Fixing Distribution Shifts of LLM Self-Critique via On-Policy Self-Play Training

Rong Bao[1], **Donglei Yu**[2], **Kai Fan**[3,*], **Minpeng Liao**[3,*]
[1] College of Computer Science and Artificial Intelligence, Fudan University
[2] University of Chinese Academy of Sciences [3] Tongyi Lab
[1]`rbao22@m.fudan.edu.cn` [3]`{k.fan,minpeng.lmp}@alibaba-inc.com`

Abstract

Self-critique mechanisms significantly improve the performance of language models in complex reasoning tasks by giving them the ability to correct errors, conduct induction and deduction, and switch thinking insights. However, synthetic data methods often require human-introduced errors or sampling of the model's reasoning results from the previous moment, and the current output distribution of the model cannot be obtained, makes the data for critique and reasoning face the problem of distribution shifts. In this work, we propose an on-policy reinforcement learning framework to synchronize the reasoning and critique capabilities of language models. To alleviate reward hacking caused by outcome-based supervision, we design a deliberate reward framework for different purposes. The reward framework not only supervises the model reasoning process based on the results, but also uses Monte Carlo sampling to give appropriate rewards to the critique content according to the success rate of the model's correction after critique. In addition, we introduce a rule-based reward function to impose penalties on the model when it generates hallucinatory critiques. When our approach is applied to the DeepSeek-Math-7B-Base and Qwen2.5-7B-Base models, model performance improves 5.40 and 3.66 points, respectively, compared to the best baseline approach. This validates the significant advantages of our method in improving model's reasoning and self-critique capability. Code will be made available at https://github.com/rbao2018/SCOP.

1 Introduction

Large language models, which are growing in model scale, show impressive performance in a wide range of linguistic tasks (OpenAI, 2023; Touvron et al., 2023; Anthropic, 2023), but their performance on complex reasoning tasks is still significantly lower than the human level (Bubeck et al., 2023; Gulati et al., 2024). Formally distinct from the de facto reasoning process of language models, human reasoning processes incorporate behaviors such as reflection, critique, induction and deduction (Hegel et al., 1991; Kahneman, 2011), rather than relying solely on a monotonous chain-of-thought generation (Wei et al., 2022). Inspired by this, a growing body of research has attempted to enhance the self-critique capabilities of language models (Shinn et al., 2023; Jaech et al., 2024), enabling them to evaluate previously generated thought processes and thereby improve performance in complex reasoning tasks.

To enhance the the self-critique ability of the model, the common approaches currently are to use iterative synthesis of critical data (Zheng et al., 2024b) and reinforcement learning methods based on result supervision (Trung et al., 2024). However, these two approaches face the challenges of distributional shift and reward hacking, respectively. The data synthesis method allows for the manual insertion of error points or the direct sampling of the model's incorrect outputs, and it relies on a more powerful annotator model to synthesize critical data (Xi et al., 2024b; Tang et al., 2025). However, neither manually introducing errors nor sampling the reasoning results in the previous round is the real current output distribution of the model. Distribution shifts between critique and reasoning data hinder the model's ability to effectively critique its own current reasoning outputs (Kumar et al., 2024). While reinforcement learning methods with outcome supervision have been proven to be effective, outcome supervision only focuses on the correctness of the final result (Xi et al., 2024a). Due to the model hallucinations, it may still output incorrect answers even after generating correct critiques, and vice versa. Consequently, outcome-only supervision will lead to reward hacking by encouraging model hallucinations (Zheng et al., 2024a).

* Corresponding authors.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 17680–17700
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

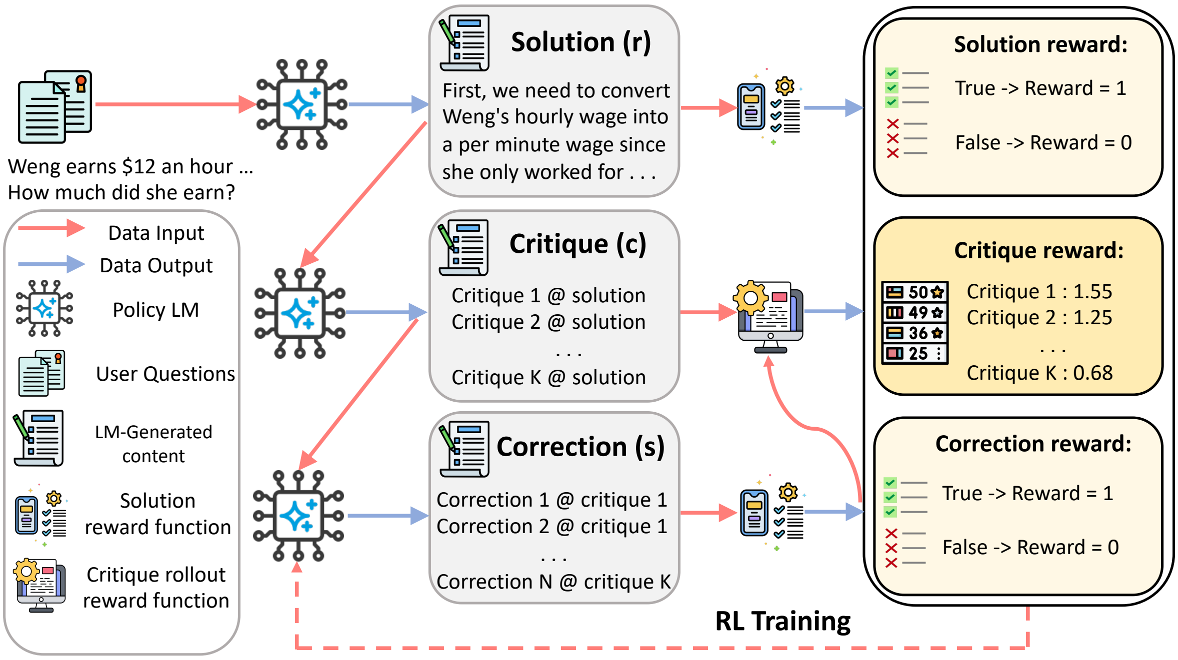


Figure 1: A schematic illustration of the **Self-Critique via On-Policy (SCOP)** training framework for enhancing a large language model's self-critique and reasoning abilities. The model relies on self-play to generate an reasoning-critique-correction sequence, and uses a multi-objective reward function to calculate the reward values for each component. Finally the reinforcement learning algorithm uses the reward values for policy model optimization.

In this work, we propose SCOP (Self-Critique via On-Policy training) to achieve the simultaneous optimization of the model's reasoning and critiquing abilities. This framework addresses the distribution shift issue by generating reasoning and critiquing data via the model's self-play, and uses a deliberate reward function for different purposes to mitigate reward hacking. Specifically, after the model generates the reasoning process for a problem, it will generate a critique based on this context and then generate multiple corrections according to the critique. The multi-objective reward function consists of two parts: Firstly, an automatic verifier is used to verify whether the reasoning and the final correction match the gold answer, and rewards are given to them respectively. Secondly, Monte Carlo sampling is employed to generate multiple corrections after the critique, and the accuracy of the model's corrections serves as a reward for the previous critique. In addition, based on the hand-designed rules, we penalize the model by reducing its reward score when it is detected that the model generates hallucinatory critiques. This framework enables the model to continuously generate reasoning-critique-correction sequences and perform reinforcement learning based on multi-objective rewards, enhancing the model's self-critique while strengthening the its reasoning ability.

Comprehensive experiments conducted on DeepSeek-Math-7B-Base and Qwen2.5-7B-Base show that SCOP has significant performance advantages on the benchmark datasets, with the average scores being 4.61 points and 3.25 points higher than the best benchmark method, respectively.

We summarize our contributions as follows:

- We propose SCOP, a reinforcement learning framework that relies solely on the model's self-play to simultaneously improve its self-critique and reasoning abilities.
- We design a multi-objective reward function to supervise the model's reasoning and critique abilities respectively, which alleviates reward hacking issues in outcome supervision.
- We conduct extensive experiments across multiple base models and datasets, demonstrating our method's significant outperformance over baselines. Additionally, we perform ablation studies and analyses to elucidate the training mechanisms that develop self-criticism and correction abilities in our framework.

