024) metrics, we propose adaptive-weighted resampling, which dynamically adjusts the resampling weights for each query based on its fail rate.

Specifically, for a query with  $n_i$  successful trajectories out of  $K$  samples, we perform  $K - n_i$  additional resampling operations. This hierarchical resampling assigns more weight to tail data, thereby increasing their proportion. Formally, at iteration  $t$ , model  $\mathcal{M}_{t-1}$  generates  $K - n_i$  new responses  $\{\tilde{r}_{i,k}^{(t)}\}_{k=1}^{K-n_i}$  for each query  $q_i \in \mathcal{D}_{\text{base}}$ , where  $\tilde{r}_{i,k}^{(t)} \sim \mathcal{M}_{t-1}(\cdot | q_i)$ . Then we obtain the resampled and refiltered datasets:

$$\mathcal{D}_{\text{resample-AR}}^{(t)} = \{(q_i, \tilde{r}_{i,k}^{(t)}) \mid 1 \leq i \leq N, 1 \leq k \leq K - n_i\}$$

$$\mathcal{D}_{\text{refilter-AR}}^{(t)} = \{(q_i, \tilde{r}_{i,k}^{(t)}) \in \mathcal{D}_{\text{resample-AR}}^{(t)} \mid rf(q_i, a_i, \tilde{a}_{i,k}^{(t)}) = 1\},$$

and finally merge into the training dataset:

$$\mathcal{D}_{\text{train-AR}}^{(t)} = \mathcal{D}_{\text{filter}}^{(t)} \cup \mathcal{D}_{\text{refilter-AR}}^{(t)}.$$

**Guided resampling.** Nevertheless, adaptive-weighted resampling is essentially brute-force sampling with limited efficiency improvements. To this end, we propose guided resampling—a novel resampling strategy that initializes model exploration from various intermediate reasoning steps.

Specifically, the model exploits guided signals to achieve efficient resampling, starting its reasoning process from different intermediate reasoning steps. This strategy enables the model to navigate toward promising trajectories within the vast exploration space, while facilitating progressive learning of complex reasoning processes. Formally, for each successful trajectory of the tail  $\hat{r}_i \in \mathcal{D}_{\text{filter}}^{(t)}$ , we decompose it into  $S$  steps  $\hat{r}_{i(1)}, \dots, \hat{r}_{i(S)}$ . Generated by  $\tilde{r}_{i(s)} \sim \mathcal{M}_{t-1}(\cdot | q_i, \hat{r}_{i(<s)})$ , the resampled dataset is expressed as:

$$\mathcal{D}_{\text{resample-GR}}^{(t)} = \{(q_i, \hat{r}_{i(<s),k}^{(t)}, \tilde{r}_{i(s),k}^{(t)}) \mid 1 \leq k \leq n_i, 1 \leq s \leq S\}.$$

Similar to adaptive-weighted resampling, we obtain the refiltered and training dataset.

## 5 Experiments

### 5.1 Experimental Setups

**Datasets.** We adopt MMPR (Wang et al., 2024b)—a multimodal reasoning dataset derived from multiple sources—as our primary dataset. From it, we randomly extract 7,980 mathematical reasoning samples (Lu et al., 2021; Cao and Xiao, 2022; Seo et al., 2015) to construct a curated subset, MMPR-mini, with details presented in Appendix A. Besides, we also extend our evaluation to science and more reasoning tasks, with results presented in Section 6.3. For out-of-domain (OOD) evaluation, we further utilize MathVerse (Zhang et al., 2024) and We-Math (Qiao et al., 2024) datasets.

**Models.** We employ two widely-used LVLMs as our base models: Qwen2-VL-7B-Instruct (Wang et al., 2024a) and InternVL2.5-4B (Zhu et al., 2025). All analytical experiments in Section 3 are conducted on Qwen2-VL-7B-Instruct. Following the paradigm of Zelikman et al. (2024), we initialize from the base model instead of a further SFT model, and restart training from this base model at each iteration.

Models	K	Method	In-domain		Out-of-domain				Avg.	
			MMPR-mini		MathVerse		We-Math			
			final	opt.	final	opt.	final	opt.	final	opt.
Qwen2-VL-7B	8	w/o self-improvement	23.71		22.46		47.36		31.18	
		vanilla self-improvement	42.79	43.04	27.79	28.93	52.36	<u>52.93</u>	40.98	41.31
		TC (threshold clipping)	44.67	44.67	28.55	<u>29.06</u>	52.13	52.76	41.78	41.78
		RP (repeat-based padding)	<u>45.67</u>	<b>46.42</b>	<b>30.08</b>	<b>30.08</b>	52.59	52.59	<b>42.78</b>	<b>42.78</b>
		AR (adaptive-weighted resampling)	<u>45.67</u>	45.67	28.93	28.93	<u>52.82</u>	52.82	<u>42.47</u>	<u>42.47</u>
	GR (guided resampling)	<b>45.80</b>	<u>45.80</u>	27.28	28.30	<b>53.91</b>	<b>53.91</b>	42.33	42.33	
	vanilla self-improvement	43.04	43.66	28.30	28.30	52.76	<u>52.76</u>	<u>41.36</u>	<u>41.36</u>	
	TC (threshold clipping)	45.92	45.92	<u>28.68</u>	28.68	52.70	52.76	<u>42.43</u>	42.43	
	16	RP (repeat-based padding)	<u>47.05</u>	<u>47.05</u>	28.30	<b>30.33</b>	51.49	52.70	42.28	<u>42.56</u>
	AR (adaptive-weighted resampling)	43.41	43.41	28.55	28.93	<u>53.91</u>	<u>53.91</u>	41.96	41.96	
GR (guided resampling)	<b>47.68</b>	<b>47.68</b>	<b>29.95</b>	<u>29.95</u>	<b>54.20</b>	<b>54.20</b>	<b>43.94</b>	<b>43.94</b>		
InternVL2.5-4B	8	w/o self-improvement	47.55		29.31		48.10		41.66	
		vanilla self-improvement	64.62	64.62	29.95	<b>33.12</b>	50.40	52.59	48.32	48.80
		TC (threshold clipping)	63.99	63.99	<u>32.36</u>	<b>33.12</b>	<u>51.72</u>	<u>53.28</u>	49.36	49.77
		RP (repeat-based padding)	<u>67.13</u>	<b>68.13</b>	31.98	31.98	50.86	52.87	<u>49.99</u>	<u>50.54</u>
		AR (adaptive-weighted resampling)	66.25	66.25	30.33	31.85	50.75	52.41	49.11	49.11
	GR (guided resampling)	<b>67.50</b>	<u>67.50</u>	<b>33.12</b>	<b>33.12</b>	<b>51.90</b>	<b>54.20</b>	<b>50.84</b>	<b>50.84</b>	
	vanilla self-improvement	66.75	66.75	<u>31.60</u>	<u>33.63</u>	50.46	52.30	49.60	<u>50.57</u>	
	TC (threshold clipping)	64.74	64.74	30.33	33.38	48.74	50.57	47.94	48.99	
	16	RP (repeat-based padding)	<b>71.64</b>	<b>71.64</b>	<b>32.61</b>	<b>34.14</b>	51.32	<b>53.33</b>	<b>51.86</b>	<b>51.86</b>
	AR (adaptive-weighted resampling)	64.99	64.99	30.58	33.25	<b>51.67</b>	51.78	49.08	49.46	
GR (guided resampling)	<u>67.63</u>	<u>67.63</u>	31.09	31.98	<u>51.38</u>	<u>52.70</u>	<u>50.03</u>	50.03		

Table 1: Main results. The best result for each setting is in **bold**, while the second-best is marked with underline. “Final” indicates the final performance, and “opt.” indicates the optimal performance across all iterations.

