Enhancing Long-Chain Reasoning Distillation through Error-Aware Self-Reflection

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have exhibited strong reasoning capabilities and achieved remarkable performance in mathematical problem-solving tasks. Recently, distilling reasoning ability from long-form Chains-of-Thought (CoTs) has emerged as a promising approach for enhancing Small Language Models (SLMs). Existing studies typically treat SLMs as student models and use long-form CoTs as supervision signals for Supervised Fine-Tuning (SFT) to transfer reasoning ability. However, such long-form CoT teachers are usually unaware of the student model's capacity, which limits the effective utilization of the provided reasoning traces. To overcome this limitation, we propose err**O**r-aware self-**R**eflect**ION** (ORION), a framework that refines teacher CoTs through an Error-Aware Reflection process. ORION enables the student model to construct more tailored teacher CoTs by refining teacher CoTs and incorporating its own reasoning errors. Experiments on multiple mathematical reasoning benchmarks demonstrate that ORION consistently improves performance by more than 2% over all baselines. Further analysis reveals that the CoTs constructed by ORION exhibit higher coherence and logical consistency, thereby serving as more effective supervision signals for SFT. All code and datasets will be released on GitHub.

1 Introduction

> *Success does not consist in never making mistakes, but in never making the same one a second time.*
>
> — George Bernard Shaw

Large Language Models (LLMs) have demonstrated remarkable reasoning capabilities in mathematical problem-solving (Brown et al., 2020; Zhang et al., 2022). By leveraging the Chain-of-Thought (CoT) paradigm (Wei et al., 2022), the LLMs can decompose and solve mathematical problems step by step (Fu et al., 2023; Li et al., 2023), and achieve better performance. However, for Small Language Models (SLMs), generating faithful and logically coherent CoTs remains a significant challenge, particularly when solving the complex mathematical tasks (Wang et al., 2025).

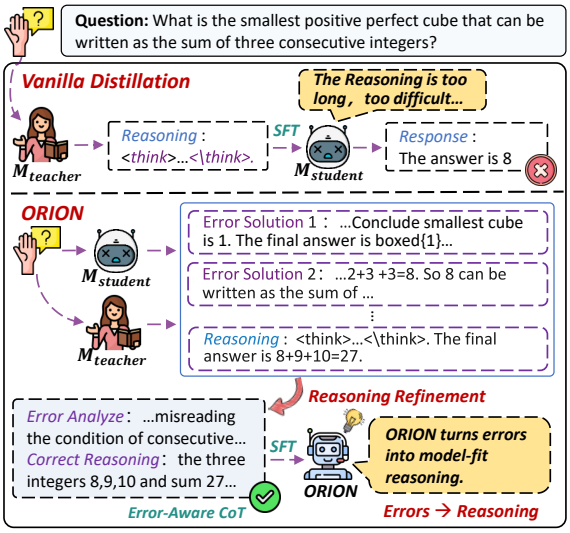


Figure 1: The Framework of Our ORION Model. ORION refines long-form reasoning via self-reflection.

Recent studies have focused on distilling reasoning abilities from long-form CoTs to enhance the performance of SLMs on mathematical reasoning tasks (Ye et al., 2025; DeepSeek-AI et al., 2025). These methods typically regard SLMs as student models and employ long-form CoTs generated by Reasoning Language Models (RLMs) as supervision signals for Supervised Fine-Tuning (SFT), thereby enabling the transfer of reasoning capabilities from teacher to student models (Hsieh et al., 2023). However, during the distillation process, the substantial capability gap between student and teacher models often leads SLMs to passively absorb knowledge from the teacher (Zhou et al., 2022), making it difficult for them to inter-

nalize overly complex or lengthy reasoning steps and thereby limiting the effectiveness of knowledge transfer (Li et al., 2025b; Anonymous, 2025).

To address this challenge, recent studies have explored error-aware learning, which leverages the solution errors generated by SLMs to train them to avoid similar mistakes, thereby improving their mathematical question answering capabilities (An et al., 2023). These approaches typically employ superior LLMs to revise the incorrect solutions generated by SLMs, treating the corrected solutions as training targets while using the original erroneous solutions as input to teach SLMs to reflect on their mistakes and avoid repeating them during inference (An et al., 2023; Li et al., 2024). However, these methods do not account for the student's own errors during the CoT refinement process, neglecting crucial signals that could be used to construct more tailored supervision when distilling reasoning capabilities from the CoTs.

In this paper, we introduce err**O**r-aware self-reflect**ION** (ORION), which constructs more effective supervisions to help distill the reasoning capabilities of a teacher model, such as Deepseek-R1 (DeepSeek-AI et al., 2025), into a student model implemented as SLMs via SFT. To generate more effective supervision signals, ORION leverages the student model itself to refine the long-form CoTs produced by LRMs through self-reflection, thereby narrowing the capability gap and enabling the student to better emulate the teacher's reasoning patterns. Moreover, solution errors made by the student are included in the prompts and considered during the refinement process, guiding the student model to produce more adaptive and accurate long-form CoTs for supervision.

Our experiments show that ORION outperforms all baselines across mathematical tasks of varying difficulty, demonstrating its overall effectiveness. Further analysis indicates that, compared to using long-form CoTs alone as the training target, ORION achieves lower and more stable variance-entropy scores during training, highlighting its improved training stability. Moreover, ORION effectively eliminates redundant reasoning patterns in long-form CoTs, enabling SLMs to learn more concise and higher-quality reasoning processes. Finally, we observe that ORION corrects various types of errors generated by vanilla SLMs, particularly reasoning errors, further validating the effectiveness of its error-aware CoT refinement mechanism.