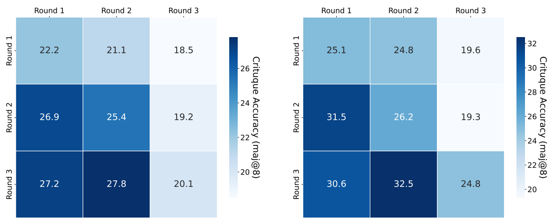
2 Related Work

LLM Self-Critique The self-critique mechanism (Leike et al., 2018; Saunders et al., 2022) enables the model to utilize additional computational resources at the time of inference to make appropriate modifications to its own output, thereby improving the quality of the response. This technique has demonstrated significant improvements in aspects such as the generalization of reward models (Ye et al., 2024; Yu et al., 2024), code generation (McAleese et al., 2024), and moral reasoning (Ganguli et al., 2023). In terms of improving the reasoning ability of language models, many studies have attempted to enhance the model's reasoning ability through self-critique capabilities without the aid of external demonstration (Wang et al., 2025; Huang et al., 2024), but are shown to be of little or even negative effect when the generalized ability of the model is limited. In contrast, enhancing the self-critique ability of the model with the help of external information is usually more effective (Xi et al., 2024b). Nevertheless, they are still subject to the failure of the method due to limited model capacity (Ke et al., 2024), data distribution shifts (Zheng et al., 2024b), and lack of high quality external demonstration (Tang et al., 2025).

Reinforcement Learning for LLMs Reasoning It has been proven that using reinforcement learning to enhance the reasoning ability of models is highly effective (DeepSeek-AI et al., 2025; Guan et al., 2024; Xiang et al., 2025). After training, models can exhibit behaviors similar to human reflection and exploration. Outcome based rewards may increase the risk of model hallucination (Trung et al., 2024; Kumar et al., 2024), and sparse rewards may also lead to the collapse of the policy model (Xi et al., 2024a). Although process reward models allow for fine-grained supervision of processes, training process reward models requires large amounts of high-quality human labeled data, which is economically expensive and requires continuous iteration to effectively supervise the policy model (Lightman et al., 2023; Zheng et al., 2024a). Our proposed approach avoids training reward models through multi-objective reward modeling, which reduces the cost and circumvents the risk of reward hacking.

3 Preliminary Analysis

Iterative synthesis of critique data often requires either artificially inserting error points into correct reasoning paths or sampling the incorrect reasoning process of the previous round of models to synthesize critique data. However, these data do not match the real output distribution of the model when it is fine-tuned using the critique data, and the resulting problem of distributional shift in turn leads to the failure of the model's self-critique training. Following previous work (Zheng et al., 2024b), we conduct an experiment on iterative synthesis of self-critique data, and the results are shown in Figure 2. We separate the models for each round in the iteration and used the model obtained in each round to criticize the others' reasoning output.



(a) Deepseek-Math-7B-Base (b) Qwen2.5-7B-Base

Figure 2: The **Distribution Shift** problem in iterative critique data synthesis. In our experiments, the model from each round is used to critique the reasoning of all rounds. It is observed that the model is more effective in critiquing the reasoning of the previous round rather than the current round.

As can be seen in Fig.2, the model under the current training round usually performs best in terms of its ability to critically evaluate the model generated in the previous round. In contrast, when the model evaluated the reasoning process generated in other rounds, its effectiveness decreased significantly. This experimental result provides motivation to propose an on-policy approach to self-critique training, and more details of the analysis and implementation can be found in Appendix B.1.

4 Methodology

Method Overview Our training framework enables the model to continuously learn during the self-play reasoning, critiquing, and correcting processes, which achieves simultaneous optimization of multiple rewards while avoiding distributional shift, thus improving the model's generalization ability in reasoning tasks. As shown in Fig.1, we divide the model's reasoning, critiquing, and correcting processes into three generation stages, each of which relies on the previous context for generation. After obtaining the rewards of each part, a reinforcement learning algorithm is used to maximize the overall reward value.

In terms of the training process, we elaborate in

§4.1 the initial self-critique data synthesis session with the initialization of the policy model, which is the starting point for the model to be able to reason, criticize and correct. After that, we design the reward functions corresponding to different generation stages. We supervise the reasoning and correction of the model based on their final results, and reward the model's critique using the accuracy rate of subsequent corrections and hand-crafted rules (§4.2). Finally, §4.3 introduces the policy gradient reinforcement learning algorithm to simultaneously maximize the rewards of multiple objectives.

4.1 Data Synthesis and Policy Initialization

Let the initial policy model be denoted as π_θ, where θ represents the pre-trained weights. Let the dataset be $D = \{x, r, y\}$, where x and r represent the question and the corresponding golden reasoning process respectively, and y represents the final answer to the question. First, we use the given training dataset to fine-tune a policy model so that it can provide the corresponding reasoning process $\hat{r}$ for the question x, and the final answer $\hat{y}$ in the reasoning can be regularly matched. For this model, we will sample multiple reasoning processes for each question in the training set, and then mark these reasoning processes as r^+ and r^- respectively according to the final answers of the reasoning.

We use an additional annotator model g_ϕ to generate corresponding critiques c and corrections s for all reasoning processes (r^+ and r^-) while referring to the golden solution process. In addition, to improve the quality of the critique data generated by the annotator model, we adopt a post-validation algorithm to filter the data (see more in Appendix B.2). The resulting dataset is then divided into two parts $D^+ = \{x, r^+, c^{r+}, s^{r+}\}$ and $D^- = \{x, r^-, c^{r-}, s^{r-}\}$. Then we use multi-task supervised learning to get π_θ^{SFT}, such that this model can generate successive reasoning-critique-correction. The loss function is shown below:

$$
\begin{aligned}
\mathcal{L}^+(\theta) &= -\sum_{(p,r^+,c^{r+},s^{r+})\in\mathcal{D}^+} \log \pi_\theta(r^+, c^{r+}, s^{r+}|x) \\
\mathcal{L}^-(\theta) &= -\sum_{(p,r^-,c^{r-},s^{r-})\in\mathcal{D}^+} \log \pi_\theta(c^{r-}, s^{r-}|r^-, x) \\
\mathcal{L}(\theta) &= \mathcal{L}^+(\theta) + \lambda \cdot \mathcal{L}^-(\theta)
\end{aligned}
\tag{1}
$$

where λ is a weight hyperparameter, and in Appendix A.2 we will further analyze the effect that the value of λ has on the model's initial reasoning ability and self-critique ability.

4.2 Reward Design

In order to ensure that the model is effectively monitored and optimized during the reasoning, critiquing and correction processes, we designed separate reward functions for each stage. The main core ideas include the following points:

- Use a verifier to evaluate the correctness of the reasoning results generated by the model, and then provide an appropriate reward signal.
- The value of the self-critique reward for the model depends on the proportion of correct answers in the subsequent correction process. We also apply hand-crafted to detect model hallucinations in critiques and impose appropriate penalties.
- In the final correction phase, we also use the correctness of the answers as the basis for rewards. And if the final correction is correct, we will give an additional reward bonus.

We define the following reward function, where end_of_x denotes the last token of reasoning, criticizing, and correcting, respectively.

$$
r(s_t, a_t) = \begin{cases} 1, & \text{if } a_t = end\_of\_r \text{ and } \hat{y}_r = y \\ \frac{m}{n} + h(c), & \text{if } a_t = end\_of\_c \\ 1 + \alpha, & \text{if } a_t = end\_of\_s \text{ and } \hat{y}_s = y \\ 0, & \text{otherwise} \end{cases}
$$

In the above formula, n denotes the number of corrections s sampled by Monte Carlo after the critique c, m is the number of correct answers obtained in these corrections, α is the bonus value for correct corrections, and $h(c)$ is the hallucination penalty function. When hallucinations in critique c are detected, $h(c) = -1$. If there is no hallucination, $h(c) = 0$. For more information on the effect of varying these hyperparameters on the performance of the policy model, you can refer to the supplementary experiments in the Appendix A.1.

4.3 Policy Optimization

In policy optimization, we integrate the reward functions of all stages to achieve a joint improvement in the model's reasoning, critiquing, and correcting abilities. At the same time, to prevent the policy model after reinforcement learning from deviating too much from the initial point, we use the

Table 1: Model performance on test datasets with self-critique mechanism, with the performance obtained by our method marked in blue. The "critique maj@8" indicates the final corrected performance that we obtain after filtering the critiques using self-consistency. Compared to both other reinforcement learning algorithms (REFT, SCoRe) and iterative critique data synthesis methods (Critic-CoT), our approach has significant advantages and further improves the model inference performance when filtering critical comments using self-consistency.