**Implementation details.** All experiments are conducted on 8 A100-80GB GPUs using SWIFT (Zhao et al., 2025) framework for training and vLLM (Kwon et al., 2023) framework for sampling and testing. We set the number of self-improvement iterations  $T = 5$  and focus on the sampling number  $K = 8$  and  $K = 16$ . For training, we use a learning rate of  $3 \times 10^{-5}$  and train for 1 epoch to avoid overfitting. For sampling, we set the temperature to 0.7 with maximum 4096 new tokens. While for testing, the temperature is set to 0. Additionally, we configure method-specific parameters, including threshold  $\rho_0 = 4$  for TC and number of steps  $S = 4$  for GR.

## 5.2 Main Results

The main results are presented in Table 1, including the final and optimal performance. Overall, our findings are as follows:

**Scaling the sampling number  $K$  shows poor cost-efficiency in self-improvement.** While vanilla self-improvement yields substantial gains over the base model—for instance, improving the average performance of Qwen2-VL-7B-Instruct by 10.13 points at a sampling number of  $K = 8$ —further increasing  $K$  proves ineffective. Compared to  $K = 8$ , the optimal average performance

at  $K = 16$  improves by merely 0.05 points. Given the doubled computational cost, such a marginal gain is not cost-effective. These findings indicate that blindly scaling the sampling number through brute-force methods fails to provide the critical breakthrough for self-improvement enhancement.

**Head-tail re-balancing improves the performance of self-improvement across various models and datasets.** For more efficient enhancement, re-balancing the distribution of head and tail data throughout self-improvement achieves significant performance gains across varying models and datasets, particularly with RP and GR strategies. For example, with Qwen2-VL-7B-Instruct at  $K = 16$ , RP outperforms vanilla self-improvement by 3.39 points on the in-domain test set and 1.20 points on average, while GR achieves improvements of 4.02 and 2.58 points, respectively.

Moreover, comparison between final and optimal performance shows that vanilla self-improvement frequently exhibits suboptimal results in the final iteration relative to its peak performance, reflecting performance bottlenecks during training. In contrast, our re-balancing strategies, especially GR, consistently reach optimal performance in the final iteration, demonstrating greater stability and potential for further improvement.

**Head-tail re-balancing mitigates the Matthew effect in self-improvement.** As shown in Figure 4, our proposed strategies effectively alleviate the imbalanced difficulty distribution in successful trajectories. Among them, distribution-reshaping methods exhibit superior mitigation on Qwen2-VL-7B-Instruct at  $K = 16$ , with RP reducing head data proportion from 51.1% to 24.8% and boosting tail data from 1.5% to 6.6%. In contrast, AR shows limited tail data augmentation due to inefficient sampling in vast solution spaces, leading to modest performance gains. Conversely, GR focuses more on enhancing tail sample coverage and proportion. Despite slightly less mitigation than TC and RP, GR provides step-by-step guidance for tail samples, enables better mastery of complex reasoning trajectories, and yields substantial performance improvements from 41.36 to 43.94. Additional results of Matthew effect mitigation on InternVL2.5-4B and other sampling settings are provided in Appendix E.

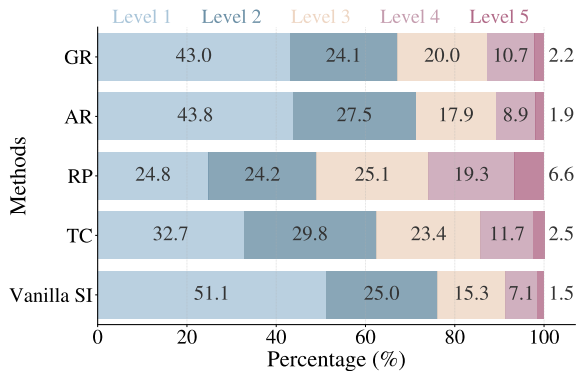


Figure 4: Data distribution of difficulty levels (1=easiest, 5=most difficult) in successful trajectories under different strategies with Qwen2-VL-7B-Instruct at  $K = 16$ .

As shown in Table 2, our method not only breaks through the performance bottleneck but also achieves superior performance at the same iteration steps. Whereas vanilla self-improvement reaches a plateau early within the five iterations, our method continues to improve, exhibiting a capability for long-term performance enhancement.

Method	Round1	Round2	Round3	Round4	Round5
vanilla self-improvement	35.51	39.27	40.78	43.04	42.79
guided resampling	37.01	42.41	43.04	43.78	45.80

Table 2: Performance during the iterative process.

Similarly, we also observe the mitigation of length reduction in Matthew effect, which discussed in Appendix B.

Method	MMPR	MathVerse	We-Math	Avg.
TC ( $\rho_0 = 2$ )	43.91	27.66	53.10	41.56
<b>TC (<math>\rho_0 = 4</math>)</b>	45.92	28.68	52.70	<b>42.43</b>
TC ( $\rho_0 = 8$ )	43.66	29.44	52.47	41.86
head clipping	41.78	28.93	53.10	41.27
repeat-based inverting	45.04	28.81	51.26	41.71
<b>RP</b>	47.05	28.30	51.49	<b>42.28</b>
GR ( $S = 2$ )	45.80	27.79	53.33	42.31
<b>GR (<math>S = 4</math>)</b>	47.68	29.95	54.20	<b>43.94</b>
GR ( $S = 8$ )	44.67	26.65	54.08	41.80

Table 3: Final performance of re-balancing variants on Qwen2-VL-7B-Instruct with sampling number  $K = 16$ .

## 6 Discussion

### 6.1 Ablation Study

**Variants of reshaping strategies.** We evaluate various re-balancing variants, with results presented in Table 3 (upper). First, experiments with TC reveal a trade-off in the choice of  $\rho_0$ : large  $\rho_0$  inadequately alleviates data imbalance, whereas small  $\rho_0$  fails to ensure sufficient diversity. We also explore **head clipping**, which removes fully-correct queries (i.e.,  $n_i = K$ ). Despite promising generalization capability, head clipping suffers from poor in-domain performance. Additionally, we test **repeat-based inverting**, a method that retains  $K - n_i$  samples for each query  $q_i \in \mathcal{D}_{\text{filter}}$  and supplements the deficit through repetition. Compared to RP, repeat-based inverting yields marginally lower performance, emphasizing the importance of data diversity.

**Variants of resampling strategies.** Table 3 (lower) illustrates the performance of GR under different values of intermediate reasoning steps  $S$ . Too small  $S$  ( $S = 2$ ) results in inadequate guidance effectiveness, while too large  $S$  not only increases computational cost but also limits diversity, hindering further performance gains. This highlights the importance of selecting an appropriate value for  $S$ .

### 6.2 Comparison with Additional Baselines

Besides, we conduct a systematic comparison of RL-based methods (which learn from both positive and negative samples) and self-improvement methods (which efficiently leverage self-generated positive samples to achieve robust performance). As shown in Table 4, our strategies achieve comparable or even superior performance.

Specifically, compared with brute-force resampling, which requires  $K$  resampling, our strategies achieve greater efficiency: AR reduces operations to  $K - n_i$  ( $\approx 50\%$  cost reduction), GR requires only

Method	MMPR	MathVerse	We-Math	Avg.
<i>RL-based methods</i>				
DPO (Rafailov et al., 2023)	40.28	24.37	51.67	38.77
GRPO (Shao et al., 2024)	45.17	29.70	49.83	41.56
<i>Self-improvement methods</i>				
vanilla self-improvement	42.79	27.79	52.36	40.98
brute-force resampling	43.04	28.30	52.76	41.36
TC	44.67	28.55	52.13	41.78
RP	45.67	<b>30.08</b>	52.59	<b>42.78</b>
AR	45.67	28.93	52.82	42.47
GR	<b>45.80</b>	27.28	<b>53.91</b>	42.33

Table 4: Performance comparison with additional base-lines on Qwen2-VL-7B-Instruct with  $K = 8$ .