2 Related Work

Large Language Models (LLMs) have proven effective in solving mathematical problems (Cobbe et al., 2021; Hendrycks et al., 2021). Advanced prompting strategies, such as Chain-of-Thought (CoT) (Wei et al., 2022), decompose the problem-solving process of LLMs into a series of intermediate steps, significantly enhancing their mathematical reasoning abilities (Li et al., 2023; Qin et al., 2023; Luo et al., 2023a). To further enhance the performance of LLMs, particularly for Small Language Models (SLMs), some studies utilize human-annotated solution labels to fine-tune the models (Hsieh et al., 2023). However, this approach often leads to model overfitting to the training labels, limiting the model's ability to learn genuine reasoning knowledge from supervision signals (Luo et al., 2023b; Gudibande et al., 2023).

To construct high-quality SFT datasets, recent studies typically explore long-form reasoning distillation methods (Yang et al., 2025a; Ye et al., 2025). Specifically, these methods treat Reasoning Language Models (RLMs), such as Deepseek-R1 (DeepSeek-AI et al., 2025), as teacher models and use their generated long-form CoTs as supervision targets to fine-tune SLMs, which enables the student model to learn not only the diverse mathematical solution strategies but also the underlying reasoning patterns (Li et al., 2025a; Yang et al., 2025c). However, several insightful studies indicate that due to the capacity and capability limitations of SLMs, these models struggle to effectively learn reasoning strategies from the long-form CoTs generated by more capable teacher models (Li et al., 2025b; Anonymous, 2025), and may even encounter issues such as content repetition and excessive reflection (Li et al., 2025b).

Different from directly distilling long-form reasoning trajectories from teacher models, another research employs superior LLMs to correct and analyze the error reasoning trajectories generated by SLMs, thereby constructing high-quality SFT datasets (Tong et al., 2024; Pan et al., 2025). Specifically, An et al. (2023) propose a reasoning error correction SFT method, where incorrect reasoning generated by SLMs is used as input, and the SLMs are tasked with reproducing the correction results generated by a superior LLM. Tong et al. (2024) utilize multiple teacher models to generate guided learning data for correcting the error solutions generated by the student model, thereby enabling the

student model to learn both correct answers and avoid errors during SFT. However, these methods only optimize the model to avoid repeating errors and do not incorporate the student model's error solutions during the CoT refinement process, making it difficult to construct high-quality supervision signals for distilling reasoning capabilities. In contrast, ORION focuses on enhancing reasoning distillation through error-aware self-reflection, enabling SLMs to reflect on the generated errors and refine long-form reasoning data for SFT.

3 Methodology

In this section, we present Error-Aware Self-Reflection (ORION) in Figure 2, a method designed to enhance the SFT of LLMs for mathematical reasoning. We first describe how to distill knowledge from long-form chain-of-thought (CoT) traces using standard SFT (Sec. 3.1). We then introduce our error-aware data refinement framework (Sec. 3.2), which identifies potential reasoning errors and incorporates them into CoT outputs, thereby enhancing the quality of SFT datasets.

3.1 Distilling Long Reasoning Capability from LLM via Supervised Fine-Tuning

Given a mathematical question q, the LLM is prompted to generate a solution using a task-specific instruction ($\text{Instruct}_{\text{QA}}$). To enhance the reasoning capabilities of LLMs, we consider a Small Language Model (SLM) as the student model $\mathcal{M}_s$ and explore two SFT strategies to optimize its parameters θ: vanilla SFT and reasoning distillation via long-form reasoning traces.

Vanilla SFT Method. In the vanilla SFT paradigm, the model is fine-tuned on human-annotated solution labels $\mathcal{D}_{\text{Raw}} = \{(q^1, L^1), \ldots, (q^n, L^n)\}$ (Wang et al., 2023), which serve as ground-truth outputs for each input question q^i. The training objective minimizes the negative log-likelihood of the ground truth solution L^i:

$$\mathcal{J} = -\sum_{i=1}^{n}\sum_{t=1}^{|L^i|} P(L_t^i | L_{<t}^i, \text{Instruct}_{\text{QA}}(q^i); \theta). \quad (1)$$

While this approach has demonstrated strong empirical performance in enhancing the problem-solving accuracy of LLMs (Hsieh et al., 2023), it often overfits to the training labels, thereby limiting the student model's ability to acquire genuine reasoning knowledge from supervision (Luo et al., 2023b; Gudibande et al., 2023). Consequently, recent research has increasingly focused on curating high-quality datasets specifically designed for SFT (Wettig et al., 2024).

Long-form Reasoning Distillation. Unlike the standard SFT approach, the reasoning distillation method incorporates Chain-of-Thought (CoT) supervision to guide the model through intermediate reasoning steps, rather than directly optimizing for the final label (Hsieh et al., 2023; DeepSeek-AI et al., 2025). This paradigm encourages LLMs to internalize not only the correct solution but also the underlying logical structure and reasoning patterns demonstrated by the teacher model $\mathcal{M}_t$.