METHOD	Datasets (Accuracy)				Datasets (BoN@16)				AVERAGE
	GSM8K	SVAMP	MATH-OAI	Olympiad	GSM8K	SVAMP	MATH-OAI	Olympiad	
Open-Sourced Instruct Models									
LLaMA3.1-8B	76.68	80.34	47.20	15.36	88.00	89.76	74.20	44.82	54.90/74.20
Deepseek-Math-7B	88.29	87.69	52.40	19.04	92.66	97.60	78.40	48.02	61.86/79.17
Qwen2.5-7B	91.21	93.09	74.00	36.38	96.38	97.88	86.60	56.06	73.67/84.23
Math-Specialized Base Model: Deepseek-Math-7B									
Pair-SFT	85.69	85.52	51.20	18.42	90.55	94.47	75.40	44.59	60.21/76.50
REFT	86.69	86.73	53.40	21.58	94.72	95.24	76.00	45.61	62.35/78.14
SCoRe	88.29	87.25	53.60	22.77	94.25	95.92	76.20	46.39	62.98/78.19
+ critique maj@8	88.49	87.43	56.60	24.14	94.91	97.57	76.80	47.03	64.17/79.08
Critic-CoT (Iter@3)	87.99	87.36	54.80	23.37	94.06	97.24	76.60	47.75	63.38/78.91
+ critique maj@8	88.79	88.90	55.20	24.85	94.86	98.23	77.40	49.69	64.44/80.05
Ours	92.74	93.95	60.40	28.04	**98.34**	99.06	81.60	56.89	68.78/83.97
+ critique maj@8	**93.21**	**93.68**	**60.60**	**29.88**	97.32	**99.53**	**82.80**	**57.12**	**69.34 / 84.19**
General Base Model: Qwen2.5-7B-Base									
Pair-SFT	90.76	92.41	72.60	33.54	94.35	96.21	84.60	55.77	72.33/83.73
REFT	91.14	93.39	74.40	35.37	95.85	97.38	86.00	56.15	73.58/83.84
SCoRe	91.40	94.44	74.80	35.04	95.62	97.68	86.20	56.30	73.92/83.96
+ critique maj@8	92.76	93.21	75.80	35.97	95.21	97.73	87.20	57.12	74.44/84.31
Critic-CoT (Iter@3)	92.26	94.45	74.60	37.71	95.50	97.06	87.60	57.24	74.75/84.35
+ critique maj@8	93.38	94.72	75.40	38.41	95.67	97.60	88.80	58.74	75.48/85.19
Ours	95.49	97.06	78.60	42.48	**97.07**	**98.76**	90.40	62.39	78.41/87.16
+ critique maj@8	**96.53**	**97.15**	**79.10**	**43.45**	97.36	98.32	**91.40**	**63.23**	**79.06 / 87.58**

KL divergence penalty term to penalize the overall reward and use the PPO (Schulman et al., 2017) to maximize this objective. The final objective to be optimized is as follows:

$$\max_{\theta} \mathbb{E}_{\tau \sim \pi_\theta(\cdot|x)} \left[\sum_{t=0}^{T} r(s_t, a_t) - \beta D_{KL}(\pi_\theta(\cdot|x) || \pi_\theta^{SFT}(\cdot|x)) \right] \tag{2}$$

where β represents the sparsity of the KL penalty, π_θ^{SFT} represents the initial policy model with frozen parameters, and $\tau = [r, c, s]$ is the response generated by the policy model for question x.

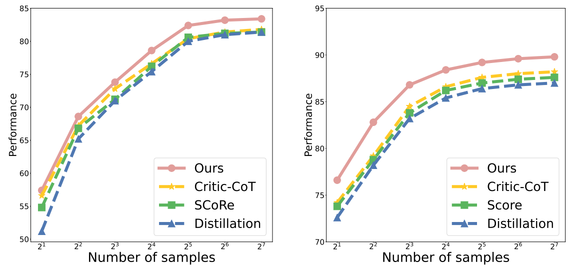
5 Experiments

5.1 Experimental Setup

The experiment is based on the Deepseek-Math-7B-Base (Shao et al., 2024) and Qwen2.5-7B-Base (Yang et al., 2024) models. Among them, the deepseek-math-7B-base model is a specialized model for the mathematics domain, while the qwen2.5-7B-Base model has more comprehensive capabilities. The Qwen2.5-72B-Instruct model was used as the annotator model for synthesizing critique data. For more details about the experimental settings, please refer to Appendix B.

Datasets We use the stronger annotator model g_ϕ to regenerate the corresponding reasoning content r for each question x in the training sets of GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) datasets. We use these data to construct the Pair-SFT dataset as described in §4.1 for training the initial policy model π_θ^{SFT}. In the subsequent reinforcement learning process, we use the cleaned Numina-CoT (Jia LI and Polu, 2024) dataset to obtain the question-answer dataset. The cleaning operation mainly eliminates proof questions without standard numerical answers.

Baseline Methods The baseline methods compared in this study are primarily the critique data synthesis and outcome-based reinforcement learning methods. In terms of critique data synthesis, the methods we compare include Pair-SFT as well as Critic-CoT (Zheng et al., 2024b), which mainly employs iterative critique data synthesis techniques to train the self-critique ability of the



(a) Deepseek-Math-7B-Base (b) Qwen2.5-7B-Base

Figure 3: We plotted the performance scaling trends of the models under exponentially growing majority voting mechanisms using the two models separately. All of the methods compared in the figure train the models to generate text that conforms to the reasoning-critique-correction format, and model performance is calculated using the correctness of the final correction answer.

model through supervised learning. In addition, we also conducted a comparative analysis of some reinforcement learning algorithms, including REFT (Trung et al., 2024) and SCoRe (Kumar et al., 2024) algorithms. These algorithms all utilize result supervision to implement reward allocation for the reasoning content of the model.

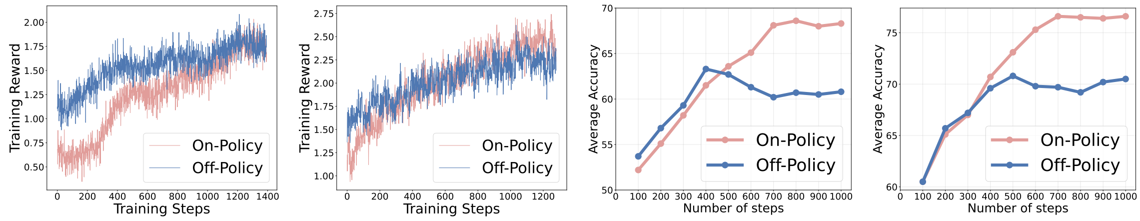
Evaluation We mainly use **Accuracy**, **BoN@N** and **Maj@N** to evaluate the performance of the policy model. The Accuracy metric is the performance of the model when decoding once using greedy decoding. BoN@N indicates whether the model is able to get the correct answer at least once in N times of generation, and Maj@N is the highest-frequency answer accuracy chosen by the model using self-consistency in N times of generation. The default decoding temperature T is set to 0.6. The evaluation dataset for this experiment contains test sets of GSM8K, MATH-OAI (Lightman et al., 2023), SVAMP (Patel et al., 2021), Olympiad (He et al., 2024) datasets, and an out-of-domain (OOD) dataset, OCW (Lewkowycz et al., 2022). These datasets cover problems of varying reasoning difficulty and enable a comprehensive assessment of the model's reasoning capabilities.

5.2 Main Results

Benchmark Performance As shown in Table 1, our method demonstrates better average performance on both models and all test datasets. We summarize some important results as follows: 1) In the greedy decoding setting, compared with the strongest baseline method (the Critic-CoT method with three rounds of iteration and without using self-consistency to filter critiques during inference), our method improves by 5.40% and 3.65% respectively (for the two base models). Moreover, our method exhibits more significant advantage compared with other baseline approaches. 2) While the performance gap between our method and baseline methods is considerable under greedy decoding, this difference narrows when examining the BoN@16 metric. This observation suggests that although expanding the search breadth during testing benefits our approach, it provides an even greater advantage to other methods. Our method shows further potential for performance enhancement when employing self-consistency to filter critiques. In contrast, methods based on iterative synthesis of critique data or other reinforcement learning approaches demonstrate minimal improvement in critique selection when utilizing self-consistency. This disparity highlights our method's superior ability to enhance the model's capacity for effective corrections based on critique opinions.

Scaling Trend We plot the performance curves of various methods on the MATH-OAI dataset in accordance with the test time scaling law. As illustrated in Fig.3, our method demonstrates a more pronounced advantage over other baseline approaches in terms of inference time scaling across different base models. However, as the computation increases during the inference phase, we observe that while our advantage gradually diminishes on the Deepseek-Math-7B-Base model, it remains substantial on the Qwen2.5-7B-Base model. One possible explanation is that the Qwen model pays more attention to the diversity of data during pre-training, which results in the pre-trained model itself having a stronger self-critique ability.

The Quantitative Benefits of Self-Critique As demonstrated in Table 2, our quantitative analysis reveals significant performance gains attributed to the self-critique mechanism. Using Deepseek-Math-7B as the base model, our method achieves an Acc@S metric of 57.4, markedly surpassing other approaches, and a $\Delta^{F\to T}$ metric of 7.2. Similarly, with Qwen2.5-7B-Base, we attain an Acc@S of 76.6 and a notably high $\Delta^{F\to T}$ of 11.7, underscoring our model's efficacy in self-critique and correction. However, it is noteworthy that several benchmark methods surpass our approach in terms of the Acc@R metric. This observation aligns with the core emphasis of our method: we do not solely prioritize immediate responses. Instead, our ap-



(a) Training Reward (b) Training Reward (c) Evaluation Accuracy (d) Evaluation Accuracy

Figure 4: The reward curve and performance changes under the setting of the on/off policy training. The two sub-figures **a** and **b** are the reward curves of Deepseek-Math-7B-Base and Qwen2.5-7B-Base during training, respectively. Figure “c“ and “d“ show the model performance on the test datasets at different training steps.