1 resampling on minimal tail data, and both TC and RP eliminate additional resampling overhead entirely. Therefore, our strategies deliver superior performance with enhanced efficiency.

### 6.3 Experiments across Different Domains

To demonstrate cross-domain generalization, we conduct experiments on MMPR-science dataset with InternVL2.5-4B model, and on ThinkLite-VL-Hard-11k dataset (Wang et al., 2025b) with Qwen2-VL-7B-Instruct model (dataset details in Appendix A). The final results are shown in Table 5, illustrating the adaptability of our approach across diverse domains.

Method	MMPR-science	ThinkLite-VL-Hard
w/o self-improvement	60.63	2.99
vanilla self-improvement	64.59	26.93
GR (guided resampling)	64.95	28.11
RP (repeated-based padding)	<b>66.13</b>	<b>29.01</b>

Table 5: Performance on MMPR-science and ThinkLite-VL-Hard-11k datasets with sampling number  $K = 8$ .

### 6.4 Comparison with STaR Setup

In this paper, we choose the retrain-from-base-model approach. To compare it with STaR (Zelikman et al., 2022) setup, we conduct experiments of continuous fine-tuning using Qwen2-VL-7B-Instruct model. As shown in Table 6, the STaR setup is more prone to reaching performance bottlenecks. We further analyze the proportion of data at different difficulty levels under both setups (see Table 7), and observe that Matthew effect is more pronounced in STaR setup.

Method	Round1	Round2	Round3	Round4	Round5
retrain-from-base-model	35.51	39.27	40.78	43.04	42.79
STaR	35.51	40.78	41.28	39.52	40.65

Table 6: Comparison of performance between retrain-from-base-model and STaR.

Method	Level1	Level2	Level3	Level4	Level5
retrain-from-base-model	36.6%	25.9%	20.7%	12.8%	3.9%
STaR	38.5%	26.0%	20.1%	12.0%	3.3%

Table 7: Comparison of difficulty distribution between retrain-from-base-model and STaR. Results indicate that the dominance of head data (Level1) is significantly more pronounced in STaR setup.

### 6.5 Self-improvement as Efficient Sampling

Driven by data distribution shifts during self-improvement, we hypothesize that inter-iteration sampling exhibits greater variance than intra-iteration sampling. To leverage this, we introduce **iterative sampling**, which combines  $K = 8$  samples over 5 iterations (40 samples total) to enhance data diversity. For comparison, we also implement **batch sampling**, which draws all 40 samples at once. Results in Table 12 demonstrate that iterative sampling outperforms batch sampling, supporting our view of self-improvement as an efficient sampling method. Furthermore, applying distribution-reshaping strategies to iterative sampling validates their effectiveness. Notably, RP outperforms both batch sampling and vanilla iterative sampling, with improvements of 3.34 and 1.79 points respectively.

Method	MMPR	MathVerse	We-Math	Avg.
batch sampling	59.10	33.12	52.18	48.13
iterative sampling	64.99	31.47	52.59	49.68
+ TC	66.62	32.23	<b>53.10</b>	50.65
+ RP	<b>69.89</b>	<b>33.88</b>	50.63	<b>51.47</b>

Table 8: Performance comparison between batch sampling and iterative sampling on InternVL2.5-4B model.

### 6.6 Self-correction Benefits Self-improvement

Self-correction has emerged as a promising learning paradigm (Ding and Zhang, 2025; Wu et al., 2025a; Cheng et al., 2025), encouraging deeper reasoning and generating longer chains of thought. Therefore, we investigate its effectiveness for head-tail re-balancing in visual self-improvement (see Appendix G for implementation details). In contrast to the resampling strategies discussed earlier, self-correction refines existing incorrect samples through  $K - n_i$  operations per query, achieving efficiency comparable to AR and fully leveraging the potential of incorrect instances. Results in Figure 5 demonstrate that self-correction effectively counteracts the Matthew effect, yielding substantial performance improvements.

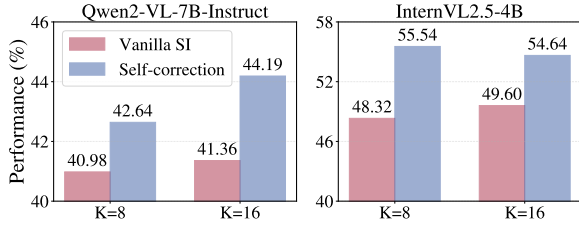


Figure 5: Average performance comparison between vanilla and self-correction in visual self-improvement.

## 6.7 The Power of “Seeing”

To validate whether the final model truly enhances reasoning capabilities, we evaluate its performance on tail data under two settings: with and without image inputs. Figure 6 indicates that the base model exhibits negligible performance difference between these settings, indicating a poor capability to leverage visual information for solving challenging problems. In contrast, vanilla self-improvement leads to a substantial gain in real reasoning performance. Our re-balanced strategies further amplify this increment, with RP strategy demonstrating an 18.8-point advantage when “seeing” images.

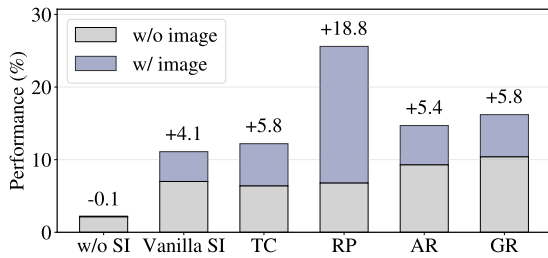


Figure 6: Comparison of tail data performance *with* and *without* images on Qwen2-VL-7B-Instruct at  $K = 8$ .

Additionally, for error type analysis and case study, please refer to Appendix C and D.

## 7 Conclusion

In this work, we identify a critical challenge behind performance bottlenecks in visual self-improvement: the “Matthew effect”, where simple samples in the head progressively dominate successful trajectories, while difficult data in the tail becomes increasingly narrowing. To counteract it, we introduce four effective re-balanced strategies from distribution-reshaping and trajectory-resampling perspectives: threshold clipping, repeat-based padding, adaptive-weighted resampling, and guided resampling. Experimental results demonstrate that these strategies successfully reduce head

dominance and increase tail proportion, thereby improving the performance ceilings. Future work will explore counteracting the Matthew effect on larger models and broader datasets, alongside developing more efficient re-balancing strategies.

## Limitations

Our work provides an in-depth investigation into self-improvement of LVLMs and proposes re-balancing strategies designed to mitigate the Matthew effect. However, several limitations remain: (1) The phenomena we observed are more correlational conclusions and lack explicit causal evidence. (2) Constrained by time and computational resources, our experiments were limited to the Qwen2-VL-7B-Instruct and InternVL2.5-4B models, primarily focusing on mathematical and scientific problems. Consequently, the findings have not been validated on larger models or larger visual reasoning datasets. (3) Our method is tailored to counteract Matthew effect in self-improvement, and has not been adapted for reinforcement learning scenarios. We believe that extending a similar idea to reinforcement learning could also mitigate distributional imbalance in sampling, an avenue we plan to explore in future work.

## Ethical considerations

This work introduces re-balancing strategies designed for Matthew effect in self-improvement of LVLMs. However, we recognize that these methods could be maliciously exploited in other areas. We firmly state that this work is intended for ethical and constructive purposes. Any misuse is strictly against the authors’ original intentions.

## Acknowledgments

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by National Natural Science Foundation of China (No.62521004, 62476061, 62576106, 62376061).

## References

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibao Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Ming-Hsuan Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025. [Qwen2.5-vl technical report](#). *CoRR*, abs/2502.13923.