Specifically, given a question q, we first regard a Large Reasoning Model $\mathcal{M}_t$, such as DeepSeek-R1 (DeepSeek-AI et al., 2025), as the teacher model and prompt it to generate a Chain-of-Thought (CoT) style response o:

$$o = \mathcal{M}_t(\text{Instruct}_{\text{QA}}(q)). \quad (2)$$

We then construct the SFT dataset $\mathcal{D} = (q^1, o^1), \ldots, (q^n, o^n)$ by pairing each input query q^i with its corresponding CoT-style response o^i generated by the teacher model $\mathcal{M}_t$. The distilled CoT response o^i is used as the supervision target for fine-tuning the student model $\mathcal{M}_s$, enabling it to acquire step-by-step reasoning abilities from the teacher's intermediate reasoning traces:

$$\mathcal{J} = -\sum_{i=1}^{n}\sum_{t=1}^{|o^i|} P(o_t^i | o_{<t}^i, \text{Instruct}_{\text{QA}}(q^i); \theta). \quad (3)$$

This distillation approach encourages the student model to internalize the reasoning behaviors of more capable teacher models, thereby fostering the generation of coherent and logically grounded CoT outputs. However, since the teacher model is unaware of the student's capacity, directly fine-tuning on long-form reasoning traces can be challenging for student models. Naively distilling from the teacher may lead to degenerate behaviors such as repetitive outputs or unstable training dynamics (Yin et al., 2025; Li et al., 2025b). To address these issues, ORION refines long-form CoTs by error-aware self-reflection, providing more tailored and effective training signals for SFT (Sec. 3.2).

3.2 Refining Long-Form Reasoning via Error-Aware Self-Reflection

To enhance the quality of long-form reasoning data for SFT, ORION introduces an error-aware

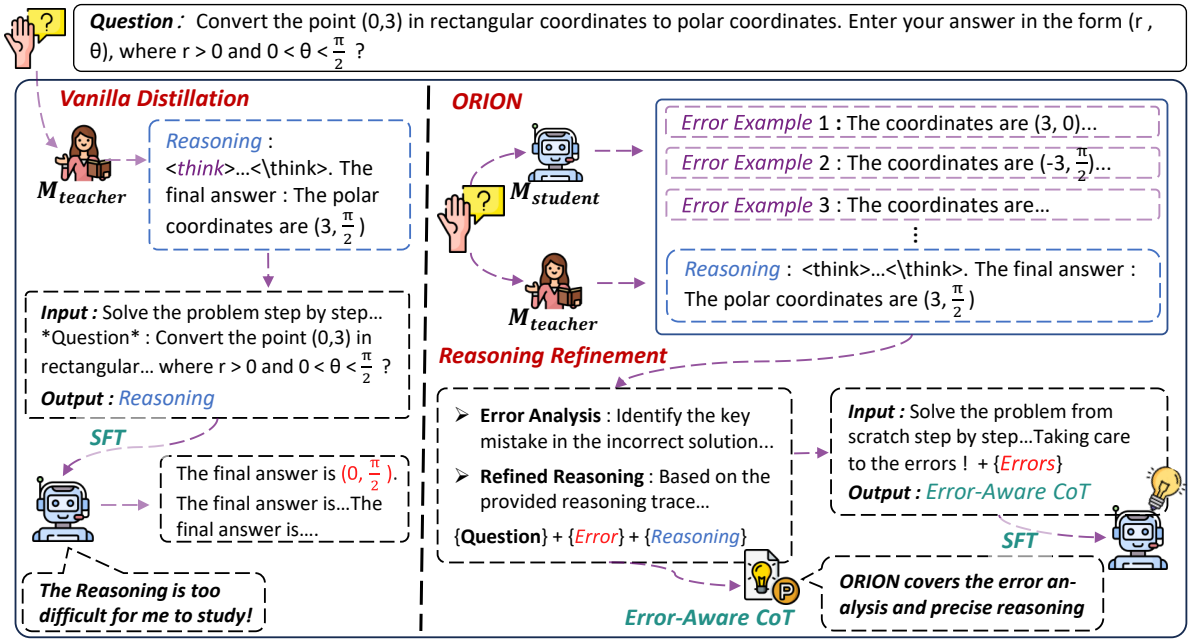


Figure 2: Illustration of Our ORION Model.

reflection mechanism that leverages the student model's own mistakes and self-reflection to refine the teacher-provided long-form CoTs.

Given a dataset $\mathcal{D} = \{(q^1, o^1), \ldots, (q^n, o^n)\}$, where q^i is a mathematical question and o^i is a corresponding CoT-style solution generated by the teacher model $\mathcal{M}_t$, ORION first prompts the student model $\mathcal{M}_s$ to generate multiple potentially incorrect solutions Y^e for each q^i (Sec. 3.2.1). Then, these erroneous outputs expose the model's failure modes and are used to guide a reflection-based refinement process, where $\mathcal{M}_s$ is prompted to revise the original teacher CoT-style response o^i into a more robust solution $\tilde{o}^i$ by explicitly avoiding previously observed errors (Sec. 3.2.2). The resulting refined dataset $\tilde{\mathcal{D}}$ is then used for SFT, as in Eq. 3, to further improve the reasoning capability of the student model $\mathcal{M}_s$.

3.2.1 Error Exposure via Response Sampling

To uncover typical reasoning errors made by the student model $\mathcal{M}_s$, we adopt a response sampling strategy inspired by prior work (An et al., 2023). For each input question q^i, we generate a diverse set of K candidate solutions $Y^i = \{y^i_1, \ldots, y^i_K\}$ by sampling from $\mathcal{M}_s$ under a range of temperatures τ:

$$Y^i \sim \text{Sample}_\tau(\mathcal{M}_s(\text{Instruct}_{\text{QA}}(q^i))). \quad (4)$$

Then each sampled response $y^i_k \in Y^i$ is then post-processed using an answer extraction function $\text{Ans}(\cdot)$, and its correctness is checked against the ground truth label L^i. These responses that produce incorrect final answers are collected into an error set Y^i_{err}:

$$Y^i_{\text{err}} = \{y^i_k \mid \text{Ans}(y^i_k) \neq L^i\}. \quad (5)$$

The erroneous solutions, denoted as $Y^i_{\text{err}} = \{y^i_1, \ldots, y^i_m\}$, serve as references for subsequent reflection, helping the student model identify failure patterns and revise the teacher-generated CoT traces accordingly.