Table 2: Model performance of different methods obtained using greedy decoding with self-critique mechanism. Our methods are shown using blue rows, while the optimal indicator for each column is **bolded**. The metric $\Delta(R, S)$ indicates the accuracy gap between the reasoning and correction answers, with $\Delta^{F \to T}$ indicating the correct correction from the incorrect reasoning after critiques and $\Delta^{T \to F}$ vice versa.

Method	Acc@R	Acc@S	$\Delta(R,S) \uparrow$	$\Delta^{F\to T} \uparrow$	$\Delta^{T\to F} \downarrow$
Math-Specialized Base Model: Deepseek-Math-7B					
Pair-SFT	51.9	51.2	−0.7	4.8	5.3
REFT	51.9	53.4	1.5	7.1	5.6
SCoRe	52.3	53.6	1.3	6.4	5.1
Critic-CoT	**53.8**	54.8	1.0	4.9	3.9
Ours	53.2	**57.4**	**4.2**	**7.2**	**3.0**
General Base Model: Qwen2.5-7B-Base					
Pair-SFT	**73.9**	72.6	−1.3	4.4	5.7
REFT	71.9	74.4	2.5	8.3	5.8
SCoRe	72.1	74.8	2.7	7.9	5.2
Critic-CoT	72.7	74.6	1.9	7.0	**5.1**
Ours	71.3	**76.6**	**5.3**	**11.7**	6.4

proach focuses on iteratively refining the reasoning process through subsequent self-critiques, ultimately yielding superior overall performance.

6 Analysis and Discussion

6.1 Offline Policy Training to Simulate Iterative Critique Synthesis

We fix the inference output of the model as the experimental setup for the off-policy scenario, while in the on-policy scenario, the model needs to respond according to the user question x and then generate the subsequent content r, c, s. This setup simulates the sampling inference process of the models obtained in previous training rounds in the critique data synthesis method. As shown in Fig.4, due to the limitations of the Deepseek model's own capabilities, even when optimization reaches its peak, the difference in maximum reward values obtainable by on-policy and off-policy strategies is

Table 3: The impact of varying the number of sampled critiques and corrections in reinforcement learning on the model's ultimate performance. It is important to emphasize that the reported model performance was achieved under the condition of sampling critiques and corrections only once during the inference process.

Models	Number of Critique	Number of Correction	Accuracy	BoN@16
Deepseek	4	4	68.34	82.19
	4	16	69.73	82.98
	16	4	70.44	84.45
	16	16	70.71	84.93
Qwen2.5	4	4	78.01	86.51
	4	16	78.28	86.82
	16	4	81.07	89.10
	16	16	82.69	89.73

not significant. In the context of the test datasets, a notable contrast emerges between the off-policy and on-policy training methods. The model trained via the off-policy approach exhibits an initial increase in performance followed by a subsequent decline. Conversely, the on-policy trained model demonstrates consistent improvement until convergence. This divergence can be attributed to two key factors in off-policy training: firstly, the model's critiquing capabilities fail to evolve in tandem with its reasoning abilities; secondly, the rigid reasoning process inherent to the model restricts its exploratory capacity, ultimately leading to performance degradation.

6.2 Critique and Correction Exploration

We investigate the impact of varying the number of critique and correction samples in the policy model's reinforcement learning process on its final performance. Table 3 presents the model's average performance across four mathematical test datasets. Our experiments reveal that increasing the sampling numbers for both critique and correction pos-

Table 4: We utilize the OCW dataset to evaluate the model's OOD performance. Notably, we employ a repeated problem solving process to obtain the Maj@N and BoN@N performance metrics. This approach involves reiterating the problem solving procedure based on the self-consistency of critique or correction.

Methods	Maj@4	Maj@16	BoN@16
Correction Maj@8			
Pair-SFT	43.01	50.00	58.46
SCoRe	44.12	53.68	62.87
Critic-CoT	44.49	54.41	63.24
Ours	50.00	59.93	69.85
Critique Maj@8			
Pair-SFT	45.96	53.68	63.24
SCoRe	49.26	55.51	66.54
Critic-CoT	50.74	57.35	68.01
Ours	53.31	62.50	72.79

itively influences performance. For instance, the Deepseek model's accuracy improved from 68.34% to 70.71%, and its BoN@16 score rose from 82.19 to 84.93 when the critique-correction numbers are increased from 4-4 to 16-16. The Qwen2.5 model exhibited a similar trend with even better performance. Notably, for both models, increasing the number of critiques led to a more substantial improvement in accuracy compared to increasing corrections under similar conditions. This suggests that a higher sampling number of critiques encourages more comprehensive exploration by the model, resulting in greater performance gains.

6.3 Out-of-Domain Performance

We conduct comprehensive experiments on the OCW dataset utilizing the Qwen2.5 base model. As illustrated in Table 4, our proposed method demonstrates remarkable efficacy in enhancing the model's self-critique capabilities. Across all experimental configurations, our approach consistently outperforms Pair-SFT, SCoRe, and Critic-CoT across every metric. This consistent superiority underscores the effectiveness of our method in optimizing the model's self-critique and correction abilities, resulting in enhanced generalization on out-of-distribution data. Furthermore, the application of self-consistency for filtering critiques yields more substantial improvements compared to screening model-generated corrections. Notably, the Maj@16 metric increases from 59.93 to 62.50, while the BoN@16 metric exhibits a significant rise from 69.85 to 72.79. These results suggest that the model's self-critique ability constitutes a critical bottleneck for further performance enhancement.

Table 5: Accuracy performance of DeepSeek-Math-7B-Base and Qwen2.5-7B-Base models under a single greedy decoding strategy, with varying KL divergence regularization coefficients β. The optimal performance values in each dataset column are **bolded**.

KL Coefficient β	GSM8K	SVAMP	MATH-OAI	Olympaid
Math-Specialized Base Model: Deepseek-Math-7B				
0.001	89.3	91.3	57.5	25.6
0.01	91.8	92.5	59.8	27.3
0.05	**92.7**	**94.0**	**60.4**	**28.0**
0.1	91.1	92.8	57.3	24.8
0.5	84.5	85.6	53.4	20.2
General Base Model: Qwen2.5-7B-Base				
0.001	93.9	95.3	74.6	40.4
0.01	94.2	96.8	77.1	41.5
0.05	**95.5**	**97.0**	**78.6**	**42.5**
0.1	93.6	95.2	76.5	40.1
0.5	90.5	93.4	72.9	39.6

6.4 KL Divergence Regularization

We evaluate the influence of the KL divergence regularization parameter β in policy optimization summarized in Table 5. The results indicate that performance across all datasets peaks at $\beta = 0.05$ for both DeepSeek-Math-7B-Base and Qwen2.5-7B-Base models. While reducing β to 0.001 causes minimal degradation, increasing β beyond 0.1 leads to significant declines in accuracy. These findings validate the paper's selection of $\beta = 0.05$ as an optimal balance between policy stability and adaptive learning. The observed performance trade-offs underscore the critical role of KL divergence regularization in reinforcement learning-based language model optimization.

6.5 Sampling Efficiency and Training Costs

In the training of strong inference models, increasing the number of samples is crucial for improving model performance; otherwise, the model is prone to policy collapse due to overly sparse rewards. To validate this conclusion, we conduct experiments on the Qwen2.5-7B-Base model by reducing the number of samples in reinforcement learning on all methods from 32 to 1. The results in Table 6 show that the performance of REFT on the MATH-OAI dataset decreased by 19.3%, SCoRe by 16.1%, while our proposed method only decreased by 10.1%. These data indicate that increas-

Table 6: The comparative results of model performance degradation and training costs under different sampling numbers, where critique@4 means that the sampling includes 4 critique steps, and correction@8 indicates that 8 different corrections will be generated after each critique step. We not only show the accuracy changes of the model on the test dataset, but also calculate the number of generated tokens corresponding to different algorithms and sampling numbers.