- Jie Cao and Jing Xiao. 2022. [An augmented benchmark dataset for geometric question answering through dual parallel text encoding](#). In *Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, October 12-17, 2022*, pages 1511–1520. International Committee on Computational Linguistics.
- Qiguang Chen, Libo Qin, Jin Zhang, Zhi Chen, Xiao Xu, and Wanxiang Che. 2024. [M<sup>3</sup>cot: A novel benchmark for multi-domain multi-step multi-modal chain-of-thought](#). *CoRR*, abs/2405.16473.
- Kanzhi Cheng, Yantao Li, Fangzhi Xu, Jianbing Zhang, Hao Zhou, and Yang Liu. 2025. [Vision-language models can self-improve reasoning via reflection](#). In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2025 - Volume 1: Long Papers, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, pages 8876–8892. Association for Computational Linguistics.
- Yihe Deng, Pan Lu, Fan Yin, Ziniu Hu, Sheng Shen, Quanquan Gu, James Y. Zou, Kai-Wei Chang, and Wei Wang. 2024. [Enhancing large vision language models with self-training on image comprehension](#). In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.
- Yi Ding and Ruqi Zhang. 2025. [Sherlock: Self-correcting reasoning in vision-language models](#). *CoRR*, abs/2505.22651.
- Yiwen Ding, Zhiheng Xi, Wei He, Lizhuoyuan Lizhuoyuan, Yitao Zhai, Shi Xiaowei, Xunliang Cai, Tao Gui, Qi Zhang, and Xuanjing Huang. 2025. [Mitigating tail narrowing in LLM self-improvement via socratic-guided sampling](#). In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2025 - Volume 1: Long Papers, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, pages 10627–10646. Association for Computational Linguistics.
- Elvis Dohmatob, Yunzhen Feng, Pu Yang, François Charton, and Julia Kempe. 2024. [A tale of tails: Model collapse as a change of scaling laws](#). In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.
- Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023. [Large language models can self-improve](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 1051–1068. Association for Computational Linguistics.
- Chunyang Jiang, Chi-Min Chan, Wei Xue, Qifeng Liu, and Yike Guo. 2025. [Importance weighting can help large language models self-improve](#). In *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 24257–24265. AAAI Press.
- Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Min Joon Seo, Hannaneh Hajishirzi, and Ali Farhadi. 2016. [A diagram is worth a dozen images](#). In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*, volume 9908 of *Lecture Notes in Computer Science*, pages 235–251. Springer.
- Deyang Kong, Qi Guo, Xiangyu Xi, Wei Wang, Jingang Wang, Xunliang Cai, Shikun Zhang, and Wei Ye. 2025. [Rethinking the sampling criteria in reinforcement learning for LLM reasoning: A competence-difficulty alignment perspective](#). *CoRR*, abs/2505.17652.
- Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D. Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei M. Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal M. P. Behbahani, and Aleksandra Faust. 2025. [Training language models to self-correct via reinforcement learning](#). In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.
- Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. [Efficient memory management for large language model serving with pagedattention](#). In *Proceedings of the 29th Symposium on Operating Systems Principles, SOSP 2023, Koblenz, Germany, October 23-26, 2023*, pages 611–626. ACM.
- Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. 2021. [Inter-gps: Interpretable geometry problem solving with formal language and symbolic reasoning](#). In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 6774–6786. Association for Computational Linguistics.
- Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. 2022. [Learn to explain: Multimodal reasoning via thought chains for science question answering](#). In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

- Robert K Merton. 1968. The matthew effect in science: The reward and communication systems of science are considered. *Science*, 159(3810):56–63.
- Roweida Mohammed, Jumanah Rawashdeh, and Malak Abdullah. 2020. Machine learning with oversampling and undersampling techniques: Overview study and experimental results.
- OpenAI. 2023. [GPT-4 technical report](#). *CoRR*, abs/2303.08774.
- Yi Peng, Chris, Xiaokun Wang, Yichen Wei, Jiangbo Pei, Weijie Qiu, Ai Jian, Yunzhuo Hao, Jiachun Pan, Tianyidan Xie, Li Ge, Rongxian Zhuang, Xuchen Song, Yang Liu, and Yahui Zhou. 2025. [Skywork R1V: pioneering multimodal reasoning with chain-of-thought](#). *CoRR*, abs/2504.05599.
- Runqi Qiao, Qiuna Tan, Guanting Dong, Minhui Wu, Chong Sun, Xiaoshuai Song, Zhuoma Gongque, Shanglin Lei, Zhe Wei, Miaoquan Zhang, Runfeng Qiao, Yifan Zhang, Xiao Zong, Yida Xu, Muxi Diao, Zhimin Bao, Chen Li, and Honggang Zhang. 2024. [We-math: Does your large multimodal model achieve human-like mathematical reasoning?](#) *CoRR*, abs/2407.01284.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. [Direct preference optimization: Your language model is secretly a reward model](#). In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.
- Min Joon Seo, Hannaneh Hajishirzi, Ali Farhadi, Oren Etzioni, and Clint Malcolm. 2015. [Solving geometry problems: Combining text and diagram interpretation](#). In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1466–1476. The Association for Computational Linguistics.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. [Deepseekmath: Pushing the limits of mathematical reasoning in open language models](#). *CoRR*, abs/2402.03300.
- Iliia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross J. Anderson, and Yarin Gal. 2024. [AI models collapse when trained on recursively generated data](#). *Nat.*, 631(8022):755–759.
- Linzhuang Sun, Hao Liang, Jingxuan Wei, Bihui Yu, Tianpeng Li, Fan Yang, Zenan Zhou, and Wentao Zhang. 2025. [Mm-verify: Enhancing multimodal reasoning with chain-of-thought verification](#). *CoRR*, abs/2502.13383.
- Naoto Tanji and Toshihiko Yamasaki. 2025. [Iterative self-improvement of vision language models for image scoring and self-explanation](#). *CoRR*, abs/2506.02708.
- Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, Dehao Zhang, Enming Yuan, Enzhe Lu, Fengxiang Tang, Flood Sung, Guangda Wei, Guokun Lai, and 75 others. 2025. [Kimi k1.5: Scaling reinforcement learning with llms](#). *CoRR*, abs/2501.12599.
- Yuxuan Tong, Xiwen Zhang, Rui Wang, Ruidong Wu, and Junxian He. 2024. [Dart-math: Difficulty-aware rejection tuning for mathematical problem-solving](#). In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.
- Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024a. [Qwen2-vl: Enhancing vision-language model’s perception of the world at any resolution](#). *CoRR*, abs/2409.12191.
- Weiyun Wang, Zhe Chen, Wenhai Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Jinguo Zhu, Xizhou Zhu, Lewei Lu, Yu Qiao, and Jifeng Dai. 2024b. [Enhancing the reasoning ability of multimodal large language models via mixed preference optimization](#). *CoRR*, abs/2411.10442.
- Xiyao Wang, Jiu Hai Chen, Zhaoyang Wang, Yuhang Zhou, Yiyang Zhou, Huaxiu Yao, Tianyi Zhou, Tom Goldstein, Parminder Bhatia, Taha A. Kass-Hout, Furong Huang, and Cao Xiao. 2025a. [Enhancing visual-language modality alignment in large vision language models via self-improvement](#). In *Findings of the Association for Computational Linguistics: NAACL 2025, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, pages 268–282. Association for Computational Linguistics.
- Xiyao Wang, Zhengyuan Yang, Chao Feng, Hongjin Lu, Linjie Li, Chung-Ching Lin, Kevin Lin, Furong Huang, and Lijuan Wang. 2025b. [Sota with less: Mcts-guided sample selection for data-efficient visual reasoning self-improvement](#). *CoRR*, abs/2504.07934.
- Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. [Self-instruct: Aligning language models with self-generated instructions](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 13484–13508. Association for Computational Linguistics.
- Xueqing Wu, Yuheng Ding, Bingxuan Li, Pan Lu, Da Yin, Kai-Wei Chang, and Nanyun Peng. 2025a. [Visco: Benchmarking fine-grained critique and correction towards self-improvement in visual reasoning](#). In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 9527–9537.