3.2.2 Reasoning Refinement through Error-Aware Self-Reflection

To help the student model $\mathcal{M}_s$ internalize more robust reasoning heuristics during SFT, we introduce a self-reflection mechanism for the student model $\mathcal{M}_s$, which leverages its own erroneous responses to improve the quality of SFT data. Specifically, given an initial long reasoning-based SFT dataset $\mathcal{D} = \{(q^1, o^1), \ldots, (q^n, o^n)\}$, the student model $\mathcal{M}_s$ is encouraged to analyze its own mistakes and learn to refine the original CoT outputs for SFT.

For each mathematical query q^i in $\mathcal{D}$, we begin with an initial long-form CoT-style response o^i and a set of model-generated erroneous attempts $Y^i_{\text{err}} = \{y^i_1, \ldots, y^i_m\}$. For each error example $y^i_k \in Y^i_{\text{err}}$, the student model $\mathcal{M}_s$ is prompted to refine the response o^i generated by the teacher model $\mathcal{M}_t$ by reflecting on these self-made errors:

$$\tilde{o}^i_k = \mathcal{M}_s(\text{Instruct}_{\text{Ref}}(q, y^i_k, o^i)), \quad (6)$$

where $\text{Instruct}_{\text{Ref}}$ denotes a self-reflection instruction that guides the student model $\mathcal{M}_s$ to analyze and refine o^i based on its error response y_k^i. This yields a set of refined reasoning responses for the mathematical query q^i:

$$\tilde{O}^i = \{\tilde{o}_1^i, \ldots, \tilde{o}_m^i\}. \tag{7}$$

To ensure correctness and eliminate spurious refinements, we filter out any $\tilde{o}_k^i$ whose final answer does not match the ground-truth label L^i:

$$\tilde{\mathcal{D}}^i = \{(q^i, \tilde{o}_k^i) \mid 1 \leq k \leq m,\ \text{Ans}(\tilde{o}_k^i) = L^i\}. \tag{8}$$

Finally, we construct the refined SFT dataset $\tilde{\mathcal{D}}$ by aggregating all validated refinements $\tilde{\mathcal{D}}$ across all mathematical queries $\{q^1, \ldots, q^n\}$:

$$\tilde{\mathcal{D}} = \bigcup_{i=1}^{n} \tilde{\mathcal{D}}^i. \tag{9}$$

4 Experimental Methodology

This section first describes the datasets, evaluation metric, and baselines, followed by the implementation details of our experiments.

Dataset. In our experiments, we randomly sample 10,000 examples from the OpenR1-Math-220k dataset (Hugging Face, 2025), which contains high-quality solutions annotated by DeepSeek-R1 (DeepSeek-AI et al., 2025), to construct our training dataset. For evaluation, we select four mathematical problem datasets spanning a range of difficulty levels. For evaluation, we select five mathematical reasoning benchmarks of varying difficulty, including GSM-Hard (Gao et al., 2023), MATH500 (Godahewa et al., 2021), AIME24, and AMC23 (Gao et al., 2024). GSM-Hard is an enhanced version that increases the computational difficulty of the GSM8K dataset (Cobbe et al., 2021). MATH500 is a benchmark of competition math problems of varying difficulty. AIME and AMC test mathematical problem-solving with arithmetic, algebra, counting, geometry, number theory, probability, and other secondary school math topics.

Baselines. We compare our ORION model with both zero-shot models and Supervised Fine-Tuning (SFT) models. We first compare with two zero-shot baselines: vanilla LLM and Wrong-of-Thought (Zhang et al., 2024). For the vanilla LLM, we provide the mathematical question q to the model and ask it to directly generate a solution. Additionally, we compare our approach with Wrong-of-Thought (Zhang et al., 2024) in our experiments. The model first generates an initial answer, then identifies solution errors using a multi-perspective verifier, and regards these errors as signals to help LLMs prevent the repetition of the same mistakes. Furthermore, we utilize two SFT-based LLMs as baselines, including SFT (Label) and SFT (Long-CoT) (DeepSeek-AI et al., 2025). Following previous work (Wang et al., 2024a), SFT (Label) and SFT (Long-CoT) fine-tune the LLMs using human-annotated labels and reasoning traces from RLMs.

Evaluation Metric. Following prior studies (Reich et al., 2023; Zhang et al., 2024), we adopt the Accuracy score as the evaluation metric to determine whether the mathematical QA models correctly generate the ground-truth answers. Specifically, we report Acc@1 and Acc@10, where Acc@1 and Acc@10 represent the average accuracy over one and ten sampled answers at inference time, respectively (Liu et al., 2025).

Implementation Details. For all experiments, we employ Qwen2.5-7b-Instruct (Yang et al., 2025b), Qwen3-8b-Instruct (Yang et al., 2025a) and Llama3.1-8b-Instruct (Grattafiori et al., 2024) as the backbone models for implementing the mathematical tasks. During training, the learning rate is set to 5×10^{-5}, with a gradient accumulation step of 8, and each model is trained for 3 epochs. We also use LoRA (Hu et al., 2022) for efficient training. Additionally, our ORION model is built on TRL[1] and LLaMA Efficient Tuning[2]. We provide additional experimental details in Appendix A.2, and more details of prompt templates are provided in Appendix A.3.

