Mitigating Tail Narrowing in LLM Self-Improvement via Socratic-Guided Sampling

Yiwen Ding[1*‡], **Zhiheng Xi**[1*], **Wei He**[1], **Zhuoyuan Li**[5], **Yitao Zhai**[2],
Xiaowei Shi[2], **Xunliang Cai**[2], **Tao Gui**[3†], **Qi Zhang**[1,4], **Xuanjing Huang**[1,4†]

[1] School of Computer Science, Fudan University [2] Meituan
[3] Institute of Modern Languages and Linguistics, Fudan University
[4] Key Laboratory of Intelligent Information Processing, Fudan University
[5] Macau University of Science and Technology
{ywding23,zhxi22}@m.fudan.edu.cn, {tgui, xjhuang}@fudan.edu.cn

Abstract

Self-improvement methods enable large language models (LLMs) to generate solutions themselves and iteratively train on filtered, high-quality rationales. This process proves effective and reduces the reliance on human supervision in LLMs' reasoning, but the performance soon plateaus. We delve into the process and find that models tend to over-sample on easy queries and under-sample on queries they have yet to master. As iterations proceed, this imbalance in sampling is exacerbated, leading to a long-tail distribution where solutions to difficult queries almost diminish. This phenomenon limits the performance gain of self-improving models. A straightforward solution is brute-force sampling to balance the distribution, which significantly raises computational costs. In this paper, we introduce **G**uided **S**elf-**I**mprovement (**GSI**), a strategy aimed at improving the efficiency of sampling challenging heavy-tailed data. It leverages Socratic-style guidance signals to help LLM reasoning with complex queries, reducing the exploration effort and minimizing computational overhead. Experiments on four models across diverse mathematical tasks show that GSI strikes a balance between performance and efficiency, while also being effective on held-out tasks[1].

1 Introduction

Large language models (LLMs) have demonstrated impressive ability in performing complex reasoning tasks (Wei et al., 2022b; Kojima et al., 2022; Zhao et al., 2023). While fine-tuning models on curated data can further boost performance, it relies heavily on human supervision, limiting scalability and generalization (Cobbe et al., 2021). To address this, the "self-improvement" paradigm emerges, where

---

[*] Equal contribution. [‡] Work done during internship at Meituan. [†] Corresponding author.

[1] Codes are publicly available at https://github.com/Yiwen-Ding/Guided-Self-Improvement.
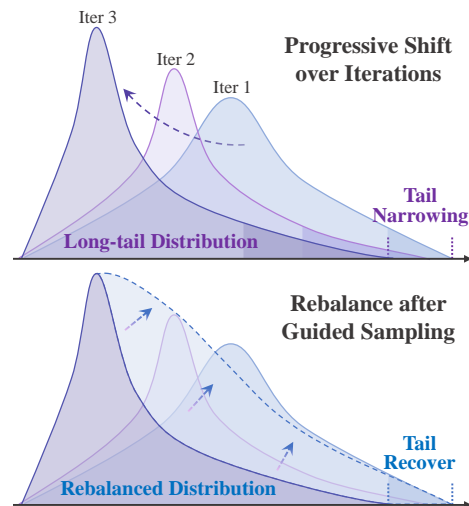

Figure 1: Illustration of distribution during the self-improvement sampling process. **Top:** The long-tail effect intensifies with iterative training on self-generated data. The low-probability data begins to diminish, leading to *tail narrowing*. **Bottem:** Guided sampling balances the distribution by improving tail data sampling efficiency.

models generate multiple reasoning paths, filter out incorrect responses and fine-tune themselves on their own outputs without human intervention (Zelikman et al., 2022; Gülçehre et al., 2023; Huang et al., 2023; Singh et al., 2024; Yuan et al., 2024).

Despite the benefits of self-improvement, its performance typically reaches a ceiling after a few iterations (Wu et al., 2024). We perform preliminary experiments (§ 4) and find that in reasoning tasks, the most significant gains from self-improvement occur in the first iteration, while subsequent iterations encounter performance bottlenecks or even degradation (Figure 2). Similar performance bottlenecks in synthetic data have also been observed in text generation (Shumailov et al., 2023) and image synthesis (Alemohammad et al., 2024).

Further, we delve into the self-improvement process and conduct an in-depth analysis (Figure 3) to investigate the underlying causes behind the per-

formance bottlenecks. On the one hand, complex problems with lengthy reasoning chains tend to amplify hallucinations, making it difficult for models to explore the vast search space and sample correct rationales (Lightman et al., 2024; Zhang et al., 2023; Xie et al., 2023; Xi et al., 2024). Consequently, the models tend to over-sample easy queries and under-sample queries they have yet to master (Tong et al., 2024). On the other hand, as iterations proceed, this imbalance in sampling is exacerbated, leading to a long-tail distribution where solutions to difficult queries almost disappear (Figure 1). This situation is also referred to as *tail narrowing* or *tail cutting* in previous studies (Dohmatob et al., 2024b; Shumailov et al., 2023). As a result, the model's self-improvement is limited, since difficult examples are also crucial for further training (Liu et al., 2024).

To address this imbalance, a common approach is to allocate more sampling trials to under-sampled, challenging queries (Tong et al., 2024), but this can be considerably more costly. In this paper, we propose **G**uided **S**elf-**I**mprovement (**GSI**), an efficient method that leverages interactive guidance signals inspired by the Socratic method (Chang, 2023; Dong et al., 2023b). Specifically, our approach introduces an additional resampling phase, termed *distribution re-balancing* for difficult queries, which is applied after the generation step in the self-improvement process. During this phase, we incorporate targeted guidance signals derived from oracle answers, stepwise rationales, and strong teacher supervision. These signals help to narrow the sampling space, reduce sampling difficulty, and minimize hallucinations during reasoning (Xie et al., 2023; Xi et al., 2024). As a result, GSI enables more effective exploration, increases solution coverage for challenging queries (Bansal et al., 2024), and mitigates the issue of tail narrowing during the sampling process.

We perform experiments across four models and six mathematical reasoning tasks, including arithmetic reasoning, abstract algebra, and formal logic. The results demonstrate that GSI mitigates the performance bottlenecks of self-improvement while maintaining computational efficiency. Further analysis shows that this method leads to a more balanced solution distribution and improved model generalization across multiple reasoning tasks. In addition to natural language reasoning, our method has also been proven effective in program-based reasoning (Chen et al., 2023).

Our contributions are summarized as follows:

- We conduct an in-depth study of the self-improvement process, revealing the performance bottlenecks driven by a long-tail distribution of solutions, which results from increasingly imbalanced data sampling.

- To efficiently mitigate the problem of tail narrowing, we introduce the Guided Self-Improvement (GSI) method that employs Socratic-style guidance signals to assist models in exploring solutions for challenging queries.

- We validate our strategy through comprehensive experiments on four backbone models across six mathematical reasoning tasks, demonstrating the effectiveness and efficiency of GSI.

2 Related Work

2.1 Self-improvement for LLMs

Self-improvement methods, where models refine themselves using self-generated data, have proven effective in enhancing problem-solving abilities without human intervention (Huang et al., 2023; Zelikman et al., 2022). To ensure the reliability of this process, the generated data is typically filtered using external supervision signals. These signals can be binary rewards, such as correctness checks based on reference answers (Yuan et al., 2023; Zelikman et al., 2022; Tong et al., 2024) or compiler execution feedback (Haluptzok et al., 2023). Alternatively, more nuanced approaches involve scoring (Gülçehre et al., 2023) or ranking systems (Dong et al., 2023a), which may be generated by the model itself (Yuan et al., 2024) or external reward models (Hosseini et al., 2024; Qi et al., 2024). Some methods adopt weaker supervision signals, such as majority voting across multiple outputs (Huang et al., 2023).