- Zijian Wu, Jinjie Ni, Xiangyan Liu, Zichen Liu, Hang Yan, and Michael Qizhe Shieh. 2025b. [Synthrl: Scaling visual reasoning with verifiable data synthesis](#). *CoRR*, abs/2506.02096.
- Zhiheng Xi, Wenxiang Chen, Boyang Hong, Senjie Jin, Rui Zheng, Wei He, Yiwen Ding, Shichun Liu, Xin Guo, Junzhe Wang, Honglin Guo, Wei Shen, Xiaoran Fan, Yuhao Zhou, Shihan Dou, Xiao Wang, Xinbo Zhang, Peng Sun, Tao Gui, and 2 others. 2024. [Training large language models for reasoning through reverse curriculum reinforcement learning](#). In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.
- Tianyi Xiong, Xiyao Wang, Dong Guo, Qinghao Ye, Haoqi Fan, Quanquan Gu, Heng Huang, and Chunyuan Li. 2025. [Llava-critic: Learning to evaluate multimodal models](#). In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 13618–13628.
- Boyang Xue, Qi Zhu, Hongru Wang, Rui Wang, Sheng Wang, Hongling Xu, Fei Mi, Yasheng Wang, Lifeng Shang, Qun Liu, and Kam-Fai Wong. 2025. [DAST: difficulty-aware self-training on large language models](#). *CoRR*, abs/2503.09029.
- Zhengyuan Yang, Jianfeng Wang, Linjie Li, Kevin Lin, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. 2023. [Idea2img: Iterative self-refinement with gpt-4v \(ision\) for automatic image design and generation](#). *arXiv preprint arXiv:2310.08541*.
- Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. 2022. [Star: Bootstrapping reasoning with reasoning](#). In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.
- Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D Goodman. 2024. [Star: Self-taught reasoner bootstrapping reasoning with reasoning](#). In *Proc. the 36th International Conference on Neural Information Processing Systems*, volume 1126.
- Di Zhang, Jingdi Lei, Junxian Li, Xunzhi Wang, Yujie Liu, Zonglin Yang, Jiatong Li, Weida Wang, Suorong Yang, Jianbo Wu, Peng Ye, Wanli Ouyang, and Dongzhan Zhou. 2025a. [Critic-v: Vlm critics help catch vlm errors in multimodal reasoning](#). In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 9050–9061.
- Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, Peng Gao, and Hongsheng Li. 2024. [MATHVERSE: does your multi-modal LLM truly see the diagrams in visual math problems?](#) In *Computer Vision - ECCV 2024 - 18th European Conference, Milan, Italy, September 29-October 4, 2024, Proceedings, Part VIII*, volume 15066 of *Lecture Notes in Computer Science*, pages 169–186. Springer.
- Renrui Zhang, Xinyu Wei, Dongzhi Jiang, Ziyu Guo, Yichi Zhang, Chengzhuo Tong, Jiaming Liu, Aojun Zhou, Shanghang Zhang, Peng Gao, and Hongsheng Li. 2025b. [MAVIS: mathematical visual instruction tuning with an automatic data engine](#). In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.
- Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. 2025. [SWIFT: A scalable lightweight infrastructure for fine-tuning](#). In *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 29733–29735. AAAI Press.
- Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, Zhangwei Gao, Erfei Cui, Xuehui Wang, Yue Cao, Yangzhou Liu, Xingguang Wei, Hongjie Zhang, Haomin Wang, Weiye Xu, and 32 others. 2025. [InternV3: Exploring advanced training and test-time recipes for open-source multimodal models](#). *CoRR*, abs/2504.10479.

## A Experimental Details

### A.1 Dataset Details

We adopt MMPR (Wang et al., 2024b) as our primary dataset. From it, we randomly extract 7,980 mathematical reasoning samples (Lu et al., 2021; Cao and Xiao, 2022; Seo et al., 2015) to construct MMPR-mini, with 7,183 for training and 797 for in-domain testing. Specifically, we select queries from Geometry3K (Lu et al., 2021), GeoQA+ (Cao and Xiao, 2022), and GEOS (Seo et al., 2015) respectively, randomly sampling 10% of data from each dataset to construct the test set. MMPR-mini primarily focuses on mathematical visual reasoning problems, including multiple-choice questions, open-ended questions, and other formats, containing only queries with their corresponding ground truth without CoT reasoning trajectories.

To demonstrate the generalization of our method to other tasks, we also extract the MMPR-science subset (comprising AI2D (Kembhavi et al., 2016), ScienceQA (Lu et al., 2022), and M3CoT (Chen et al., 2024)) from MMPR. Using a similar methodology, we partition this data into a training set of 10,009 samples and a test set of 1,110 samples.

Similarly, we also leverage the ThinkLite-VL-Hard-11k multimodal dataset (Wang et al., 2025b), which covers three key visual tasks: mathematical

reasoning, natural image understanding, and chart comprehension. We randomly split the dataset into a training set containing 9,928 instances and a test set containing 1,103 instances.

Additionally, for out-of-domain (OOD) evaluation, we select two widely-adopted mathematical visual reasoning datasets: MathVerse (Zhang et al., 2024) and We-Math (Qiao et al., 2024), comprising 788 and 1,740 queries respectively. All datasets we utilized are open-source.

## A.2 Difficulty Level Categorization

Using Qwen2-VL-7B-Instruct, we perform 64-shot sampling on each query, and categorize the queries into 5 difficulty levels based on their pass@64 performance. This classification prioritizes a balanced data distribution across all levels, with data difficulty ascending from level 1 to level 5.

To verify the reasonableness of the classification, we used InternVL2.5-4B model to categorize the data via the same pass@64 approach. Experimental results show that the difference in difficulty levels defined by the two models is 0.75 on average, indicating that these levels remain generally consistent across different models.

## A.3 Prompt Details

In this work, we use the unified prompt template (see Figure 7) in the phase of training, sampling and testing across varying datasets.

Your task is to answer the question below. Give step by step reasoning before you answer, and when you're ready to answer, please use the format ``Final answer: ..``.

Question: {Question}

Your step by step reasoning:

Figure 7: Prompt for training, sampling and testing.

# B In-Depth Analysis of Response Length

## B.1 Average Response Length Degradation in Matthew Effect

In Section 3.3, we observe the shorter average response length for difficult problems in the later stages of iteration. Based on a meticulous analysis of the generated data, we offer the following two explanations:

First, in vanilla self-improvement (vanilla SI), during the early stages of training, simple data typically has shorter response lengths, while difficult

data tends to have longer responses. However, due to the distribution imbalance (where difficult data is underrepresented), as iterations progress, the model tends to produce shorter responses even for difficult data.

Second, in the later stages of training, due to the higher uncertainty in the model’s reasoning process for difficult problems, it becomes more inclined to output answers directly. To validate this, we examine the proportion of extremely short responses (less than 10 tokens) among the correct outputs after 5 iterations of vanilla self-improvement. The results in Table 9 show that as the difficulty level increases, the proportion of extremely short responses also increases, significantly lowering the overall average response length.

Method	Level 1	Level2	Level3	Level 4	Level5
vanilla SI	5.60%	13.72%	15.63%	20.47%	27.33%
ours	1.33%	3.13%	4.31%	5.50%	6.50%

Table 9: The proportion of extremely short responses of different difficulty levels.

## B.2 Our Methods Mitigate the Degradation of Average Response Length

Taking the results of the Qwen2-VL-7B-Instruct model with as an example, we examine the response lengths of correct data after 5 iterations. The average length statistics for different difficulty levels and all data under the vanilla self-improvement and ours methods are shown in Table 10.

We also perform a statistical analysis on the proportion of extremely short data. As shown in Table 9, ours methods significantly mitigate the impact of extremely short data on the overall results.