5 Evaluation Result

In this section, we first evaluate the overall performance of ORION through the main experience. Next, we conduct ablation studies to investigate the specific contribution of solution error and self-reflection components. Furthermore, we then analyze the effectiveness of distilled models via ORION. Finally, some case studies are shown in Appendix A.6.

5.1 Overall Performance

As shown in Table 1, we compare the overall performance of ORION with baseline models across diverse mathematical reasoning tasks.

[1] https://github.com/huggingface/trl
[2] https://github.com/hiyouga/LLaMA-Factory

Model	AIME24		AMC23		Math500		GSM-H		Avg.	
	Acc@1	Acc@10	Acc@1	Acc@10	Acc@1	Acc@10	Acc@1	Acc@10	Acc@1	Acc@10
Qwen3-8B-Instruct										
Vanilla LLM	20.00	21.10	55.00	53.10	81.40	80.75	57.40	55.82	53.45	52.69
Wrong-of-Thought	16.67	16.90	55.50	54.85	79.83	80.14	58.16	57.95	52.54	52.46
SFT (Label)	16.67	18.83	52.50	51.55	80.95	79.82	57.13	56.55	51.81	51.69
SFT (Long-CoT)	23.33	22.50	57.50	56.90	82.90	81.40	59.27	58.82	55.75	54.91
ORION	**26.67**	**25.60**	**62.50**	**61.85**	**83.50**	**82.95**	**59.83**	**59.75**	**58.13**	**57.54**
Qwen2.5-7B-Instruct										
Vanilla LLM	10.00	6.20	47.50	44.00	69.80	69.80	55.72	54.34	45.76	43.59
Wrong-of-Thought	3.33	2.79	22.75	24.45	70.50	69.87	**59.39**	**58.91**	38.99	39.01
SFT (Label)	6.67	6.55	45.00	46.25	68.85	69.35	54.31	54.09	43.71	44.06
SFT (Long-CoT)	13.33	11.35	50.00	48.55	72.10	70.21	57.21	55.85	48.16	46.49
ORION	**16.67**	**14.83**	**55.00**	**53.51**	**73.80**	**72.30**	58.86	58.24	**51.08**	**49.72**
Llama3.1-8B-Instruct										
Vanilla LLM	3.33	2.84	22.50	22.50	43.60	41.80	31.23	32.16	25.17	24.83
Wrong-of-thought	0.00	0.00	12.50	9.95	44.00	43.98	31.92	32.97	22.11	21.73
SFT (Label)	3.33	2.28	20.00	21.83	42.80	42.14	30.95	31.18	24.27	24.36
SFT (Long-CoT)	6.67	4.00	25.00	24.50	44.50	43.76	31.83	32.10	27.00	26.09
ORION	**10.00**	**6.80**	**30.00**	**27.00**	**45.70**	**44.78**	**33.34**	**33.95**	**29.76**	**28.13**

Table 1: Overall Performance. The **best** result among all methods is highlighted for clarity and emphasis.

Overall, ORION outperforms all baseline models across datasets, demonstrating its effectiveness. Compared with the prompt-based error-aware learning method Wrong-of-Thought (Zhang et al., 2024), ORION achieves more than a 5% improvement, indicating its superior ability to incorporate self-generated errors into model learning. As the evaluation results show, simply injecting solution errors into prompts during inference may degrade LLM performance. This degradation likely occurs because such errors act as noise that interferes with the reasoning process (Wang et al., 2024b). In contrast to Wrong-of-Thought, ORION provides a more effective mechanism for guiding LLMs to avoid repeating similar solution errors by fine-graining long-form CoT traces as supervision signals and then fine-tuning student models.

Regarding different SFT strategies, compared with the Vanilla model, SFT (Label) yields nearly identical performance, while SFT (Long-CoT) achieves noticeable improvements. This suggests that the reasoning traces derived from the CoT outputs of RLMs, such as DeepSeek-R1 (DeepSeek-AI et al., 2025), provide richer reasoning patterns that can better guide SLMs to imitate during the SFT process. Building upon ORION, the supervision signals are further refined by the student models themselves through self-reflection and the incorporation of self-identified errors, resulting in an additional 2% performance gain. These results further demonstrate the effectiveness of ORION in providing more tailored and effective supervision for student models. Furthermore, to assess generalization, we evaluate ORION on different backbone models, including Qwen2.5-7B, Qwen3-8B, and LLaMA3.1-8B. The results consistently show notable improvements across all backbones, demonstrating the generalization ability of ORION in enhancing mathematical reasoning performance.

5.2 Ablation Study

As shown in Table 2, we conduct an ablation study to isolate the individual effects of the solution error and self-reflection components in ORION. We compare several ablated variants: ORION w/o Solution Error removes the injected erroneous solutions during CoT data refinement, while ORION w/o Self-Reflection directly uses the long-form CoTs as supervisions without refinement and incorporates solution errors into the prompt during training.