Once filtered, the high-quality data supports post-training through methods like SFT (Zelikman et al., 2022; Liang et al., 2024) or preference-based techniques like Direct Preference Optimization (DPO, Yuan et al., 2024). This process is often iterative, allowing models to continually generate new data, filter it, and use it to refine their performance further (Zelikman et al., 2022; Gülçehre et al., 2023; Yuan et al., 2024).

2.2 Distribution Shift in Synthetic Data

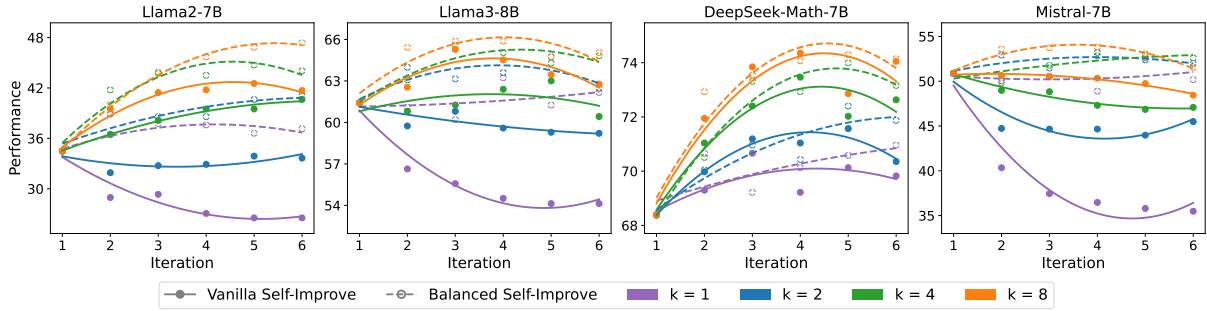The scaling law reveals a predictable increase in model performance as the volume of training data

Figure 2: **Iterative performance in the self-improvement.** Experiments are conducted on GSM8K with varying sampling numbers k. Solid markers show the performance of vanilla self-improve, with the solid line fitting these points. The performance plateaus after a few iterations. Hollow markers represent the performance after supplementing tail data, with a dashed line trend. It balances the distribution and alleviates performance bottlenecks.

grows (Kaplan et al., 2020). With the development of LLMs, the demand for vast amounts of high-quality data has surged, leading researchers to rely increasingly on synthetic data. These methods have proven effective across various tasks, from general-purpose chatbots (Dubey et al., 2024; Adler et al., 2024) to specialized fields such as mathematical reasoning (Yue et al., 2024; Yu et al., 2024).

However, the synthetic data also introduces the risk of *model collapse*, where their performance degrades due to recursive training on model-generated data (Shumailov et al., 2023; Alemohammad et al., 2024). This phenomenon arises from *distribution shift*: as models favor high-probability outputs, their results become increasingly uniform, leading to reduced variance. Over time, this shift manifests in the declining diversity of model responses (Guo et al., 2024b), the disappearance of tail behaviors (Dohmatob et al., 2024b), and the amplification of systematic bias (Yu et al., 2023). Wu et al. (2024) have also observed similar degradation in self-improvement loops, which aligns with the core argument of this paper.

3 Preliminaries

3.1 Formulation of Self-improvement

Given a large language model M_0 and the original training dataset $\mathcal{D} = \{(x_i, r_i, y_i)\}_{i=1}^N$, where x_i is the problem, r_i is the chain-of-thought rationale (Wei et al., 2022b) and y_i represents the final answer. Each rationale r_i consists of several intermediate steps, i.e., $r_i = [r_{i,1}, \ldots, r_{i,L}]$, where L denotes the number of steps. The self-improvement process enhances the model's reasoning ability through iterative refinement over T cycles. Each iteration $t \in [1, T]$ consists of two main steps: *Generate* and *Improve*.

Generate step. At iteration t, the previous model M_{t-1} generates multiple reasoning paths for each problem. Specifically, we allocate k sampling times to each query $x_i \in \mathcal{D}$:

$$(\hat{r}_i, \hat{y}_i) = M_{t-1}(x_i)$$

The newly generated data points form a set $\mathcal{D}' = \{(x_i, \hat{r}_i^j, \hat{y}_i^j) \mid x_i \in \mathcal{D}, j = [1, k]\}$. Each candidate solution $\hat{y}_i^j$ is evaluated by a binary reward function $\mathrm{rf}(y_i, \hat{y}_i) \in \{0, 1\}$, which verifies correctness based on ground-truth answers y_i. Only correct solutions with $\mathrm{rf}(\cdot) = 1$ are filtered to form the high-quality dataset $\mathcal{D}_t$ for the current iteration t.

Improve step. In the t-th iteration, the model M_t is fine-tuned on the self-generated high-quality dataset $\mathcal{D}_t$. The fine-tuning objective is to minimize the negative log-likelihood (NLL) loss:

$$\mathcal{L}_{\text{SFT}} = -\mathbb{E}_{(x,r) \sim \mathcal{D}_t} \sum_{l=1}^{L} \log M(r_l \mid r_{<l}, x).$$

By minimizing $\mathcal{L}_{\text{SFT}}$, the model iteratively improves its ability to generate correct rationales, and this process is repeated for T iterations to achieve self-improvement. Note that in the first iteration, we directly fine-tune M_0 on the original dataset $\mathcal{D}$ to obtain M_1.

3.2 Biased Sampling and Tail Effect

In the self-improvement sampling process, there is a tendency to select higher-quality and more accurate data, which introduces a phenomenon known as sampling bias (Alemohammad et al., 2024; Shumailov et al., 2023). This bias often results in the truncation or narrowing of low-probability "tails" in the data distribution.

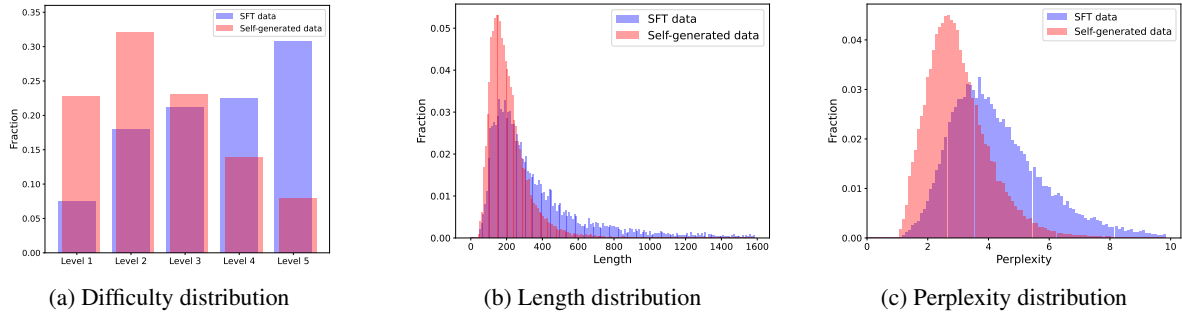|		
(a) Difficulty distribution	(b) Length distribution	(c) Perplexity distribution

Figure 3: **Comparison of data distributions** between the self-generated and original (SFT) datasets. **(a)** Difficulty distribution across five levels in MATH tasks, with level 1 representing the easiest and level 5 the most difficult. The self-generated data has a lower proportion of difficult problems. **(b)** Length distribution indicates that the self-generated data tends to be shorter compared to the original dataset. **(c)** Perplexity diagram of each training sequence measured with the Llama3-8B, shows that the tails in the self-generated data are diminished.