Method	Level 1	Level2	Level3	Level 4	Level5	All Data
vanilla SI	214.07	204.98	197.57	182.32	149.95	201.48
ours	225.68	251.34	253.42	250.29	228.51	241.91

Table 10: Average response length of different difficulty levels on Qwen2-VL-7B-Instruct with sampling number  $K = 8$ .

## C Error Type Analysis

First, we analyze the incorrect data and categorize it into the following error types:

- **No Reasoning Process (NP):** Models provide answers directly without demonstrating any step-by-step reasoning or explanation process.

- **Comprehension Error (CE):** Models exhibit misunderstanding of either the question content or the visual information presented in the image.
- **Knowledge Error (KE):** Models employ incorrect formulas, theorems, or other factual information.
- **Logic Error (LE):** Models generate flawed reasoning steps, perform incorrect calculations, or establish faulty cause-and-effect relationships.
- **Format Error (FE):** Models produce responses whose format does not meet the specified requirements.

Then we utilize GPT-4.1 (OpenAI, 2023) for extensive error type classification, setting temperature=0.1 and permitting up to 3 outputs. The prompt template is illustrated in Figure 8, with classification outcomes detailed in Table 11.

Please carefully analyze the following multimodal conversation question and incorrect response, and determine its error type.

Question: {Question}  
 Incorrect Response: {Response}

Now please select the most appropriate error type from the following options:

A. No Reasoning Process - Providing answers directly without showing any step-by-step reasoning or explanation  
 B. Comprehension Error - Misunderstanding of the question or image content  
 C. Knowledge Error - Using incorrect formulas, theorems, or other factual information  
 D. Logic Error - Flawed reasoning steps, incorrect calculations, or faulty cause-effect relationships  
 E. Format Error - Response format does not meet requirements

Please return only the corresponding letter option (A, B, C, D, or E) without any other content.

Figure 8: Prompt for determining error type.

The results reveal several key findings: (1) The overall distribution of error types varies significantly across different models. For instance, with the Qwen2-VL-7B-Instruct model, the number of samples directly providing final answers increases substantially, particularly under the TC strategy, which reduces the quantity of data and further amplifies the proportion of direct-answer samples, making “No Reasoning Process” the most frequent error type in TC’s

Model	Method	NP	CE	KE	LE	FE
Qwen2-VL	w/o SI	4	286	24	286	2
	Vanilla SI	36	189	21	216	1
	TC	182	137	16	91	0
	RP	34	152	10	222	0
	AR	77	185	13	170	2
	GR	153	128	8	124	2
InternVL2.5	w/o SI	0	227	13	176	0
	Vanilla SI	1	132	12	135	2
	TC	0	150	12	123	0
	RP	14	123	9	116	0
	AR	7	137	12	113	0
	GR	0	126	14	117	1

Table 11: Error type analysis. NP, CE, KE, LE and FE denote No Reasoning Process, Comprehension Error, Knowledge Error, Logic Error, and Format Error, respectively. SI, TC, RP, AR, and GR denote self-improvement, threshold clipping, repeat-based padding, adaptive-weighted resampling, and guided resampling, respectively.

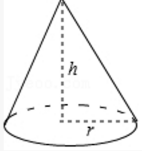
incorrect data. In contrast, the InternVL2.5-4B model exhibits few “No Reasoning Process” errors. (2) Even without self-improvement, format errors occur with extremely low frequency, indicating that the self-improvement process truly enhances models’ reasoning capabilities rather than simple instruction-following abilities for format. (3) Self-improvement achieves significant improvements mainly in comprehension errors and logic errors, substantially reducing the occurrence of both error types. (4) Compared to vanilla self-improvement, our re-balanced strategies enhance models’ capabilities in question and image comprehension as well as reasoning logic.

In rare instances, the model consistently returns “T”; these cases are omitted from the table. Manual examination of these cases reveals that the responses are accurate but were misclassified as incorrect due to our exact-match rules, with an illustrative example provided in Figure 9.

## D Case Study

Our main results show that repeat-based padding and guided resampling achieve superior performance among our proposed strategies. Therefore, we showcase examples of these two strategies in Figure 10 and 11 respectively. Both cases reveal that vanilla self-improvement exhibits limited capabilities in visual comprehension and geometric element understanding. For instance, in Figure 10, vanilla self-improvement incorrectly determines the relationship between  $\angle AOC$  and  $\angle BDC$ , while

**Question:**  
As shown in the figure, the radius  $R$  of the cone base is 6, and the height  $H$  is 8. The lateral area of the cone is ( ).



**Ground Truth:**  $60\pi$

**Response:**  
To find the lateral area of the cone, we need to use the formula for the lateral surface area of a cone, which is given by:

$$A = \pi \cdot r \cdot l$$

where  $r$  is the radius of the base and  $l$  is the slant height of the cone.

First, we need to find the slant height  $l$ . The slant height can be found using the Pythagorean theorem in the right triangle formed by the radius  $r$ , the height  $H$ , and the slant height  $l$ :  $l = \sqrt{r^2 + H^2}$

Given: Radius  $r = 6$ , Height  $H = 8$

Substitute the given values into the formula for the slant height:

$$l = \sqrt{6^2 + 8^2}$$

$$l = \sqrt{36 + 64}$$

$$l = \sqrt{100}$$

$$l = 10$$

Now that we have the slant height  $l = 10$ , we can find the lateral area  $A$

$$A = \pi \cdot r \cdot l$$

$$A = \pi \cdot 6 \cdot 10$$

$$A = 60\pi$$

Final answer:  $60\pi$

Figure 9: A case misclassified as incorrect due to exact-match rules, where  $\pi$  and  $\pi$  appear in different formats.

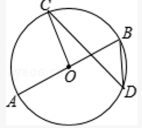
in Figure 11 demonstrates the confusion of height  $h$  with radius  $r$ , resulting in final errors. Both repeat-based padding and guided resampling successfully address these problems.

## E Matthew Effect Mitigation

In Section 5, we analyzed the effectiveness of our proposed rebalanced strategies in mitigating the Matthew effect and their performance on visual reasoning tasks using Qwen2-VL-7B-Instruct at  $K = 16$ . Here, we present three additional configurations—Qwen2-VL-7B-Instruct at  $K = 8$ , InternVL2.5-4B at  $K = 8$ , and InternVL2.5-4B at  $K = 16$ —in Figure 12, Figure 13 and Figure 14.

Results demonstrate that our proposed rebalancing strategies effectively mitigate the Matthew effect in self-improvement across different experimental settings. Overall, the repeat-based padding (RP) method exhibits the best mitigation performance. Among the two trajectory-resampling strategies, guided resampling (GR) generally outperforms adaptive-weighted resampling (AR), which aligns with the performance patterns

**Question:**  
As shown in the figure,  $AB$  is the diameter of circle  $O$ . If  $\angle BDC = 40^\circ$ , then the measure of  $\angle AOC$  is ( ).



**Vanilla:**  
To find the measure of  $\angle AOC$ , we can use the properties of angles in a circle. Since  $AB$  is the diameter of circle  $O$ ,  $\angle ACB$  is a right angle ( $90^\circ$ ) because the angle subtended by a diameter in a semicircle is always  $90^\circ$ . Given that  $\angle BDC = 40^\circ$ , we can use the fact that the angle subtended by the same arc at the center of the circle is twice the angle subtended by the same arc at any point on the circumference. Therefore,  $\angle AOC = 2 * \angle BDC = 2 * 40^\circ = 80^\circ$ .  
Final answer:  $80^\circ$ .

**RP:**  
Since  $AB$  is the diameter of circle  $O$ ,  $\angle ACB$  is a right angle ( $90^\circ$ ). Therefore,  $\angle ABC = 90^\circ - \angle BDC = 90^\circ - 40^\circ = 50^\circ$ . Since  $\angle AOC$  is the central angle corresponding to  $\angle ABC$ ,  $\angle AOC = 2\angle ABC = 2 * 50^\circ = 100^\circ$ . Therefore, the measure of  $\angle AOC$  is  $100^\circ$ .  
Final answer:  $100^\circ$ .