Number of Sampling	Method	GSM8K	MATH-OAI	Training Tokens per Step
critique@4 correction@8	REFT	91.4	53.4	1,794,348
	SCoRe	91.1	53.6	1,885,475
	Ours	93.1	60.4	1,924,239
critique@2 correction@4	REFT	84.2$_{-7.2}$	44.5$_{-8.9}$	403,952
	SCoRe	85.7$_{-5.4}$	45.2$_{-8.4}$	421,047
	Ours	90.1$_{-3.0}$	54.8$_{-5.6}$	444,481
critique@1 correction@1	REFT	78.3$_{-13.1}$	34.1$_{-19.3}$	52,528
	SCoRe	82.7$_{-8.4}$	37.5$_{-16.1}$	55,282
	Ours	87.8$_{-5.3}$	50.3$_{-10.1}$	58,318

ing samples helps the model find correct inference paths, and our method can more efficiently train the model's self-critique ability, effectively mitigating performance degradation.

Additionally, we ensure that the experiments are conducted with the same sampling budget. For REFT, we directly set the number of samples to 32, and for SCoRe, we use 4 samples in the first round and 8 samples in the second round. To quantify the cost differences between algorithms, we statistically analyze the number of training tokens in a single optimization step and find that our method has similar numbers of training tokens to REFT and SCoRe but demonstrates a significant performance advantage. This demonstrates that our method achieves superior performance with controllable training costs. In the revised version of the paper, we further explore the relationship between computational cost and model performance.

6.6 Key Components Ablation Study

We conduct ablation experiments to evaluate the training processes of two fundamental models. The results, presented in Table 7, reveal several key insights. Notably, reducing the sampling quantity for critique and correction has the least impact on model performance, suggesting that while encouraging model exploration is important, it is not the most essential factor in enhancing self-critique capabilities. Conversely, replacing reinforcement learning with rejection sampling fine-tuning or eliminating process rewards leads to more significant performance declines, underscoring the crucial role of process supervision in cultivating the model's self-critique abilities.

Table 7: Experiments on ablation of key components in training policy models. The components we study include the supervision of reward shaping (critique reward), the number of samples for critique and correction in reinforcement learning, and the use of rejection sampling fine-tuning instead of reinforcement learning.

Models	Methods	Accuracy	Maj@16	BoN@16
Deepseek	Ours	68.78	74.11	83.97
	w reject finetune	63.48	70.83	80.89
	w/o reward shaping	64.22	71.41	81.41
	w/o multi sampling	65.83	72.98	81.83
Qwen2.5	Ours	78.41	81.23	87.16
	w reject finetune	72.45	76.92	83.86
	w/o reward shaping	73.23	77.43	84.46
	w/o multi sampling	74.49	78.97	85.83

7 Conclusion

This study introduces Self-Critique via On-Policy (SCOP) training, an innovative reinforcement learning framework designed to concurrently enhance language models' reasoning and self-critique capabilities. Our framework addresses distribution shifts and reward hacking challenges through self-play and a multi-objective reward function, effectively optimizing both reasoning and self-critique abilities without relying on synthetic data or simplistic outcome-based supervision. Experimental results on DeepSeek-Math-7B-Base and Qwen2.5-7B-Base demonstrate significant performance improvements over baseline methods, validating the efficacy of our training framework.

Limitations

Our work has several limitations. Firstly, computational resource constraints precluded experiments on larger-scale models. Secondly, while effective, the Monte Carlo sampling method used for reward scoring of critiques is computationally intensive compared to traditional discriminative reward modeling approaches, potentially impacting the practicality of the proposed method. Lastly, for problems lacking labeled solution processes, the model's inherent limitations may result in persistent incorrect answers, risking policy collapse and optimization failure.

Acknowledgments

This work was done during an internship at Tongyi Lab and was supported by Alibaba Research Intern Program.

Ethical Consideration

This study aims to enhance the reasoning capabilities of large language models, with a particular emphasis on their performance in mathematical and physical problem-solving. From our research perspective, we have not identified any significant ethical risks associated with this line of inquiry. The focus on improving cognitive abilities in specific domains appears to pose minimal ethical concerns.

References

Anthropic. 2023. Model card and evaluations for claude models.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott M. Lundberg, Harsha Nori, Hamid Palangi, Marco Túlio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with GPT-4. *CoRR*, abs/2303.12712.

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.

Tri Dao. 2024. Flashattention-2: Faster attention with better parallelism and work partitioning. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Deep Ganguli, Amanda Askell, Nicholas Schiefer, Thomas I. Liao, Kamile Lukosiute, Anna Chen, Anna Goldie, Azalia Mirhoseini, Catherine Olsson, Danny Hernandez, Dawn Drain, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jackson Kernion, Jamie Kerr, Jared Mueller, Joshua Landau, Kamal Ndousse, Karina Nguyen, Liane Lovitt, Michael Sellitto, Nelson Elhage, Noemí Mercado, Nova DasSarma, Oliver Rausch, Robert Lasenby, Robin Larson, Sam Ringer, Sandipan Kundu, Saurav Kadavath, Scott Johnston, Shauna Kravec, Sheer El Showk, Tamera Lanham, Timothy Telleen-Lawton, Tom Henighan, Tristan Hume, Yuntao Bai, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, Christopher Olah, Jack Clark, Samuel R. Bowman, and Jared Kaplan. 2023. The capacity for moral self-correction in large language models. *CoRR*, abs/2302.07459.

Melody Y. Guan, Manas Joglekar, Eric Wallace, Saachi Jain, Boaz Barak, Alec Helyar, Rachel Dias, Andrea Vallone, Hongyu Ren, Jason Wei, Hyung Won Chung, Sam Toyer, Johannes Heidecke, Alex Beutel, and Amelia Glaese. 2024. Deliberative alignment: Reasoning enables safer language models. *CoRR*, abs/2412.16339.

Aryan Gulati, Brando Miranda, Eric Chen, Emily Xia, Kai Fronsdal, Bruno de Moraes Dumont, and Sanmi

Koyejo. 2024. Putnam-AXIOM: A functional and static benchmark for measuring higher level mathematical reasoning. In *The 4th Workshop on Mathematical Reasoning and AI at NeurIPS'24*.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. Olympiadbench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. 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 3828–3850. Association for Computational Linguistics.

G.W.F. Hegel, T.F. Geraets, W.A. Suchting, and H.S. Harris. 1991. *The Encyclopaedia Logic, with the Zustze: Part I of the Encyclopaedia of Philosophical Sciences with the Zusatze*. Hackett Classics Series. Hackett.

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*.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2024. Large language models cannot self-correct reasoning yet. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett, Ananya Kumar, Andre Saraiva, Andrea Vallone, Andrew Duberstein, Andrew Kondrich, Andrey Mishchenko, Andy Applebaum, Angela Jiang, Ashvin Nair, Barret Zoph, Behrooz Ghorbani, Ben Rossen, Benjamin Sokolowsky, Boaz Barak, Bob McGrew, Borys Minaiev, Botao Hao, Bowen Baker, Brandon Houghton, Brandon McKinzie, Brydon Eastman, Camillo Lugaresi, Cary Bassin, Cary Hudson, Chak Ming Li, Charles de Bourcy, Chelsea Voss, Chen Shen, Chong Zhang, Chris Koch, Chris Orsinger, Christopher Hesse, Claudia Fischer, Clive Chan, Dan Roberts, Daniel Kappler, Daniel Levy, Daniel Selsam, David Dohan, David Farhi, David Mely, David Robinson, Dimitris Tsipras, Doug Li, Dragos Oprica, Eben Freeman, Eddie Zhang, Edmund Wong, Elizabeth Proehl, Enoch Cheung, Eric Mitchell, Eric Wallace, Erik Ritter, Evan Mays, Fan Wang, Felipe Petroski Such, Filippo Raso, Florencia Leoni, Foivos Tsimpourlas, Francis Song, Fred von Lohmann, Freddie Sulit, Geoff Salmon, Giambattista Parascandolo, Gildas Chabot, Grace Zhao, Greg Brockman, Guillaume Leclerc, Hadi Salman, Haiming Bao, Hao Sheng, Hart Andrin, Hessam Bagherinezhad, Hongyu Ren, Hunter Lightman, Hyung Won Chung, Ian Kivlichan, Ian O'Connell, Ian Osband, Ignasi Clavera Gilaberte, and Ilge Akkaya. 2024. Openai o1 system card. *CoRR*, abs/2412.16720.

Lewis Tunstall Ben Lipkin Roman Soletskyi Shengyi Costa Huang Kashif Rasul Longhui Yu Albert Jiang Ziju Shen Zihan Qin Bin Dong Li Zhou Yann Fleureau Guillaume Lample Jia LI, Edward Beeching and Stanislas Polu. 2024. Numinamath. `[https://github.com/project-numina/aimo-progress-prize](https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf)`.

Daniel Kahneman. 2011. *Thinking Fast and Slow*. Farrar, Straus and Giroux.

Pei Ke, Bosi Wen, Andrew Feng, Xiao Liu, Xuanyu Lei, Jiale Cheng, Shengyuan Wang, Aohan Zeng, Yuxiao Dong, Hongning Wang, Jie Tang, and Minlie Huang. 2024. Critiquellm: Towards an informative critique generation model for evaluation of large language model generation. 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 13034–13054. Association for Computational Linguistics.