To illustrate the effects of sampling bias, consider a one-dimensional Gaussian distribution $X^0 \sim \mathcal{N}(\mu, \sigma^2)$. Let $\lambda \in [0, 1]$ represent the sampling bias parameter and biased sampling from $\mathcal{N}(\mu, \lambda\sigma^2)$. When $\lambda = 1$, the sampling is unbiased, maintaining the original variance. Conversely, when $\lambda = 0$, the sampling is derived from the modes of the generative distribution M_t with zero variance.

Initially, the tails of the distribution, which represent low-probability data, begin to diminish due to their low likelihood of being sampled. As the iteration continues, the heavy-tailed data is increasingly excluded, causing a shift in the overall distribution. This phenomenon is referred to as tail narrowing or tail cutting, leading to a more peaked distribution (Dohmatob et al., 2024b).

4 Performance Bottleneck and Tail Narrowing in Self-Improvement

Despite extensive research into self-improvement methods (Gülçehre et al., 2023; Singh et al., 2024), the performance dynamics across successive training iterations remain underexplored. To this end, we conduct experiments across four backbone models to investigate the effects of sampling and tail data during iterative training. Tail data refers to samples for which the model rarely generates correct solutions during sampling. These samples lie in the "long tail" of data distributions.

Performance trends. To uncover the relationship between performance trends with sampling, we vary the number of sampling k. Figure 2 reveals the following: (1) Lower sampling times (e.g., $k = 1$ or $k = 2$) degrade model performance, leading to negative gains. Llama2-7B, Mistral-7B, and

Llama3-8B underperform compared to SFT on the original dataset (iteration 1). This decline stems from the models' weaker reasoning abilities, which limit their coverage of challenging queries. Consequently, the models tackle only basic problems, resulting in a degradation or collapse of reasoning ability (Shumailov et al., 2023; Dohmatob et al., 2024a). (2) With larger sampling numbers k, performance improves across multiple iterations. The most notable gains occur during the first iteration, as shown by the green and orange solid lines in Figure 2. However, after the third iteration, progress halts, eventually reaching a performance plateau.

Impact of tail data. As previously discussed, merely scaling the number of sampling eventually encounters performance bottlenecks. However, challenging low-probability samples, often referred to as tail data, are often considered more crucial for improving model performance (Sorscher et al., 2022; Liu et al., 2024; Tong et al., 2024).

To investigate the impact of tail data, we conduct additional experiments targeting these difficult examples. For queries that do not yield a correct response after k sampling attempts, we supplement them with a golden rationale, ensuring that every query has at least one practically correct response and effectively rebalancing the long-tail distribution. We use hollow markers to represent the performance with tail data. As shown in Figure 2, across different numbers of sampling and model variants, the performance represented by the dashed line exceeds that of the solid line representing vanilla self-improve. The results demonstrate that rebalanced data helps mitigate performance degradation under a small number of sampling attempts (e.g., $k = 1$ or $k = 2$). Moreover, as k increases, a

larger number of sampling alleviates performance plateaus and boosts model efficacy to some extent. This study highlights the value of solutions to challenging queries in overcoming the limits of finite and progressively biased sampling.

Emergence of tail narrowing. To analyze the distributional characteristics of tail data, we analyze three dimensions: difficulty, response length, and perplexity, revealing the differences between self-generated and original data. Figure 3a shows that model-generated data tends to be at a low level and simpler compared to the original MATH dataset, which contains more challenging queries. Consistently, DART-Math (Tong et al., 2024) also identifies a synthesis bias towards easy queries.

Using response length as a complexity indicator, as suggested by Fu et al. (2023), we find that self-generated responses are generally shorter, peaking around 200 tokens (Figure 3b). This indicates a bias towards simplicity and a lower proportion of complex data.

From a semantic perspective, we computed the perplexity based on Llama3-8B. As shown in Figure 3c, the self-generated data has lower perplexity, indicating a shift toward more probable and coherent tokens. This reduces the occurrence of rare, complex and diverse tokens in the tail.

These observations indicate a diminishing trend of tail data, termed *Tail Narrowing* (Dohmatob et al., 2024b). In each iteration, the distribution progressively narrows, hindering performance to challenging queries.

5 Guided Self-Improvement

To mitigate the observed tail-narrowing phenomenon, a straightforward solution is to increase the sampling trials for tail data. However, directly tackling these difficult queries from scratch often results in low success rates and higher costs due to repeated failed attempts. Drawing from Socratic-style education (Chang, 2023; Dong et al., 2023b), we introduce guidance-based exploration techniques to improve sampling efficiency, such as learning from demonstrations (Schaal, 1996; Subramanian et al., 2016). We provide the model with tailored assistance in structured contexts, enabling it to address difficult queries more effectively. The following paragraphs outline four guiding strategies we propose.

Answer-driven. This strategy incorporates the ground-truth answer y_i along with the input query x_i as context to guide the generation process. It helps the model better align with the expected solution, particularly for challenging queries (Zelikman et al., 2022). Formally, at each iteration t, instead of inputting only x_i into the model M_{t-1}, we extend the prompt by appending y_i as a hint:

$$(\hat{r}_i, \hat{y}_i) = M_{t-1}(x_i, \text{hint}(y_i)).$$

This approach helps the model focus on the reasoning process behind the answer, reducing the overall difficulty of the task.

Rationale-driven. In this strategy, we further extend the input by introducing a rationale r_i, which helps the model derive the correct reasoning process. Unlike the answer-driven method, providing a rationale offers a more detailed reference for the model to follow, narrowing the exploration space of reasoning paths (Yang et al., 2024). Formally, at iteration t, the input to the model M_{t-1} is augmented as follows:

$$(\hat{r}_i, \hat{y}_i) = M_{t-1}(x_i, \text{hint}(r_i)).$$

This approach enables the model to handle queries it has yet to master and ensures a higher coverage of solved problems. Importantly, it alleviates the hallucinations when the model tries to provide reasoning paths for problems it doesn't fully understand (Lanham et al., 2023), improving the reliability of generated data.

Interactive sampling. Inspired by previous work in the area of Interactive RL (Subramanian et al., 2016; Suay and Chernova, 2011), we introduce feedback from a stronger model M_s after the model M_{t-1} fails. Instead of providing hints along with the query, this dynamic process ensures that the model can explore its own solution before receiving external guidance. Formally, after the model M_{t-1} generates an incorrect answer, we re-sample by giving it both its prior incorrect output and the feedback f_i from M_s as additional context:

$$\begin{aligned} (\hat{r}_i^{\text{error}}, \hat{y}_i^{\text{error}}) &= M_{t-1}(x_i), \\ f_i &= M_s(x_i, r_i, y_i, \hat{r}_i^{\text{error}}), \\ (\hat{r}_i, \hat{y}_i) &= M_{t-1}(x_i, \hat{r}_i^{\text{error}}, f_i). \end{aligned}$$

This feedback includes an analysis and correction of the model's errors, reducing its reliance on the correct answer. Through this interactive process,

we balance exploration and correction, enabling the model to learn from its mistakes without overly restricting its reasoning path.

State reset. Drawing inspiration from the concept of *state reset*, i.e., going back to intermediate states during problem-solving to refine the approach (Chang et al., 2024; Xi et al., 2024), we adopt a strategy where the model is guided step by step with partial rationales. Instead of supplying the full rationale r_i immediately, after l incorrect attempts, the model is provided with the preceding reasoning steps $r_{<l}$ from $r_i = [r_{i,1}, \ldots, r_{i,L}]$, gradually narrowing down the exploration space:

$$(\hat{r}_i, \hat{y}_i) = M_{t-1}(x_i, \text{hint}(r_{i,<l})).$$

This method reduces the difficulty of queries at a fine-grained level, relieving the model of cognitive overload while still allowing flexibility in the model's solution. Although it increases the number of attempts, the incremental hint helps identify the threshold where the model can solve problems independently with minimal guidance.