Figure 10: An example of repeat-based padding with Qwen2-VL-7B-Instruct model on MMPr-mini test set. Compared to vanilla self-improvement, repeat-based padding successfully finds the relationship between  $\angle AOC$  and  $\angle ABC$ , reaching the right answer.

of these strategies on the test sets.

## F Batch Sampling and Iterative Sampling

In Section 6, we used InternVL2.5-4B at  $K = 8$  as an example to discuss that self-improvement can be viewed as an efficient sampling approach. Additionally, we conducted similar experiments on Qwen2-VL-7B-Instruct at  $K = 16$  and observed similar phenomena. Specifically, we combine  $K = 16$  samples across 5 iterations (totaling 80 samples) as iterative sampling, while batch sampling involves directly sampling 80 times from the base model. As shown in Table 12, iterative sampling delivers improvements of 5.65 points on the in-domain test set and 3.17 points on average. Nevertheless, our distribution-reshaping strategies do not consistently perform well under the iterative sampling perspective. This might be because for  $K = 16$ , TC with  $\rho_0 = 4$  significantly reduces the data quantity, while RP leads to excessive repetition, both of which may hinder advantages in efficient sampling scenarios.

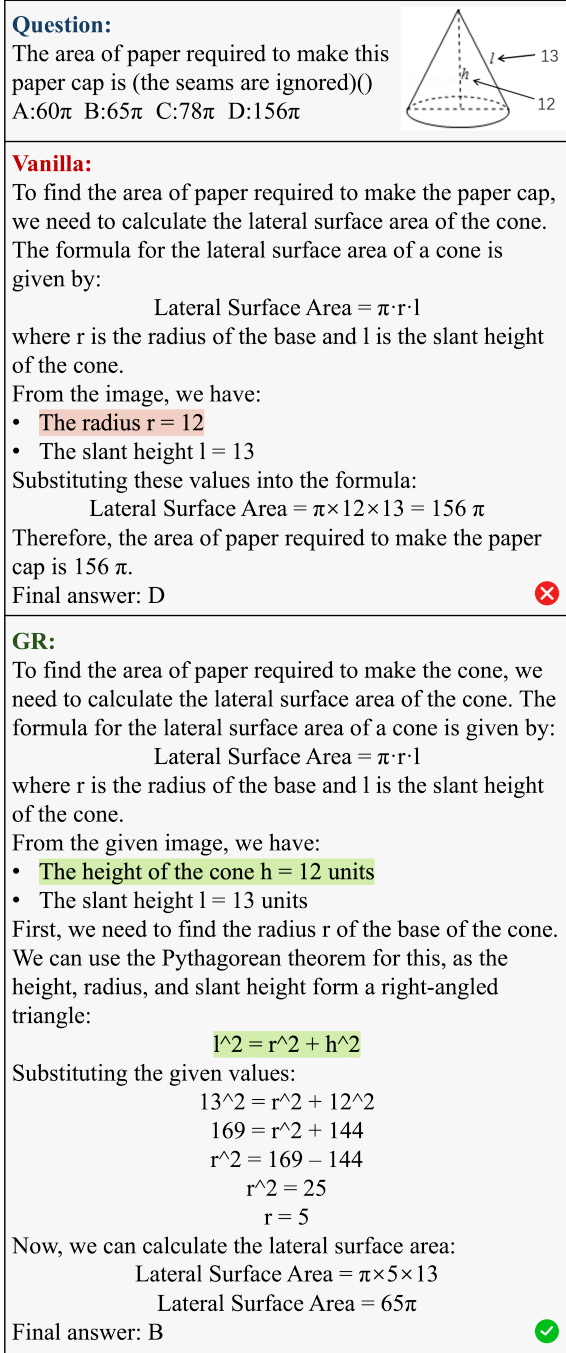


Figure 11: An example of guided resampling with InternVL2.5-4B model on MathVerse test set. In this case, vanilla self-improvement demonstrates the confusion of height  $h$  with radius  $r$ , while guided resampling addresses this problem successfully.

## G Applying Self-correction to Self-improvement

Given self-correction capabilities shown in Figure 15, we explore applying self-correction to self-improvement for head-tail re-balancing, incorporating the refined outputs into subsequent training iterations.

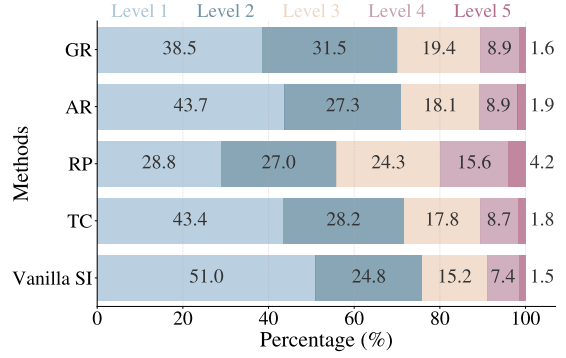


Figure 12: Data distribution of difficulty levels in successful trajectories under different strategies with Qwen2-VL-7B-Instruct at  $K = 8$ .

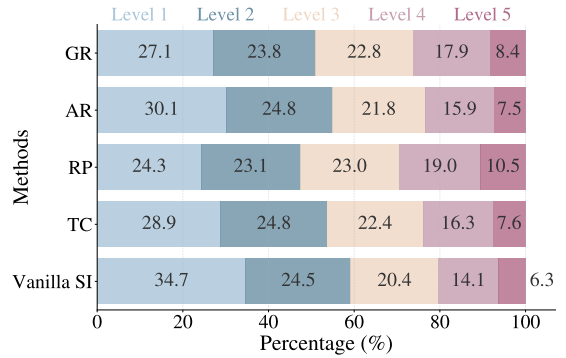


Figure 13: Data distribution of difficulty levels in successful trajectories under different strategies with InternVL2.5-4B at  $K = 8$ .

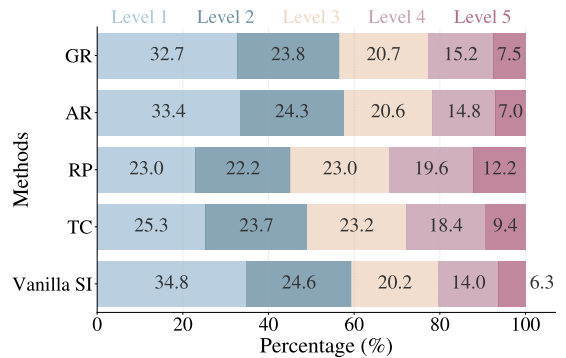


Figure 14: Data distribution of difficulty levels in successful trajectories under different strategies with InternVL2.5-4B at  $K = 16$ .

Formally, we first define the set of incorrect samples as:

$$\mathcal{D}_{\text{discard}}^{(t)} = \{(q_i, \hat{r}_{i,k}^{(t)}) \in \mathcal{D}_{\text{sample}}^{(t)} \mid r f(q_i, a_i, \hat{a}_{i,k}^{(t)}) = 0\}.$$

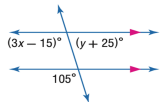
For  $\hat{r}_i \in \mathcal{D}_{\text{discard}}^{(t)}$ , we design the self-correction prompt  $p$  to resample  $\tilde{r}_i \sim \mathcal{M}_{t-1}(\cdot | q_i, \hat{r}_i, p)$  and form the resample dataset:

$$\mathcal{D}_{\text{self-correction}}^{(t)} = \{(q_i, \hat{r}_{i,k}^{(t)}, p, \tilde{r}_{i,k}^{(t)}), (q_i, \tilde{r}_{i,k}^{(t)}) \mid n_i < K\},$$

Method	MMPR	MathVerse	We-Math	Avg.
batch sampling	38.52	26.52	49.37	38.14
iterative sampling	<b>44.17</b>	<b>28.68</b>	51.15	<b>41.33</b>
+ TC	41.78	27.41	<b>52.24</b>	40.48
+ RP	42.03	26.52	48.45	39.00

Table 12: Performance comparison between batch sampling and iterative sampling on Qwen2-VL-7B-Instruct model.