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. 2024. Training language models to self-correct via reinforcement learning. *CoRR*, abs/2409.12917.

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

Jan Leike, David Krueger, Tom Everitt, Miljan Martic, Vishal Maini, and Shane Legg. 2018. Scalable agent alignment via reward modeling: a research direction. *CoRR*, abs/1811.07871.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay V. Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. 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*.

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

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Nat McAleese, Rai Michael Pokorny, Juan Felipe Ceron Uribe, Evgenia Nitishinskaya, Maja Trebacz, and Jan Leike. 2024. LLM critics help catch LLM bugs. *CoRR*, abs/2407.00215.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

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.

William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. 2022. Self-critiquing models for assisting human evaluators. *CoRR*, abs/2206.05802.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. 2016. High-dimensional continuous control using generalized advantage estimation. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347.

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.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: language agents with verbal reinforcement learning. 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*.

Zhengyang Tang, Ziniu Li, Zhenyang Xiao, Tian Ding, Ruoyu Sun, Benyou Wang, Dayiheng Liu, Fei Huang, Tianyu Liu, Bowen Yu, and Junyang Lin. 2025. Enabling scalable oversight via self-evolving critic. *Preprint*, arXiv:2501.05727.

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.

Luong Quoc Trung, Xinbo Zhang, Zhanming Jie, Peng Sun, Xiaoran Jin, and Hang Li. 2024. Reft: Reasoning with reinforced 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 7601–7614. Association for Computational Linguistics.

Yubo Wang, Xiang Yue, and Wenhu Chen. 2025. Critique fine-tuning: Learning to critique is more effective than learning to imitate. *Preprint*, arXiv:2501.17703.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. 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*.

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, Qi Zhang, and Xuanjing Huang. 2024a. 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.

Zhiheng Xi, Dingwen Yang, Jixuan Huang, Jiafu Tang, Guanyu Li, Yiwen Ding, Wei He, Boyang Hong, Shihan Dou, Wenyu Zhan, Xiao Wang, Rui Zheng, Tao Ji, Xiaowei Shi, Yitao Zhai, Rongxiang Weng, Jingang Wang, Xunliang Cai, Tao Gui, Zuxuan Wu, Qi Zhang, Xipeng Qiu, Xuanjing Huang, and Yu-Gang Jiang. 2024b. Enhancing LLM reasoning via critique models with test-time and training-time supervision. *CoRR*, abs/2411.16579.

Violet Xiang, Charlie Snell, Kanishk Gandhi, Alon Albalak, Anikait Singh, Chase Blagden, Duy Phung, Rafael Rafailov, Nathan Lile, Dakota Mahan, Louis Castricato, Jan-Philipp Franken, Nick Haber, and Chelsea Finn. 2025. Towards system 2 reasoning in llms: Learning how to think with meta chain-of-thought. *Preprint*, arXiv:2501.04682.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2024. Qwen2.5 technical report. *CoRR*, abs/2412.15115.

Zihuiwen Ye, Fraser Greenlee-Scott, Max Bartolo, Phil Blunsom, Jon Ander Campos, and Matthias Gallé. 2024. Improving reward models with synthetic critiques. *CoRR*, abs/2405.20850.

Yue Yu, Zhengxing Chen, Aston Zhang, Liang Tan, Chenguang Zhu, Richard Yuanzhe Pang, Yundi Qian, Xuewei Wang, Suchin Gururangan, Chao Zhang, Melanie Kambadur, Dhruv Mahajan, and Rui Hou. 2024. Self-generated critiques boost reward modeling for language models. *CoRR*, abs/2411.16646.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, Alban Desmaison, Can Balioglu, Pritam Damania, Bernard Nguyen, Geeta Chauhan, Yuchen Hao, Ajit Mathews, and Shen Li. 2023. Pytorch FSDP: experiences on scaling fully sharded data parallel. *Proc. VLDB Endow.*, 16(12):3848–3860.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2024a. Processbench: Identifying process errors in mathematical reasoning. *CoRR*, abs/2412.06559.

Xin Zheng, Jie Lou, Boxi Cao, Xueru Wen, Yuqiu Ji, Hongyu Lin, Yaojie Lu, Xianpei Han, Debing Zhang, and Le Sun. 2024b. Critic-cot: Boosting the reasoning abilities of large language model via chain-of-thoughts critic. *CoRR*, abs/2408.16326.

Contents of Appendix

A Supplementary Experiments

A.1 Critique Reward Design

Models	α	$h(c)$	Accuracy	Maj@16
Deepseek	0.2	$\{-1, 0\}$	67.11	82.27
	0.2	$\{0, 1\}$	67.88	82.83
	0.2	$\{-1, 1\}$	68.53	83.68
	1.0	$\{-1, 0\}$	67.18	82.49
	1.0	$\{0, 1\}$	67.85	82.82
	1.0	$\{-1, 1\}$	**68.57**	**83.80**
Qwen2.5	0.2	$\{-1, 0\}$	75.37	85.09
	0.2	$\{0, 1\}$	76.12	85.56
	0.2	$\{-1, 1\}$	77.40	86.83
	1.0	$\{-1, 0\}$	76.13	85.71
	1.0	$\{0, 1\}$	77.51	86.84
	1.0	$\{-1, 1\}$	**78.45**	**87.86**

Table 8: We investigated the impact of key hyperparameters in the reward function on the final policy model's performance. Our primary focus was on examining how the final correction bonus α and the hallucination penalty $h(c)$ for critiqus affect the model's ultimate performance. Across all experiments, we set the number of corrections n generated following each critique to 8.

In instances where the initial reasoning r yields a correct answer, yet the subsequent critique erroneously identifies flaws in the reasoning process, we categorize such critique as hallucination, irrespective of the final correction's accuracy. Conversely, when the reasoning r is flawed but the model's critique fails to detect these errors, we similarly classify this as a hallucination. For scenarios where r produces an incorrect answer and the critique attempts to identify errors, but its accuracy is ambiguous, we employ Monte Carlo sampling to estimate the reward of the critiques.

The design of a multi objective reward function requires balancing the model's reasoning and self-critique abilities. Thus, the additional bonus for final corrected results and the hallucination penalty for critique are important hyperparameters. As shown in Table 8, we experiment with these two hyperparameters using two base models. The results show that incrementing the final corrected reward value α produces only marginal improvements in model performance, a pattern that is consistent across both base models. With respect to the critical hallucination penalty $h(c)$, optimal performance is achieved when both reward and penalty values are set to $\{-1, 1\}$ for both models. This finding suggests that moderately expanding the range of critical illusion penalties is more conducive to the overall improvement of the models.

A.2 Initialize Model Self-Critique Capability

When initializing the self-critique ability of the model using Pair-SFT dataset, the key lies in finding a balance between the dataset D^+ constructed from the correct reasoning paths and the dataset D^- constructed from the incorrect reasoning paths. To achieve this goal, we search for the optimal balance point by adjusting the weight λ in Eq.1.

As can be seen from Fig.5, when the value of λ is too small, the model will tend to only learn the correct reasoning paths, which will lead to poor final accuracy. As λ gradually increases, the model can obtain the best self-critique performance in the vicinity of the range of 0.9 to 1.1. However, when λ is further increased, due to the model's excessive focus on learning the self-critique ability, the proportion of the model learning the dataset of correct reasoning paths in the initial stage is relatively low. If the model cannot solve the problem correctly in the initial stage, it will ultimately lead to a decline in the model's performance. Therefore, in the main experiment, we set the value of λ to 1.0 as the baseline method.

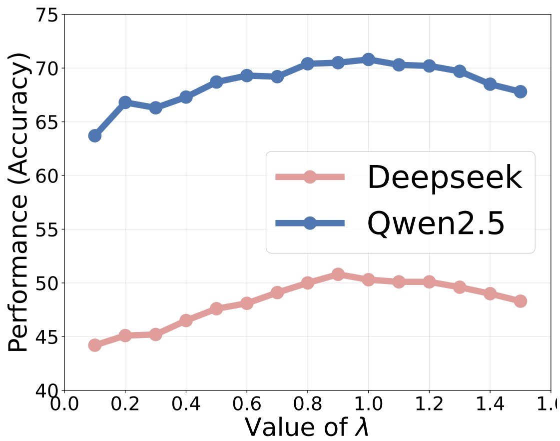
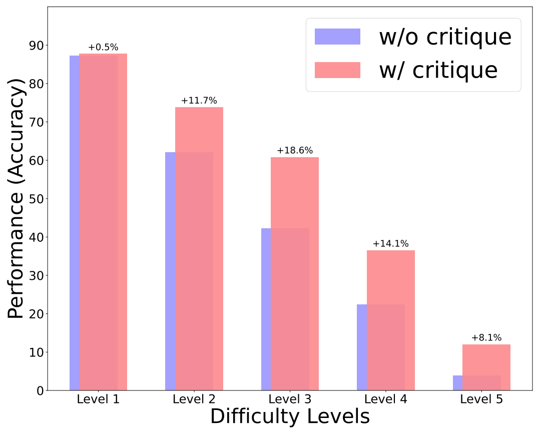


Figure 5: The influence of changing the λ parameter in Eq.1 on the initial self-critique ability of the model. We set the range of λ to be from 0.1 to 1.5 with an interval of 0.1. To evaluate model performance, we utilized the accuracy of the final corrections generated by the model as our primary metric.