6 Experiments

6.1 Experimental Setups

Models. We conduct experiments using four widely adopted foundation models, including Llama2-7B-Base (Touvron et al., 2023), Llama3-8B-Base (Dubey et al., 2024), Deepseek-Math-7B-Base (Shao et al., 2024), and Mistral-7B-v0.3 (Jiang et al., 2023). For the stronger model in the interactive sampling process, we employ Llama3-70B-Instruct (Dubey et al., 2024).

Datasets. We utilize six math reasoning datasets. These include arithmetic reasoning datasets such as GSM8K (Cobbe et al., 2021), AQuA (Ling et al., 2017), MathQA (Amini et al., 2019) and SVAMP (Patel et al., 2021), as well as a more challenging dataset MATH (Hendrycks et al., 2021). We also include TheoremQA for abstract algebra and formal logic. To evaluate generalization, we choose AQuA, GSM8K, MATH as held-in datasets and MathQA, SVAMP, TheoremQA as held-out datasets. To ensure consistency in answer format across different datasets, we utilize the unified data provided by the MathInstruct dataset (Yue et al., 2024) and follow its train-test splits. More dataset statistics can be found in Appendix A.

Implementation details. Following Huang et al. (2023) and Singh et al. (2024), we fine-tune the pre-trained model M_0 during the Improve Step of each iteration to prevent overfitting. We set the iteration number $T = 4$ and sampling number $k = 8$. To mitigate the tail-narrowing effect, we identify queries with less than a 50% probability of yielding correct completions as heavy-tailed data. For the tail data, we apply the GSI strategy, resampling up to k times until the query no longer falls into the tail data. All experiments are performed on 8 A100-80GB GPUs. We run the SFT and Improve Step for 1 epoch. The learning rate is set to 1×10^{-5}. For sampling and evaluation, we leverage the vLLM (Kwon et al., 2023) framework, setting a maximum of 1024 output tokens. The temperature is set to 0.7 during sampling and 0 during evaluation. The prompt templates are detailed in Appendix D.

Baselines. To evaluate the impact of GSI on distributional adjustments, we compare it against SFT and Self-improve variants. We also include baselines with significantly higher sampling trials than GSI.

- **SFT**: Fine-tuning on the original dataset for 1 epoch, which corresponds to the first iteration of self-improvement.
- **Self-Improve** ($k = 8$): In each iteration, we sample $k = 8$ completions per query from the original dataset, filter out correct reasoning paths, and then improve on the self-generated data.
- **Brute-Force Self-Improve** ($k = 64$ or $k = 128$): Based on Vanilla Self-Improve ($k = 8$), we add a distribution re-balancing stage for tail data in each iteration. It performs non-guided brute-force sampling of tail-end data up to k times without additional guidance.

6.2 Main Results

The main results are shown in Table 1. We have the following key findings:

Re-balancing tail data improves coverage and performance of self-improvement. Compared to SFT, vanilla self-improve with $k = 8$ boosts reasoning on held-out datasets but shows only marginal gains, with performance bottlenecks and degradation on held-in datasets. This aligns with our observation (§ 4), where we identify tail narrowing as the primary cause. To mitigate this, Brute-Force Self-Improve ($k = 64$ or $k = 128$) performs additional sampling on tail data, which

Models	Methods	Sample Budget	Coverage	Held-in Datasets				Held-out Datasets			
				Avg.	AQuA	GSM8K	MATH	Avg.	MathQA	SVAMP	Thm.QA
Llama2-7B	SFT	-	-	21.00	27.17	31.31	4.52	21.08	22.08	36.80	4.38
	SI ($k=8$)	0.24M	52.6%	21.53	25.59	33.13	5.86	24.46	27.37	39.50	6.50
	Brute-Force SI ($k=64$)	1.11M	77.1%	23.73	29.53	35.86	5.80	23.42	25.76	39.50	5.00
	Brute-Force SI ($k=128$)	1.67M	80.7%	23.84	27.17	<u>37.98</u>	6.36	24.22	26.57	40.60	5.50
	Guided Self-Improve ($k=8$)										
	+ Answer-driven	0.37M	99.0%	23.44	28.34	35.94	6.04	25.27	25.83	**43.60**	6.38
	+ Rationale-driven	0.34M	99.9%	24.32	30.32	36.16	**6.48**	25.68	<u>28.11</u>	41.30	**7.63**
	+ Interactive Sampling	0.36M	96.3%	<u>25.00</u>	<u>30.71</u>	37.83	<u>6.46</u>	26.25	27.84	<u>43.40</u>	<u>7.50</u>
	+ State Reset	0.38M	82.0%	**25.91**	**31.10**	**40.18**	6.44	**26.79**	**29.45**	43.30	**7.63**
Llama3-8B	SFT	-	-	37.27	39.37	57.47	14.96	38.68	44.29	63.00	8.75
	SI ($k=8$)	0.24M	68.6%	36.87	39.76	59.14	11.70	39.09	45.63	62.40	9.25
	Brute-Force SI ($k=64$)	0.85M	86.4%	38.16	38.98	61.11	14.40	38.30	45.86	60.90	8.13
	Brute-Force SI ($k=128$)	1.26M	88.7%	37.55	41.34	<u>61.64</u>	9.68	39.32	45.90	62.80	9.25
	Guided Self-Improve ($k=8$)										
	+ Answer-driven	0.31M	98.5%	38.71	42.52	59.82	13.80	40.15	46.85	63.60	<u>10.00</u>
	+ Rationale-driven	0.29M	99.8%	39.14	<u>42.91</u>	60.27	14.24	41.08	<u>47.94</u>	**65.30**	10.00
	+ Interactive Sampling	0.31M	97.3%	<u>39.34</u>	42.52	60.05	<u>15.46</u>	<u>41.12</u>	47.24	64.50	**11.63**
	+ State Reset	0.32M	90.2%	**41.64**	**46.46**	**62.62**	**15.54**	**41.33**	**49.25**	<u>65.00</u>	9.75
DeepSeek-Math-7B	SFT	-	-	50.01	60.63	60.73	28.66	21.15	21.61	37.10	4.75
	SI ($k=8$)	0.24M	79.8%	52.22	56.69	68.92	31.06	49.63	64.26	67.50	17.13
	Brute-Force SI ($k=64$)	0.64M	91.5%	53.95	57.87	70.89	**33.08**	50.92	<u>65.36</u>	70.40	17.00
	Brute-Force SI ($k=128$)	0.93M	92.2%	52.43	59.45	72.02	25.82	48.76	64.69	68.20	13.38
	Guided Self-Improve ($k=8$)										
	+ Answer-driven	0.30M	99.5%	53.83	<u>61.02</u>	70.05	30.42	51.80	64.32	**72.70**	**18.38**
	+ Rationale-driven	0.29M	99.7%	53.71	58.66	71.65	30.82	51.20	64.02	72.20	17.38
	+ Interactive Sampling	0.30M	96.9%	**55.67**	**61.81**	<u>72.63</u>	32.56	<u>51.69</u>	**66.83**	71.10	17.13
	+ State Reset	0.31M	93.6%	<u>55.04</u>	59.45	**72.78**	32.90	**51.85**	64.59	**72.70**	<u>18.25</u>
Mistral-7B	SFT	-	-	28.27	31.10	44.96	8.74	21.08	22.08	36.80	4.38
	SI ($k=8$)	0.24M	61.9%	25.22	27.95	40.56	7.14	26.96	33.84	43.30	3.75
	Brute-Force SI ($k=64$)	0.87M	82.5%	28.23	28.74	<u>47.16</u>	8.80	30.16	34.94	49.90	5.36
	Brute-Force SI ($k=128$)	1.27M	85.1%	28.32	30.32	45.79	8.84	28.81	33.03	49.40	4.00
	Guided Self-Improve ($k=8$)										
	+ Answer-driven	0.33M	98.3%	28.09	**34.65**	42.00	7.62	29.70	34.34	49.90	4.88
	+ Rationale-driven	0.31M	99.7%	<u>29.13</u>	32.68	46.17	8.54	**32.14**	**35.24**	**53.80**	**7.38**
	+ Interactive Sampling	0.31M	96.4%	29.04	32.28	45.87	<u>8.98</u>	<u>30.22</u>	<u>35.04</u>	50.00	<u>5.63</u>
	+ State Reset	0.34M	86.7%	**31.23**	<u>33.07</u>	**50.95**	**9.68**	30.21	34.94	<u>50.20</u>	5.50