Overall, both Self-Reflection and Solution Error are effective in improving the quality of long-form CoT based supervision signals. Specifically, compared with the SFT (Long-CoT) model, ORION w/o Solution Error achieves over a 1% improvement, demonstrating the effectiveness of the self-reflection mechanism. This mechanism enables the student model to refine long-form CoTs by itself, thereby narrowing the gap between the supervision signal and the capability of the student model. Moreover, the ORION w/o Self-Reflection model also outperforms SFT (Long-CoT), indicating that incorporating self-generated solution errors during the SFT process can guide the student model to avoid repeating potential mistakes. Further, comparing ORION w/o Solution Error with the full ORION, we observe additional gains in reasoning performance, suggesting that considering

Model	AIME24		AMC23		Math500		GSM-H		Avg.	
	Acc@1	Acc@10	Acc@1	Acc@10	Acc@1	Acc@10	Acc@1	Acc@10	Acc@1	Acc@10
Qwen3-8B-Instruct										
SFT (Long-CoT)	23.33	22.50	57.50	56.90	82.90	81.40	59.27	58.82	55.75	54.91
ORION	**26.67**	**25.60**	**62.50**	**61.85**	**83.50**	**82.95**	**59.83**	**59.75**	**58.13**	**57.54**
w/o Solution Error	26.67	24.95	60.00	58.25	83.15	82.10	59.27	59.13	57.27	56.11
w/o Self-Reflection	20.00	20.90	57.50	56.50	82.75	81.25	58.86	58.34	54.78	54.25
Qwen2.5-7B-Instruct										
SFT (Long-CoT)	13.33	11.35	50.00	48.55	72.10	70.21	57.21	55.85	48.16	46.49
ORION	**16.67**	**14.83**	**55.00**	**53.51**	**73.80**	72.30	**58.86**	**58.24**	**51.08**	**49.72**
w/o Solution Error	16.67	10.00	52.50	52.90	73.10	**72.90**	58.24	58.20	50.13	48.50
w/o Self-Reflection	10.00	5.83	50.00	47.86	70.20	69.21	57.35	56.55	46.89	44.86
Llama3.1-8B-Instruct										
SFT (Long-CoT)	6.67	4.00	25.00	24.50	44.50	43.76	31.83	32.10	27.00	26.09
ORION	**10.00**	**6.80**	**30.00**	**27.00**	**45.70**	**44.78**	**33.34**	**33.95**	**29.76**	**28.13**
w/o Solution Error	6.67	5.50	30.00	26.50	45.40	44.34	33.13	32.65	28.80	27.25
w/o Self-Reflection	6.67	4.80	27.50	25.50	43.90	43.10	32.91	31.95	27.75	26.34

Table 2: Ablation Study. We evaluate the performance of distilled models trained with different strategies, excluding the Solution Error and Self-Reflection components, to demonstrate the contribution of each component in ORION.

self-made errors during self-reflection brings complementary benefits. These findings highlight that the advantage of incorporating solution errors can be extended to the CoT refinement process, providing higher-quality and more tailored supervision signals for the student model.

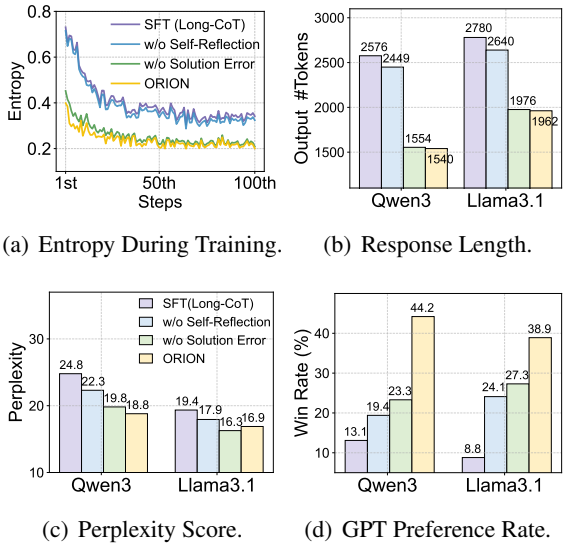
5.3 The Effectiveness of Distilled Models via Different Training Strategies

In this section, we evaluate the training stability and effectiveness of the distilled LLMs, as shown in Figure 3. Three models, SFT (Long-CoT), ORION w/o Solution Error, and ORION w/o Self-Reflection, are used as baselines for comparison.

Training Stability. We further analyze the training stability of different strategies by plotting the entropy scores in Figure 3(a). Compared with both SFT (Long-CoT) and ORION w/o Self-Reflection, the ORION and ORION w/o Solution Error models exhibit significantly lower entropy scores during training, demonstrating their effectiveness in promoting a more stable and learnable training process. This phenomenon further confirms that the self-reflection mechanism benefits the student model's learning process during SFT. Moreover, when incorporating self-generated solution errors, the entropy scores decrease slightly, suggesting that these errors can be effectively utilized to refine long-form CoTs and better align them with the learning capacity of student models. Overall, these results highlight the crucial role of narrowing the gap between teacher-provided supervision and the learning ability of student models in SFT-based distillation.

Effectiveness of Distilled Models. To evaluate the distilled models trained with different strategies, we analyze both the length (Figure 3(b)) and quality (Figures 3(c) and 3(d)) of the generated CoTs.