**Question:**  
Find the value of the variable x in the figure.



**Incorrect sample:**  
Step 1: Identify the type of angles formed by the intersecting lines. The figure shows two intersecting lines, creating four angles at the intersection.  
Step 2: Determine the relationship between the angles. Since the lines are parallel, the angles on opposite sides of the intersection (opposite angles) are congruent. Therefore, we have:  
 $(3x - 15)^\circ = (y + 25)^\circ$   
Step 3: Solve for x. To find the value of x, we can set up an equation based on the congruence of the opposite angles and solve for x.  
 $(3x - 15)^\circ = (y + 25)^\circ$   
 $3x - 15 = y + 25$   
Step 4: Substitute the known value of y. We are given that one of the angles is  $105^\circ$ . Since the angles on opposite sides of the intersection are congruent, we can substitute y with  $105^\circ$ .  
 $3x - 15 = 105 + 25$   
 $3x - 15 = 130$   
Step 5: Solve for x.  
 $3x - 15 = 130$   
 $3x = 145$   
 $x = 48.333...$  ❌  
Final answer: The value of the variable x is approximately 48.333.

**Correction:**  
The error in the solution above is that we assumed y to be  $105^\circ$  without considering the other angle of  $105^\circ$ . We should use the given angle of  $105^\circ$  to solve for x.  
**Corrected Step 4:** We are given two angles, one of which is  $105^\circ$ . Since the angles on opposite sides of the intersection are congruent, we can set up the equation based on the congruence of the opposite angles:  
 $(3x - 15)^\circ = 105^\circ$   
 $3x - 15 = 105$   
Solving these equations will give us the value of x.  
**Corrected Step 5:** Solve for x in the equation:  
 $3x - 15 = 105$   
 $3x = 120$   
 $x = 40$   
Therefore, the correct value of the variable x is 40.  
Final answer: The value of the variable x is 40. ✅

Figure 15: An example of self-correction capabilities of Qwen2-VL-7B-Instruct model.

where we filter out samples with failed corrections and insufficient CoT reasoning length. Finally, we obtain the training dataset as following:

$$\mathcal{D}_{\text{train-BP}}^{(t)} = \mathcal{D}_{\text{train}}^{(t)} \cup \mathcal{D}_{\text{self-correction}}^{(t)}$$

The prompt  $p$  we used (Kumar et al., 2025) is presented in Figure 16.

Your task is to answer the question below. Give step by step reasoning before you answer, and when you're ready to answer, please use the format ``Final answer: ..``.

Question: {Question}

Your step by step reasoning: {CoT Reasoning}

There might be an error in the solution above because of lack of understanding of the question. Please correct the error, if any, and rewrite the solution.

Your step by step reasoning:

Figure 16: Prompt for self-correction.

Results in Section 6.6 demonstrate that applying self-correction to self-improvement yields substantial performance gains. We provide an illustrative example in Figure 17, where iterative self-correction enables the model to engage in deeper reasoning through generating longer thinking chains, ultimately correcting errors in vanilla self-improvement.

Notably, in addition to verifying final results, we also filter out data without CoT reasoning process. The results in the table 13 indicate that CoT length filtering improved the quality of tail data, demonstrating superior overall performance.

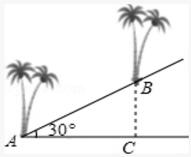
Method	MMPR	MathVerse	We-Math	Avg.
w/o CoT filtering	44.04	29.44	<b>55.11</b>	42.87
w/ CoT filtering	<b>46.67</b>	<b>30.96</b>	54.94	<b>44.19</b>

Table 13: Performance comparison between *with* and *without* CoT filtering on Qwen2-VL-7B-Instruct model.

## H The Use of AI Assistants

AI assistants are utilized in this manuscript for partial grammatical checks and language polishing. The authors are fully responsible for the final content.

**Question:**  
As shown in the figure, if two trees are to be planted on a slope with an incline of  $30^\circ$ , the horizontal distance AC between them is 6m, and the slope length AB between the two trees is ( )  
A. 12m B.  $3\sqrt{3}$ m C.  $4\sqrt{3}$ m D.  $12\sqrt{3}$ m



**Vanilla:**  
To solve this problem, we need to use trigonometric relationships in a right triangle. The slope forms a right triangle with the horizontal distance AC and the slope length AB. The angle of inclination is  $30^\circ$ .  
**Step 1:** Identify the right triangle and the given values. The right triangle has a horizontal side (AC) of 6m and an angle of  $30^\circ$  at point A.  
**Step 2:** Use the tangent function to find the slope length AB. The tangent of an angle in a right triangle is the ratio of the opposite side to the adjacent side. In this case,  $\tan(30^\circ) = AB / AC$ .  
**Step 3:** Substitute the given values into the equation.  $\tan(30^\circ) = AB / 6m$ .  
**Step 4:** Solve for AB.  $\tan(30^\circ) \approx 0.577$ .  $0.577 = AB / 6m$ .  $AB \approx 0.577 * 6m$ .  $AB \approx 3.462m$ .  
**Step 5:** Compare the calculated value with the given options. The closest option to 3.462m is  $3\sqrt{3}$ m.  
**Final answer:** B.  $3\sqrt{3}$ m

**Self-correction:**  
To solve this problem, we need to use trigonometric relationships in a right triangle. The slope AB forms a right triangle with the horizontal distance AC and the vertical height BC.  
Given:  
• The incline angle  $\theta = 30^\circ$   
• The horizontal distance AC = 6 meters  
We need to find the length of the slope AB.  
**Step 1:** Identify the trigonometric function relevant to the problem. Since we have the angle and the adjacent side (horizontal distance), we use the tangent function, which relates the opposite side (vertical height) to the adjacent side (horizontal distance).  
**Step 2:** Use the tangent function:  $\tan \theta = \text{opposite} / \text{adjacent}$   
**Step 3:** Substitute the given values:  $\tan(30^\circ) = BC / AC$   
**Step 4:** Solve for BC:  $BC = AC \cdot \tan(30^\circ)$   
**Step 5:** Calculate  $\tan(30^\circ)$ :  $\tan(30^\circ) = 1 / \sqrt{3}$   
**Step 6:** Substitute  $\tan(30^\circ)$  into the equation:  $BC = 6 \cdot (1 / \sqrt{3})$   
**Step 7:** Simplify the expression:  $BC = 6 / \sqrt{3}$   
**Step 8:** Rationalize the denominator:  $BC = (6 / \sqrt{3}) \cdot (\sqrt{3} / \sqrt{3}) = 6\sqrt{3} / 3 = 2\sqrt{3}$   
**Step 9:** Use the Pythagorean theorem to find AB:  $AB = \sqrt{AC^2 + BC^2}$   
**Step 10:** Substitute the values of AC and BC:  $AB = \sqrt{6^2 + (2\sqrt{3})^2}$   
**Step 11:** Simplify the expression:  $AB = \sqrt{36 + 4 \cdot 3} = \sqrt{36 + 12} = \sqrt{48} = 4\sqrt{3}$   
**Final answer:** C.  $4\sqrt{3}$ m

Figure 17: An example of self-correction with InternVL2.5-4B model on MMPR-mini test set. In this case, the model of vanilla self-improvement made identification errors during tangent calculations, indicating deficient visual comprehension capabilities. In contrast, the self-correction method successfully addresses this problem and performs detailed computations to reach the correct answer.