Table 1: Main results on six math reasoning tasks. The best result for each dataset is highlighted in **bold**, while the second-best result is marked with <u>underline</u>. Results marked in blue indicate average scores. Thm.QA denotes the TheoremQA task. *Coverage* refers to the number of unique problems solved in the Generate Step, while *Sample Budget* indicates the total number of sampling times during this step. "SI" refers to Self-Improve. The baselines include SFT, the vanilla Self-Improve, and Brute-Force Self-Improve.

re-balances the distribution. This adjustment significantly enhances both coverage and overall performance, with further improvements as the number of samples increases. For example, in Llama2-7B, coverage increases from 52.6% to 80.7%, with performance improving from 21.53 to 23.84. In Mistral-7B, the rebalancing also reverses the observed performance decline. Thus, incorporating a resampling stage for challenging, heavy-tailed data proves essential.

GSI outperforms brute-force sampling with better efficiency. To optimize the amount of sampling computation required for self-improvement, we analyze the sampling efficiency during the Generate step phase. For each query $x_i \in \mathcal{D}$ at iteration t, we perform $k_{i,t}$ sampling operations. The cumulative sample budget across all queries and T iterations is defined as: Sample Budget = $\sum_{x_i \in \mathcal{D}} \sum_{t=1}^{T} k_{i,t}$.

When scaling resampling operations from 64 to 128, we observe that the improvement in sampling coverage becomes slower, and the performance gains diminish despite the additional computational costs. This suggests that the model may get trapped in vast search space, particularly when dealing with more challenging queries. In contrast, our strategy GSI, which leverages Socratic-style guidance, achieves more compute-efficient sampling. It outperforms brute-force sampling while using only one-third of the sampling budget. Specifically, on Llama3-8B, the state reset strategy performs 0.32M
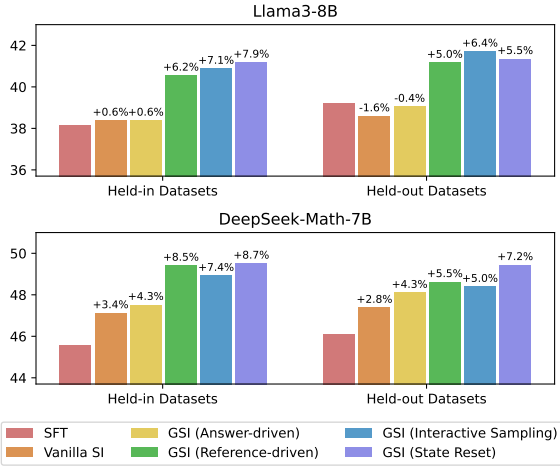
Figure 4: Comparison of average performance on six math tasks using PoT. The percentage of improvement is significant in the held-in datasets.

sampling, which is only one-third of the budget required for brute-force sampling. Moreover, the model shows improved held-in performance from 37.55 to 41.64 and generalizes well to held-out datasets.

The effectiveness of different strategies. Among the four strategies, the state reset strategy, which samples from different initial states and generates diverse reasoning paths, performs relatively better. However, the effectiveness of different strategies depends on the model's inherent capabilities. For example, the answer-driven strategy, which provides only the correct value and requires the model to reason backward to generate a rationale, demands advanced reasoning abilities (Zelikman et al., 2022). Therefore, this approach yields modest performance gains on weaker models such as Llama2-7B. Further investigation is needed to explore how different models can be optimally paired with various strategies.

7 Discussion

7.1 Effectiveness on PoT Reasoning

To fully exploit the potential of diverse reasoning processes, we extend our investigation to Program-of-Thought (PoT, Chen et al., 2023) prompting. In the self-improvement process, we utilize PoT rationales for training, then filter data and evaluate performance based on compiler-executed results. As shown in Figure 4, four strategies consistently outperform the self-improvement baseline in program-based reasoning. The state reset strategy on the DeepSeek-Math-7B model shows notable

Models	Methods	Held-in	Held-out
DeepSeek-Coder-1.3B	SFT	10.66	14.75
	Vanilla Self-Improve	12.61	13.98
	GSI (Answer-driven)	13.20	15.62
	GSI (Rationale-driven)	<u>14.99</u>	16.37
	GSI (Interactive)	14.13	<u>15.94</u>
	GSI (State Reset)	**16.04**	**17.38**
CodeLlama-13B	SFT	23.43	28.15
	Vanilla Self-Improve	27.85	29.99
	GSI (Answer-driven)	31.11	31.57
	GSI (Rationale-driven)	31.38	30.13
	GSI (Interactive)	<u>32.27</u>	**33.77**
	GSI (State Reset)	**33.04**	<u>31.68</u>

Table 2: Effectiveness of GSI on different sizes of models. The improvement becomes more pronounced as the model size increases.

relative gains, with an improvement of up to 8.7%. Similarly, the reference-driven strategy leads to a performance boost of 8.5%.

7.2 Performance of Different Model Sizes

To further explore, we investigate the effectiveness of the proposed strategy across different model sizes. We choose a smaller model, DeepSeek-Coder-1.3B (Guo et al., 2024a), and a larger model, CodeLlama-13B (Rozière et al., 2023), in our experiments. As shown in Table 2, DeepSeek-Coder-1.3B exhibits only marginal improvements when applying the answer-driven strategy compared to the others. The limited improvement may be attributed to the nature of the strategy, which requires the model to reverse-engineer a solution from a given true answer (Zelikman et al., 2022). While the final result is provided, deriving a good justification can be challenging for smaller models (Wei et al., 2022a). However, when scaling up to the 13B model, we observe a pronounced performance boost. The results suggest that GSI is more effective with larger models, which are equipped with advanced reasoning abilities.

7.3 Short-cutting in Generated Rationales

This experiment explores how different guiding strategies influence the quality of rationales. Inspired by Zelikman et al. (2022), we focus on the number of rationale steps. Figure 5 shows that, in most cases, the model's reasoning steps align with the original annotated steps. However, under the rationale-driven strategy, the model is more likely to generate fewer steps than others. Further analysis reveals that providing rationales can cause the model to skip reasoning steps, a phenomenon we refer to as "Hint Short-cutting" (Zelikman et al.,
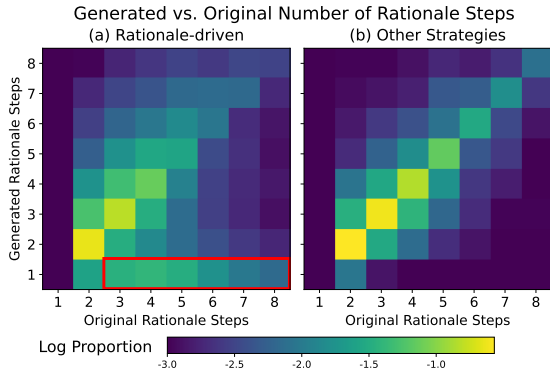
Figure 5: Comparison of the number of rationale steps generated by the model relative to the number of steps used in the ground truth. The red box highlights the occurrence of skip steps in the rationale-driven strategy.