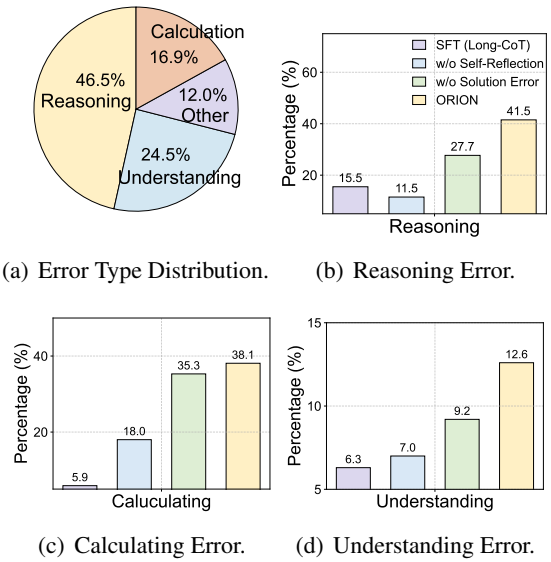
As shown in Figure 3(b), we prompt each distilled model to generate reasoning trajectories and measure their average lengths. The results indicate that both ORION and ORION w/o Solution Error produce significantly shorter reasoning trajectories compared with other models. This finding suggests that the self-reflection mechanism helps eliminate rigid or redundant reasoning patterns, enabling the student model to learn more concise and efficient reasoning processes. Consequently, it helps mitigate the overthinking phenomenon (Chen et al., 2024) and reduces inference costs. Next, we assess the quality of the generated CoTs. As shown in Figure 3(c), we collect the generated reasoning traces and use the corresponding LLMs as judges to compute the perplexity scores with respect to the ground-truth answers. The results show that ORION achieves the lowest perplexity among all models, demonstrating that its generated CoTs better align with the true reasoning process and can effectively guide zero-shot LLMs toward correct answers. Furthermore, as shown in Figure 3(d), we employ GPT-4 as an additional evaluator to assess the generated responses based on criteria such as correctness and clarity. Specifically, GPT-4 is prompted to select the best responses among all four models using the prompt templates provided in Appendix A.4. The evaluation results demonstrate that ORION achieves twice the win rate of other models, highlighting its effectiveness in helping to distill more capable and reliable reasoning models.



(a) Entropy During Training. (b) Response Length.

(c) Perplexity Score. (d) GPT Preference Rate.

Figure 3: Performance of Distilled Models Optimized with Different Training Strategies. We first report the entropy scores during distillation under various strategies (Figure 3(a)). We then present the response lengths generated by the distilled models (Figure 3(b)). Finally, we evaluate the quality of CoT and final responses using both Vanilla LLMs and GPT-4 as judges (Figure 3(c) and Figure 3(d), respectively).



(a) Error Type Distribution. (b) Reasoning Error.

(c) Calculating Error. (d) Understanding Error.

Figure 4: Analysis of Distilled Models on Different Error Types. All experiments are based on the Qwen3-8B model. Figure 4(a) shows the distribution of distinct error types encountered by vanilla LLMs, while Figures 4(b), 4(c), and 4(d) present the corresponding correction rates for each error type.

5.4 The Performance of Distilled Models on Different Error Types

To further validate the effectiveness of our targeted error exposure strategy, we conduct a detailed analysis of error types and their corresponding correction outcomes, as illustrated in Figure 4. Specifically, we first prompt the vanilla LLM to generate responses on the entire test set and sample 500 instances that are incorrectly answered for further analysis. These sampled instances are then categorized into four distinct error types, reasoning, calculation, and understanding errors, using the prompt template provided in Appendix A.5.

As shown in Figure 4(a), we present the distribution of error categories produced by the vanilla LLM. The analysis reveals that reasoning errors constitute the majority of all mistakes, indicating that vanilla LLMs struggle most with reasoning-intensive problems. This observation underscores the importance of enhancing the reasoning ability of student models in mathematical problem-solving, such as distilling reasoning capabilities from LRMs. We then evaluate the effectiveness of different distilled models in correcting each error type, as shown in Figures 4(b), 4(c), and 4(d). For reasoning errors, ORION achieves the highest correction rate of approximately 41%, demonstrating its strong ability to help the student model overcome the most common error type. This improvement can be attributed to the refined supervision signals provided by the self-reflection mechanism, which guide the model to internalize effective reasoning patterns. Furthermore, compared with ORION w/o Solution Error, ORION achieves significantly higher correction rates on both reasoning and understanding errors, indicating that the inclusion of self-generated solution errors further enhances the model's ability to refine supervision signals and avoid repeating similar reasoning or comprehension mistakes.

6 Conclusion

This paper proposes the Error-Aware Self-Reflection (err**O**r-aware self-**R**eflect**ION**, ORION) method, which distills the strong reasoning capabilities of teacher models into a student model implemented as SLMs via SFT. ORION leverages the student model's own self-reflection to improve the long-form CoTs generated by Long Reasoning Models (LRMs), thereby generating more effective supervision signals. Experimental results show that ORION outperforms all baselines across multiple mathematical tasks.

Limitation

Although ORION has proven effective in constructing higher-quality SFT data, it still relies on powerful closed-source reasoning language models to generate long-form reasoning chains, which incur significant computational costs. In addition, ORION only utilizes long-form CoTs from Deepseek-R1 as the training target and has not explored the effectiveness of using other closed-source long-form reasoning models as teacher models, primarily due to the high time and computational costs involved in generating reasoning data with closed-source models.

References

Shengnan An, Zexiong Ma, Zeqi Lin, Nanning Zheng, Jian-Guang Lou, and Weizhu Chen. 2023. Learning from mistakes makes llm better reasoner. *ArXiv preprint*, abs/2310.20689.

Anonymous. 2025. MORALE: Segment-guided distillation framework for small reasoning models. In *Submitted to ACL Rolling Review - May 2025*. Under review.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, and Amanda Askell. 2020. Language models are few-shot learners. In *Proceedings of the NeurIPS*.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, et al. 2024. Do not think that much for 2+ 3=? on the overthinking of o1-like llms. *Arxiv prepint*, abs/2412.21187.

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. *ArXiv preprint*, abs/2110.14168.