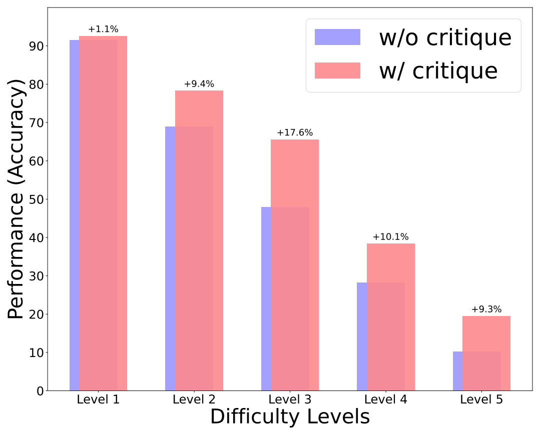
A.3 Critique Benefit Across Difficulty Levels

As shown in the Fig.6, we have conducted an in-depth exploration of the beneficial effects brought about by the self-critique mechanism for problems of different difficulty levels. From an overall perspective, for the two models involved, the self-critique mechanism has achieved performance improvement when dealing with problems of various difficulty levels. Specifically, compared with Qwen2.5-7B-Base, the Deepseek-Math-7B-Base has obtained more significant gains.

A common phenomenon is that the relative benefits of self-critique are less pronounced for both simple and highly complex problems, reaching optimal efficacy for moderately difficult tasks. This observation strongly suggests that the effectiveness of the self-critique mechanism is constrained by the model's inherent capabilities. For straightforward questions, where the model already achieves high accuracy, the additional value of self-critique is limited. Conversely, for extremely challenging problems, the mechanism's implementation does not significantly improve the model's ability to derive correct solutions.



(a) Deepseek-Math-7B-Base



(b) Qwen2.5-7B-Base

Figure 6: The benefits of the self-critique on problems of varying difficulty in the MATH-OAI dataset.

B Implementation Details

In this section, we present some details of the experimental methodology, including the data synthesis method, the hyperparameters in the training framework and the experiment environment.

B.1 Iterative Synthetic Critique Data

Iterative data synthesis requires a powerful annotator model to correct the wrong reasoning paths of the policy model. In our experiment, the Qwen2.5-72B-Instruct model was used as the annotator (§5.1). We referred to previous work (Xi et al., 2024b; Zheng et al., 2024b) and designed the following data synthesis process:

- Assuming that the current model already has the ability of self-critique, we sample the current policy model and identify the content that finally corrects the wrong answers.
- Use a more powerful annotator model g_ϕ to generate corresponding critiques in the context of the question x, the incorrect inference path $\hat{r}$ and the reference answer r_{ref}.
- After screening the correctness of the collected critiques, merge them with the training sets of the previous rounds, and use supervised learning to fine-tune the policy model.
- Iterate the above process multiple times.

During the experiment, we saved the weights of the model in each iteration. For the model obtained in each iteration, sampling operations were carried out during its reasoning process. After the sampling was completed, based on the obtained context information, different models were asked to generate corresponding critiques. In order to ensure the quality and reliability of the critiques, the method of self-consistency was adopted to filter these comments. Finally, the context containing the reasoning content and critiques was input into the model again, so as to generate the final correction process. The prompt template used in the data synthesis process can be obtained in §C.

B.2 Critique Data Synthesis

We rewrite the question-response data using the annotator model g_ϕ. This format includes the "Step" notation, which is convenient for subsequent training. In the data synthesis after obtaining the initial policy model, we use such response-question data as the standard reference answer r_{ref} to help the annotator model generate critiques. We add a filtering operation for the critique data. Specifically, after combining the incorrect reasoning path of the question and the critique, we input them again as the context into the annotator. If at this time the annotator can generate the correct correction with a certain threshold (in our setting, it is that there are at least 6 correct results out of 8 samplings), then we retain this reasoning-critique-correction data point. After such a process, we obtain 60K self-critique data on the GSM8K and MATH datasets. After dividing these data into D^+ and D^-, we fine-tune the policy model using Eq.1.

B.3 Training Hyper-Parameters

Pair-SFT Training We set the coefficient λ for multi-task learning to 1.0 and the training batch size to 256. The learning rates for Deepseek and Qwen2.5 models are 5e-5 and 1e-5, respectively. We apply cosine learning rate decay and use AdamW (Loshchilov and Hutter, 2019) as the optimizer for parameter optimization. More details of the hyperparameters can be found in Table 9.

Table 9: Key hyperparameters for Pair-SFT Training

Hyper-parameter	Value
Training Batch size	256
Optimizer type	AdamW
Learning rate	5e-5/1e-5
lr scheduler type	cosine
Warmup ratio	0.03
Epochs	5
Weight decay	0.0

Policy Optimization In the policy optimization stage, we conduct data filtering for the input prompts of Numina-CoT. We use the annotator model and the initial policy model to generate multiple responses ($n = 8$) for the prompt dataset, and remove the prompts that meet the following conditions: 1) The initial policy model generates the correct answer in all responses. 2) The annotator model does not generate a correct solution process even once. We use the prompt dataset cleaned in this way to perform reinforcement learning on the policy model. In the training process, the GAE (Schulman et al., 2016) parameter λ is set to 0.95, the reward discount coefficient γ is set to 1, and the value model is initialized with the weights of the policy model. More details of the hyperparameters can be found in Table 10.

Table 10: Key hyper-parameters in Policy Optimization

Hyperparameter	Value
Training Batch size	1024
Optimizer type	AdamW
Policy model learning rate	2e-6
Value model learning rate	5e-6
Learning rate scheduler	cosine
Critique Sampling	4
Correction Sampling	8
Maximum training steps	1500
Correction bonus α	1.0
Critique penalty $h(c)$	$\{-1, 1\}$
KL penalty	0.05
Sampling temperature	0.6
Sampling top-p	0.9
Maximum output tokens	2048

B.4 Experiment Environments

All the experiments we conduct are carried out on Linux system machines equipped with 8 NVIDIA A100 GPUs. The code is mainly written in Python[1], and the deep learning framework we adopt is PyTorch [2], with a version of 2.4.0. During the training process, we highly rely on *verl* (Sheng et al., 2024) as the training framework. This training framework integrates projects such as *FSDP* (Zhao et al., 2023), Flash-attention-2 (Dao, 2024), and *vLLM* (Kwon et al., 2023) to accelerate the reinforcement learning process for large language models. A complete reinforcement learning training of the policy model takes about 960 GPU hours. In addition, all the pre-trained model weights we use are derived from *HuggingFace* [3].

C Prompts

We show the prompt words and templates needed in data synthesis, including the template for rewriting question- responses, the template for critique data synthesis, and the prompt for instructing the annotator model to synthesize critique data.

[1] https://www.python.org/
[2] https://pytorch.org/
[3] https://huggingface.co/

C.1 Rewrite Reasoning Process

Rewrite Reasoning Prompt

You are a math expert. Please reason step by step, use "Step X:" to mark the process of each step of your solution. At the end of your solution, give your final answer in the form like "Final Answer:'boxed{}'"

C.2 Critique System Prompt

Critique System Prompt

<role>
You are an expert mathematician specializing in grading math problems.
</role>
<task>
The user will provide a mathematical problem along with "'Correct"' and "'Current"' responses. Carefully analyze both solutions, comparing them step by step. Identify the exact point where the solution deviates from the correct path.
Your output should follow this template:
<critic>
Let's check the current solution step by step
...
<conclution>
So, the current solution becomes incorrect at "'Step X"'.
</conclution>
</critic>
</task>

C.3 Critique Data Synthesis

Input Format for Critique Data Synthesis

System Prompt: [Prompt]
Question: [x]
Current Solution: [r_{ref}]
Incorrect Solution: [$\hat{r}$]
{Annotator Response}

D Case Study

We present two typical examples of successful corrections achieved after self-critique, while showing a case where self-critique failed to succeed.

Success Critique Example 1

Problem. Two different points, C and D, lie on the same side of line AB so that $\triangle ABC$ and $\triangle BAD$ are congruent with $AB = 9$, $BC = AD = 10$, and $CA = DB = 17$. The intersection of these two triangular regions has area $\frac{m}{n}$, where m and n are relatively prime positive integers. Find $m + n$.