2022). This tendency may weaken the model's ability to think step-by-step, potentially hindering iterative training (Dohmatob et al., 2024a). We show an example in Appendix B, where the model skips critical steps.

7.4 Impact of the Sampling Hyperparameter

We investigate the impact of the hyperparameter k, which determines the number of sampling times allocated to each query. As shown in Table 3, increasing k yields substantial performance improvements. Notably, the most significant improvements occur up to $k = 8$, after which the gains begin to plateau. Therefore, we choose $k = 8$ as a practical configuration, achieving a balanced trade-off between computational cost and performance gains.

8 Conclusion

In this work, we delve into the performance bottlenecks in the self-improvement process of LLMs, identifying the issue of tail narrowing caused by progressively imbalanced data sampling. To mitigate this, we propose Guided Self-Improvement (GSI), a new method that incorporates a distribution re-balancing phase and Socratic-style guidance to enhance solution coverage for challenging queries. Experimental results across multiple models and mathematical reasoning tasks demonstrate the effectiveness of this method in improving reasoning performance while maintaining computational efficiency. We believe GSI offers a promising direction for enhancing the scalability and generalization of self-improving models in the future.

Models	Setting	AQuA	GSM8K	MATH
Llama3-8B	$k = 2$	40.95	56.94	13.20
	$k = 4$	42.91	61.49	15.28
	$k = 8$	46.46	**62.62**	**15.84**
	$k = 16$	**47.24**	62.32	15.64
Mistral-7B	$k = 2$	29.92	40.10	6.82
	$k = 4$	31.49	47.46	8.76
	$k = 8$	33.07	**50.95**	9.68
	$k = 16$	**33.46**	50.19	**9.82**

Table 3: Performance of GSI (State Reset) with varying values of k.

Limitations

While our work introduces a new approach to mitigating the tail narrowing through GSI, there are still several limitations. First, for computational efficiency, we do not scale the sampling in each iteration. However, we conduct a series of experiments (§ 4) to provide insights into how scaling the number of sampling can boost performance. Second, following prior self-improvement works, we use binary signals for supervision based on final answer checks. However, poor and spurious rationales while yielding correct answers may be utilized, which could hinder the improvement of reasoning ability. Filtering low-quality reasoning paths and ensuring the quality of self-generated data remains an area for further investigation.

Acknowledgment

References

Bo Adler, Niket Agarwal, Ashwath Aithal, Dong H. Anh, Pallab Bhattacharya, Annika Brundyn, Jared Casper, Bryan Catanzaro, Sharon Clay, Jonathan M. Cohen, Sirshak Das, Ayush Dattagupta, Olivier Delalleau, Leon Derczynski, Yi Dong, Daniel Egert, Ellie Evans, Aleksander Ficek, Denys Fridman, Shaona Ghosh, Boris Ginsburg, Igor Gitman, Tomasz Grzegorzek, Robert Hero, Jining Huang, Vibhu Jawa,

Joseph Jennings, Aastha Jhunjhunwala, John Kamalu, Sadaf Khan, Oleksii Kuchaiev, Patrick LeGresley, Hui Li, Jiwei Liu, Zihan Liu, Eileen Long, Ameya Sunil Mahabaleshwarkar, Somshubra Majumdar, James Maki, Miguel Martinez, Maer Rodrigues de Melo, Ivan Moshkov, Deepak Narayanan, Sean Narenthiran, Jesus Navarro, Phong Nguyen, Osvald Nitski, Vahid Noroozi, Guruprasad Nutheti, Christopher Parisien, Jupinder Parmar, Mostofa Patwary, Krzysztof Pawelec, Wei Ping, Shrimai Prabhumoye, Rajarshi Roy, Trisha Saar, Vasanth Rao Naik Sabavat, Sanjeev Satheesh, Jane Polak Scowcroft, Jason Sewall, Pavel Shamis, Gerald Shen, Mohammad Shoeybi, Dave Sizer, Misha Smelyanskiy, Felipe Soares, Makesh Narsimhan Sreedhar, Dan Su, Sandeep Subramanian, Shengyang Sun, Shubham Toshniwal, Hao Wang, Zhilin Wang, Jiaxuan You, Jiaqi Zeng, Jimmy Zhang, Jing Zhang, Vivienne Zhang, Yian Zhang, and Chen Zhu. 2024. Nemotron-4 340b technical report. *CoRR*, abs/2406.11704.

Sina Alemohammad, Josue Casco-Rodriguez, Lorenzo Luzi, Ahmed Imtiaz Humayun, Hossein Babaei, Daniel LeJeune, Ali Siahkoohi, and Richard G. Baraniuk. 2024. Self-consuming generative models go MAD. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2357–2367. Association for Computational Linguistics.

Hritik Bansal, Arian Hosseini, Rishabh Agarwal, Vinh Q. Tran, and Mehran Kazemi. 2024. Smaller, weaker, yet better: Training LLM reasoners via compute-optimal sampling. *CoRR*, abs/2408.16737.

Edward Y. Chang. 2023. Prompting large language models with the socratic method. In *13th IEEE Annual Computing and Communication Workshop and Conference, CCWC 2023, Las Vegas, NV, USA, March 8-11, 2023*, pages 351–360. IEEE.

Jonathan D. Chang, Wenhao Zhan, Owen Oertell, Kianté Brantley, Dipendra Misra, Jason D. Lee, and Wen Sun. 2024. Dataset reset policy optimization for RLHF. *CoRR*, abs/2404.08495.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. 2023. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Trans. Mach. Learn. Res.*, 2023.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias

Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

Elvis Dohmatob, Yunzhen Feng, Arjun Subramonian, and Julia Kempe. 2024a. Strong model collapse. *Preprint*, arXiv:2410.04840.

Elvis Dohmatob, Yunzhen Feng, Pu Yang, François Charton, and Julia Kempe. 2024b. 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.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. 2023a. RAFT: reward ranked finetuning for generative foundation model alignment. *Trans. Mach. Learn. Res.*, 2023.

Qingxiu Dong, Li Dong, Ke Xu, Guangyan Zhou, Yaru Hao, Zhifang Sui, and Furu Wei. 2023b. Large language model for science: A study on P vs. NP. *CoRR*, abs/2309.05689.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2023. Complexity-based prompting for multi-step reasoning. In *The Eleventh International*

Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023. OpenReview.net.

Çağlar Gülçehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. 2023. Reinforced self-training (rest) for language modeling. *CoRR*, abs/2308.08998.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y. Wu, Y. K. Li, Fuli Luo, Yingfei Xiong, and Wenfeng Liang. 2024a. Deepseek-coder: When the large language model meets programming - the rise of code intelligence. *CoRR*, abs/2401.14196.

Yanzhu Guo, Guokan Shang, Michalis Vazirgiannis, and Chloé Clavel. 2024b. The curious decline of linguistic diversity: Training language models on synthetic text. In *Findings of the Association for Computational Linguistics: NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 3589–3604. Association for Computational Linguistics.

Patrick Haluptzok, Matthew Bowers, and Adam Tauman Kalai. 2023. Language models can teach themselves to program better. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Arian Hosseini, Xingdi Yuan, Nikolay Malkin, Aaron C. Courville, Alessandro Sordoni, and Rishabh Agarwal. 2024. V-star: Training verifiers for self-taught reasoners. *CoRR*, abs/2402.06457.

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.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *CoRR*, abs/2310.06825.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *CoRR*, abs/2001.08361.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. 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*.

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.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamile Lukosiute, Karina Nguyen, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Robin Larson, Sam McCandlish, Sandipan Kundu, Saurav Kadavath, Shannon Yang, Thomas Henighan, Timothy Maxwell, Timothy Telleen-Lawton, Tristan Hume, Zac Hatfield-Dodds, Jared Kaplan, Jan Brauner, Samuel R. Bowman, and Ethan Perez. 2023. Measuring faithfulness in chain-of-thought reasoning. *CoRR*, abs/2307.13702.

Yiming Liang, Ge Zhang, Xingwei Qu, Tianyu Zheng, Jiawei Guo, Xinrun Du, Zhenzhu Yang, Jiaheng Liu, Chenghua Lin, Lei Ma, Wenhao Huang, and Jiajun Zhang. 2024. I-SHEEP: self-alignment of LLM from scratch through an iterative self-enhancement paradigm. *CoRR*, abs/2408.08072.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 158–167. Association for Computational Linguistics.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. 2024. What makes good data for alignment? A comprehensive study of automatic data selection in instruction tuning. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021, pages 2080–2094. Association for Computational Linguistics.

Zhenting Qi, Mingyuan Ma, Jiahang Xu, Li Lyna Zhang, Fan Yang, and Mao Yang. 2024. Mutual reasoning makes smaller llms stronger problem-solvers. *CoRR*, abs/2408.06195.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton-Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2023. Code llama: Open foundation models for code. *CoRR*, abs/2308.12950.

Stefan Schaal. 1996. Learning from demonstration. In *Advances in Neural Information Processing Systems 9, NIPS, Denver, CO, USA, December 2-5, 1996*, pages 1040–1046. MIT Press.

Amrith Setlur, Saurabh Garg, Xinyang Geng, Naman Garg, Virginia Smith, and Aviral Kumar. 2024. RL on incorrect synthetic data scales the efficiency of LLM math reasoning by eight-fold. *CoRR*, abs/2406.14532.

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.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Yarin Gal, Nicolas Papernot, and Ross J. Anderson. 2023. The curse of recursion: Training on generated data makes models forget. *CoRR*, abs/2305.17493.

Avi Singh, John D. Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J. Liu, James Harrison, Jaehoon Lee, Kelvin Xu, Aaron T. Parisi, Abhishek Kumar, Alexander A. Alemi, Alex Rizkowsky, Azade Nova, Ben Adlam, Bernd Bohnet, Gamaleldin Fathy Elsayed, Hanie Sedghi, Igor Mordatch, Isabelle Simpson, Izzeddin Gur, Jasper Snoek, Jeffrey Pennington, Jiri Hron, Kathleen Kenealy, Kevin Swersky, Kshiteej Mahajan, Laura Culp, Lechao Xiao, Maxwell L. Bileschi, Noah Constant, Roman Novak, Rosanne Liu, Tris Warkentin, Yundi Qian, Yamini Bansal, Ethan Dyer, Behnam Neyshabur, Jascha Sohl-Dickstein, and Noah Fiedel. 2024. Beyond human data: Scaling self-training for problem-solving with language models. *Trans. Mach. Learn. Res.*, 2024.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. 2022. Beyond neural scaling laws: beating power law scaling via data pruning. 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*.

Halit Bener Suay and Sonia Chernova. 2011. Effect of human guidance and state space size on interactive reinforcement learning. In *2011 Ro-Man*, pages 1–6. IEEE.

Kaushik Subramanian, Charles Lee Isbell Jr., and Andrea Lockerd Thomaz. 2016. Exploration from demonstration for interactive reinforcement learning. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems, Singapore, May 9-13, 2016*, pages 447–456. ACM.

Yuxuan Tong, Xiwen Zhang, Rui Wang, Ruidong Wu, and Junxian He. 2024. Dart-math: Difficulty-aware rejection tuning for mathematical problem-solving. *CoRR*, abs/2407.13690.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022a. Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022b. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Ting Wu, Xuefeng Li, and Pengfei Liu. 2024. Progress or regress? self-improvement reversal in post-training. *CoRR*, abs/2407.05013.

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

10638

Fan, Yuhao Zhou, Shihan Dou, Xiao Wang, Xinbo Zhang, Peng Sun, Tao Gui, Qi Zhang, and Xuanjing Huang. 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.

Yuxi Xie, Kenji Kawaguchi, Yiran Zhao, James Xu Zhao, Min-Yen Kan, Junxian He, and Michael Qizhe Xie. 2023. Self-evaluation guided beam search for reasoning. 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*.

Zhaorui Yang, Tianyu Pang, Haozhe Feng, Han Wang, Wei Chen, Minfeng Zhu, and Qian Liu. 2024. Self-distillation bridges distribution gap in language model fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 1028–1043. Association for Computational Linguistics.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Yue Yu, Yuchen Zhuang, Jieyu Zhang, Yu Meng, Alexander J. Ratner, Ranjay Krishna, Jiaming Shen, and Chao Zhang. 2023. Large language model as attributed training data generator: A tale of diversity and bias. 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*.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Chuanqi Tan, and Chang Zhou. 2023. Scaling relationship on learning mathematical reasoning with large language models. *CoRR*, abs/2308.01825.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2024. Mammoth: Building math generalist models through hybrid instruction tuning. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. 2022. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, Longyue Wang, Anh Tuan Luu, Wei Bi, Freda Shi, and Shuming Shi. 2023. Siren's song in the AI ocean: A survey on hallucination in large language models. *CoRR*, abs/2309.01219.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A survey of large language models. *CoRR*, abs/2303.18223.

Appendix

A Dataset Details

	AQuA	GSM8K	MATH	MathQA	SVAMP	Thm.QA
# CoT Train	3000	3000	4000	-	-	-
# PoT Train	1961	3000	4000	-	-	-
# Test	254	1319	5000	2985	1000	800

Table 4: Dataset statistics of the train and test set.

For the training set, we randomly select a subset from the extensive datasets provided by MathInstruct (Yue et al., 2024), including AQuA, GSM8K, and MATH. The test set is consistent with MathInstruct. The specific data quantities are shown in Table 4. We comply with the license for the use of these datasets in our work.

B Error Patterns

Here, we analyze two classic types of errors in model-generated rationales.

Hint Short-cutting. This occurs when the model is provided with the final answer or rationales as a hint during generation, causing it to skip steps (Zelikman et al., 2022). This leads to responses like those shown in Figure 6 and Figure 7.

Spurious Correctness. This refers to cases where, when given the final answer as a hint, the model fails to generate the correct intermediate reasoning due to its limited capabilities but still copies the correct final answer and misleads the reward function. Responses classified as positive may contain incorrect or irrelevant rationales, and training on such data often incentivizes the model to overfit to spurious correlations, resulting in flat

or even inverse scaling with more data during fine-tuning (Setlur et al., 2024). Figure 8 shows a case of spurious correct response.

C Case Study

Here, we present two successful cases: GSI (interactive sampling) in Figure 9 and GSI (state reset) in Figure 10, demonstrating the effectiveness of our proposed method.

D Prompt Details

The prompt template of the vanilla self-improve method is presented in Figure 11. The prompt templates of proposed Guided Self-Improvement (GSI) are detailed in Figure 12, 13, 14, 15, and 16.

Question: Kathryn moved to a new city for the new job she had landed two weeks ago. Her rent was $1200, 1/2 of what she spent on food and travel expenses in a month. Luckily, she found a new friend Shelby, who moved in with her to share the rent. If her salary was $5000 per month, how much money remained after her expenses?