The reasoning process (correct).

The task is to find the intersection area of two congruent triangles. To solve this, we can extend AB to form a right triangle with legs 6 and 8 such that AD is the hypotenuse and connect the points CD so that we have a rectangle.
Step 1: Determine the length of base CD of the rectangle
Since we know that $\triangle ADE$ is a $6-8-10$ triangle, and $\triangle DEB$ is an $8-15-17$ triangle, we can calculate the length of base CD of the rectangle by adding the lengths of AB, BE, and ED. Then we have

$$CD = AB + BE + ED = 9 + 6 + 6 = 21$$

Step 2: Set up a proportion to find the height from E to DC
Since $\triangle ABE$ and $\triangle DCE$ are similar with ratio $\frac{21}{9} = \frac{7}{3}$, we can set up a proportion, knowing that the two heights add up to 8. Let y be the height from E to DC, and x be the height of $\triangle ABE$. Then we have

$$\frac{7}{3} = \frac{y}{x}$$

$$\frac{7}{3} = \frac{8-x}{x}$$

Solving for x, we get

$$7x = 24 - 3x$$

$$10x = 24$$

$$x = \frac{12}{5}$$

Step 3: Calculate the intersection area of the two triangles
Given the base AB and the height x of $\triangle ABE$, we can calculate the intersection area of the two triangles by using the formula for the area of a triangle. Then we have

$$A = \frac{1}{2}(9)\left(\frac{12}{5}\right) = \frac{54}{5}$$

Final Answer: $\boxed{59}$

Success Critique Example 1

The critique process (correct).

<critic>
Step 1: Length of $CD = 21$ - Given that we have a $6-8-10$ triangle ($\triangle ADE$) and an $8-15-17$ triangle ($\triangle DEB$) - $CD = AB + BE + ED = 9 + 6 + 6 = 21$
Step 2: Finding the height - Using similar triangles $\triangle ABE$ and $\triangle DCE$ with ratio $\frac{21}{9} = \frac{7}{3}$ - Let x be the height of triangle $\triangle ABE$ and $(8 - x)$ be the height of triangle $\triangle DCE$ - Setting up the proportion: $\frac{7}{3} = \frac{8-x}{x}$
- Solving:

$$7x = 24 - 3x, 10x = 24, x = \frac{12}{5}$$

Step 3: Calculating intersection area - Using the formula: $A = \frac{1}{2}(\text{base})(\text{height})$ - $A = \frac{1}{2}(9)\left(\frac{12}{5}\right)$ - $A = \frac{54}{5}$
Final answer: - $A = \frac{54}{5}$ where $m = 54$ and $n = 5$ - $m + n = 54 + 5 = 59$ ✓ This step is correct
<conclution>
The entire solution is logically sound and mathematically accurate. The answer 59 is correct.
</conclution>
</critic>

Success Critique Example 2

Problem. Suppose $f(x) = \frac{3}{2-x}$. If $g(x) = \frac{1}{f^{-1}(x)} + 9$, find $g(3)$.

The reasoning process (incorrect). To solve the problem, we need to find $g(x)$ and then evaluate it at $x = 3$. Step 1:

Find the inverse function $f^{-1}(x)$ We start with the function $f(x) = \frac{3}{2-x}$. To find the inverse function, we first solve for x in terms of y:

$$y = \frac{3}{2-x}$$

Multiply both sides by $2 - x$:

$$y(2-x) = 3$$

Distribute y on the left-hand side:

$$2y - xy = 3$$

Rearrange the equation to isolate x:

$$x = 2 - \frac{3}{y}$$

Thus, the inverse function is:

$$f^{-1}(x) = 2 - \frac{3}{x}$$

Step 2: Substitute $f^{-1}(x)$ into $g(x)$ We are given:

$$g(x) = \frac{1}{f^{-1}(x)} + 9$$

Substitute $f^{-1}(x) = 2 - \frac{3}{x}$:

$$g(x) = \frac{1}{2 - \frac{3}{x}} + 9$$

Step 3: Simplify the expression First, simplify the denominator:

$$2 - \frac{3}{x} = \frac{2x-3}{x}$$

Now, substitute back into $g(x)$:

$$g(x) = \frac{1}{\frac{2x-3}{x}} + 9$$

Simplify the fraction inside the fraction:

$$\frac{1}{\frac{2x-3}{x}} = x \div \frac{2x-3}{x} = x \cdot \frac{x}{2x-3} = \frac{x^2}{2x-3}$$

Thus,

$$g(x) = \frac{x^2}{2x-3} + 9$$

Step 4: Evaluate $g(3)$ Substitute $x = 3$ into the simplified $g(x)$:

$$g(3) = \frac{3^2}{2(3)-3} + 9$$

$$g(3) = \frac{9}{6-3} + 9$$

$$g(3) = \frac{9}{3} + 9$$

$$g(3) = 3 + 9$$

$$g(3) = 12$$

Final Answer

$$\boxed{12}$$

Success Critique Example 2

The critique process (correct). <critic>

Step 1. Finding the inverse function $f^{-1}(x)$: This step is correct in both solutions.
Step 2. Substituting $f^{-1}(x)$ into $g(x)$: This step is also correct in both solutions.
Step 3. Simplifying the expression: Here's where the deviation begins. In the current solution, the simplification of the fraction is incorrect:
Current solution:

$$\frac{1}{\frac{2x-3}{x}} = x \div \frac{2x-3}{x} = x \cdot \frac{x}{2x-3} = \frac{x^2}{2x-3}$$

This is incorrect. The correct simplification should be:

$$\frac{1}{\frac{2x-3}{x}} = x \cdot \frac{1}{2x-3} = \frac{x}{2x-3}$$

The error is in multiplying by x twice instead of once.
Step 4. Evaluating $g(3)$: Due to the error in step 3, the evaluation of $g(3)$ is also incorrect.
<conclution>
So, the current solution becomes incorrect at Step 3, during the simplification of the expression for $g(x)$. This error leads to an incorrect final answer of 12 instead of the correct answer of 10.
</conclution> </critic>

Failed Critique Example 1

Problem. Three of the edges of a cube are $\overline{AB}$, $\overline{BC}$, and $\overline{CD}$, and $\overline{AD}$ is an interior diagonal. Points P, Q, and R are on $\overline{AB}$, $\overline{BC}$, and $\overline{CD}$, respectively, so that $AP = 5$, $PB = 15$, $BQ = 15$, and $CR = 10$. What is the area of the polygon that is the intersection of plane PQR and the cube?

The reasoning process (correct).

The task is to find the area of the polygon that is the intersection of plane PQR and the cube. We can first analyze the intersection of the plane with the cube, and then calculate the area of the polygon.
1. Analyze the intersection of the plane with the cube: The polygon that is the intersection of plane PQR and the cube is a hexagon. The hexagon can be divided into two quadrilaterals and a rectangle.
2. Calculate the area of the rectangle: The area of the rectangle is $10 \times 15 = 150$.
3. Calculate the area of the quadrilateral on the front: The quadrilateral on the front can be divided into two triangles. The area of the quadrilateral is $2 \times \frac{1}{2} \times 15 \times 15 = 225$.
4. Calculate the area of the quadrilateral on the side:
The quadrilateral on the side can be divided into two triangles. The area of the quadrilateral is $2 \times \frac{1}{2} \times 10 \times 15 = 150$.
5. Calculate the total area of the hexagon: The total area of the hexagon is $150 + 225 + 150 = 525$.
Conclusion: We have calculated the area of the polygon that is the intersection of plane PQR and the cube.
Final Answer: $\boxed{525}$

Failed Critique Example 1

The critique process (incorrect).

<critic>
The current solution has several issues:
1. The first major issue is assuming the intersection forms a hexagon that can be divided into a rectangle and two quadrilaterals. This assumption is not justified.
2. The solution doesn't use the given information about $AP = 5$ and $PB = 15$, which are crucial for determining the position of point P on edge AB.
3. The solution assumes dimensions without proper justification. For example, it uses 10 and 15 as dimensions for the rectangle without proving why these should be the dimensions.
4. The solution doesn't consider that:
- A cube has equal edges.
- The diagonal AD creates a specific geometric relationship.
- Points P, Q, R define a plane that intersects the cube at specific angles.
5. The solution skips the crucial step of:
- Establishing the cube's dimensions.
- Finding the position of diagonal AD.
- Determining how plane PQR intersects other faces of the cube.
<conclution>
The solution becomes incorrect at Step 1 where it makes an unjustified assumption about the shape of the intersection being a hexagon that can be divided into specific parts. The entire approach lacks proper geometric analysis and relies on unfounded assumptions about the dimensions and shapes involved.
</conclution>
</critic>