Original Rationales

```
If the rent is half what she spends on
food and travel expenses, she spends
1200 * 2 = $2400 on food and travel
costs. When her friend moved in and
shared the rent costs with her, she
started paying 1/2 * 1200 = $600 in
rent costs. Her total monthly expenses
is now 600 + 2400 = $3000.

When she deducts her monthly expenses
from her salary, she has $5000 - $3000
= $2000 remaining.

The answer is 2000.
```

Generated Rationales

```
Kathryn's monthly expenses is $3000 and
her salary is $5000, so she has $5000 -
$3000 = $2000 remaining.

The answer is 2000.
```

Figure 6: An example of **skipped steps** on the GSM8K dataset. The key steps within the original rationales are highlighted in **purple**, while the corresponding steps generated by Llama2 are indicated in **blue**. In this case, the model, guided by rationale-driven hints, skips critical steps and directly arrives at a premature conclusion.

Question: Two white socks cost 25 cents more than a single brown sock. If two white socks cost 45 cents, how much would you pay for 15 brown socks?

Original Rationales

```
Two white socks cost 25 cents more
than a single brown sock, meaning a
brown sock is sold at $0.45 - $0.25 =
$0.20.

The cost of buying 15 brown socks is
15 * $0.20 = $3.

The answer is 3.
```

Generated Rationales

```
The answer is 3.
```

Figure 7: An example of **skipped steps** on the GSM8K dataset. The key steps within the original rationales are highlighted in **purple**. In this case, the model ignores all the steps and copies the final answer. Such cases arise mainly with answer-driven or rationale-driven hints.

Question: Andy started out the year weighing 156 pounds. He then grew 3 inches and gained 36 pounds. Andy wasn't happy with his weight and decided to exercise. Over the next 3 months, he lost an eighth of his weight every month. How much less does Andy weigh now than at the beginning of the year?

Original Rationales

```
Andy weighed 156 + 36 = 192 pounds
after growing taller. An eighth of his
weight is 1/8 x 192 = 24 pounds. Over
the next 3 months, Andy lost 24 x 3 =
72 pounds. He now weighs 192 - 72 =
120 pounds.

Andy weighs 156 - 120 = 36 pounds less
than at the beginning of the year.

The answer is 36.
```

Generated Rationales

```
Andy grew 3 inches and gained 36 pounds,
so he weighed 156+36 = 192 pounds. He
lost 36/12 = 3 pounds each month. So in
3 months, he lost 3*3 = 9 pounds. So he
weighs 192-9 = 183 pounds now.

The answer is 36.
```

Figure 8: An example of a **hallucinated reasoning process** on the GSM8K dataset. The key steps within the original rationales are highlighted in **purple**, while the corresponding steps generated by Llama2 are indicated in **blue**. In this case, the model fails to generate the correct intermediate reasoning but still copies the correct final answer. This behavior can mislead the reward function and introduce noise into the training data in subsequent iterations. Such cases arise mainly with answer-driven hints.

Question: Average monthly income of a family of 3 earning members was Rs. 735. One of the earning members died and therefore, the average income came down to Rs 650. The income of the deceased was?

Choices: (A) Rs. 692.80 (B) Rs. 905 (C) Rs. 990 (D) Rs. 1385 (E) None

Original Rationales

```
Let's think about the multi-choice
question.

Income of the deceased
= Total income of 3 members - total
income of remaining 2 members.
= 735 x 3 - 650 x 2 Rs.
= 905 Rs.

The answer is B.
```

Generated Rationales

```
Let's solve the multi-choice question
step by step.

Income of deceased
= 735 x 3 - 650 x 3
= 2205 - 1950
= Rs. 255

None of the answer choices match our
calculation, so the correct answer is
indeed (E) None.

The answer is E.
```

Feedback from Stronger Model

Feedback: The incorrect step in your original solution is: "Income of deceased = 735 x 3 - 650 x 3". The mistake lies in the fact that you subtracted the total income of 3 members (after the death) from the total income of 3 members (before the death). You should have subtracted the total income of 2 members (after the death) from the total income of 3 members (before the death). In other words, you incorrectly used 650 x 3 instead of 650 x 2 in your calculation.

Corrected Rationales

```
Let's solve the multi-choice question step by step.

Income of deceased = Total income of 3 members - total income of remaining 2
member = 735 x 3 - 650 x 2 = 2205 - 1300 = Rs. 905

The answer is B.
```

Figure 9: An example of **interactive sampling** on the AQuA dataset. The key steps within the original rationales are highlighted in **purple**, while the corresponding steps generated by Llama2 are indicated in **blue**. In the feedback from the stronger model, errors pointed out are marked in **red**, and corrections are indicated in **green**. In this case, the model successfully corrected its mistake after receiving feedback and arrived at the correct final answer.

Question: A married couple opened a savings account. The wife committed to saving $100 every week while the husband committed to saving $225 every month. After 4 months of savings, they decided to invest half of their money in buying stocks. Each share of stocks costs $50. How many shares of stocks can they buy?

Original Rationales

```
The wife saves $100/week x 4
weeks/month = $400 a month.

Together, the wife and husband save
$400/month + $225/month = $625/month.
In 4 months, their total savings is
$625/month x 4 months = $2500.

They are going to invest $2500 / 2 =
$1250. So, they can buy $1250 /
$50/share = 25 shares of stocks.

The answer is 25.
```

Generated Rationales

```
The wife saves $100/week x 4
weeks/month = $400 a month. (Given)

So, the total monthly savings of the
wife is $400/month x 4 months = $1600.
Therefore, the total monthly savings of
the husband is $225/month x 4 months =
$900. Therefore, the total monthly
savings of the couple is $1600 + $900 =
$2500.

Therefore, the total money they can
invest is $2500/2 = $1250. Therefore,
they can buy $1250/$50 = 25 shares of
stocks.

The answer is 25.
```

Figure 10: An example of **state reset** on the GSM8K dataset. The key steps within the original rationales are highlighted in **purple**, while the corresponding steps generated by Llama2 are indicated in **blue**. The (Given) highlights the partial rationale provided to the model as a hint. In this case, the model starts from the intermediate steps and gives a different solution, enriching the rationales in the training data.

Input:

Below is a math problem, please give a step-by-step answer.

Question:
{question}

Your step-by-step answer:

Figure 11: Prompt template for vanilla self-improve method. It is also used in the first step of GSI (Interactive Sampling), with the process transitioning into the subsequent interactive steps when incorrect results are sampled.

> **Input:**
>
> Below is a math problem, please give a step-by-step answer.
>
> ### Question:
> {question}
> {answer}
>
> ### Your step-by-step answer:

Figure 12: Prompt template for GSI (Answer-driven) method.

> **Input:**
>
> Below is a math problem with a reference answer. Using the reference answer as a guide, write your own answer.
>
> ### Question:
> {question}
>
> ### Reference Answer:
> {rationale}
>
> ### Your detailed, complete and step-by-step answer:

Figure 13: Prompt template for GSI (Rationale-driven) method.

> **Strong Model Input:**
>
> You're a patient teacher who corrects mistakes and guides students, helping them find the correct answers on their own. For the following math problem, the original solution is incorrect. Please identify the incorrect step and explain why it is incorrect.
>
> ### Question:
> {question}
>
> ### Student's original wrong answer:
> {wrong_answer}
>
> ### Correct reference answer:
> {rationale}
>
> ### Your correction:

Figure 14: Prompt template for the strong model in GSI (Interactive Sampling) method.

Input:

Below is a correction to your previous solution. Review this carefully and use it to re-vise your solution. Ensure that it includes all necessary steps clearly and thoroughly.

Question:
{question}

Your original wrong answer:
{wrong_answer}

Correction and guidance:
{correct_message}

Your revised, complete step-by-step solution:

Figure 15: Prompt template for the self-improved model in GSI (Interactive Sampling) method.

Input:

Below is a math problem, please give a step-by-step answer.

Question:
{question}

Your step-by-step answer:
{partial_rationale}

Figure 16: Prompt template for GSI (State Reset) method. The partial rationale is truncated from the complete one and placed at the end to guide the model in completing it.