DeepSeek-AI, Daya Guo, Dejian Yang, and Haowei Zhang. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Arxiv preprint*, abs/2501.12948.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. 2023. Specializing smaller language models towards multi-step reasoning. In *Proceedings of the ICML*, pages 10421–10430.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, and et al. Yibo Miao. 2024. Omni-math: A universal olympiad level mathematic benchmark for large language models. *arXiv preprint arXiv:2410.07985*.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. PAL: program-aided language models. In *Proceedings of the ICML*, pages 10764–10799.

Rakshitha Godahewa, Christoph Bergmeir, Geoffrey I. Webb, Rob J. Hyndman, and Pablo Montero-Manso. 2021. Monash time series forecasting archive. *arXiv preprint arXiv:2105.06643*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, and Abhishek Kadian. 2024. The llama 3 herd of models. *Arxiv preprint*, abs/2407.21783.

Arnav Gudibande, Eric Wallace, Charlie Snell, Xinyang Geng, Hao Liu, Pieter Abbeel, Sergey Levine, and Dawn Song. 2023. The false promise of imitating proprietary llms. *ArXiv preprint*, abs/2305.15717.

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. *ArXiv preprint*, abs/2103.03874.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8003–8017, Toronto, Canada. Association for Computational Linguistics.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *Proceedings of ICLR*.

Hugging Face. 2025. Open r1: A fully open reproduction of deepseek-r1.

Chengpeng Li, Zheng Yuan, Guanting Dong, Keming Lu, Jiancan Wu, Chuanqi Tan, Xiang Wang, and Chang Zhou. 2023. Query and response augmentation cannot help out-of-domain math reasoning generalization. *ArXiv preprint*, abs/2310.05506.

Dacheng Li, Shiyi Cao, Tyler Griggs, Shu Liu, Xiangxi Mo, Eric Tang, Sumanth Hegde, Kourosh Hakhamaneshi, Shishir G Patil, Matei Zaharia, et al. 2025a. Llms can easily learn to reason from demonstrations structure, not content, is what matters! *arXiv preprint arXiv:2502.07374*.

Yuetai Li, Xiang Yue, Zhangchen Xu, Fengqing Jiang, Luyao Niu, Bill Yuchen Lin, Bhaskar Ramasubramanian, and Radha Poovendran. 2025b. Small models struggle to learn from strong reasoners. *arXiv preprint arXiv:2502.12143*.

Zhuochun Li, Yuelyu Ji, Rui Meng, and Daqing He. 2024. Learning from committee: Reasoning distillation from a mixture of teachers with peer-review. *arXiv preprint arXiv:2410.03663*.

Junnan Liu, Hongwei Liu, Linchen Xiao, Ziyi Wang, Kuikun Liu, Songyang Gao, Wenwei Zhang, Songyang Zhang, and Kai Chen. 2025. Are your llms capable of stable reasoning? *arXiv preprint arXiv:2412.13147*.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023a. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *ArXiv preprint*, abs/2308.09583.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. 2023b. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *ArXiv preprint*, abs/2308.08747.

OpenAI, Josh Achiam, Steven Adler, and Sandhini Agarwal. 2023. Gpt-4 technical report.

Zhuoshi Pan, Yu Li, Honglin Lin, Qizhi Pei, Zinan Tang, Wei Wu, Chenlin Ming, H Vicky Zhao, Conghui He, and Lijun Wu. 2025. Lemma: Learning from errors for mathematical advancement in llms. *arXiv preprint arXiv:2503.17439*.

Libo Qin, Qiguang Chen, Fuxuan Wei, Shijue Huang, and Wanxiang Che. 2023. Cross-lingual prompting: Improving zero-shot chain-of-thought reasoning across languages. In *Proceedings of EMNLP*, pages 2695–2709.

Taly Reich, Alex Kaju, and Sam J Maglio. 2023. How to overcome algorithm aversion: Learning from mistakes. *Journal of Consumer Psychology*, 33(2):285–302.

Yongqi Tong, Dawei Li, Sizhe Wang, Yujia Wang, Fei Teng, and Jingbo Shang. 2024. Can llms learn from previous mistakes? investigating llms' errors to boost for reasoning. *ArXiv preprint*, abs/2403.20046.

Peiyi Wang, Lei Li, Liang Chen, Feifan Song, Binghuai Lin, Yunbo Cao, Tianyu Liu, and Zhifang Sui. 2023. Making large language models better reasoners with alignment. *ArXiv preprint*, abs/2309.02144.

Renxi Wang, Haonan Li, Xudong Han, Yixuan Zhang, and Timothy Baldwin. 2024a. Learning from failure: Integrating negative examples when fine-tuning large language models as agents. *ArXiv preprint*, abs/2402.11651.

Yanbo Wang, Yongcan Yu, Jian Liang, and Ran He. 2025. A comprehensive survey on trustworthiness in reasoning with large language models. *arXiv preprint arXiv:2509.03871*.

Zihao Wang, Anji Liu, Haowei Lin, Jiaqi Li, Xiaojian Ma, and Yitao Liang. 2024b. Rat: Retrieval augmented thoughts elicit context-aware reasoning in long-horizon generation. *arXiv preprint arXiv:2403.05313*.

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 *Proceedings of NeurIPS*.

Alexander Wettig, Aatmik Gupta, Saumya Malik, and Danqi Chen. 2024. Qurating: Selecting high-quality data for training language models. *ArXiv preprint*, abs/2402.09739.

An Yang, Anfeng Li, Baosong Yang, and et al. Beichen Zhang. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

An Yang, Baosong Yang, Beichen Zhang, and Binyuan Hui. 2025b. Qwen2.5 technical report. *Arxiv preprint*, abs/2412.15115.

Ling Yang, Zhaochen Yu, Tianjun Zhang, Minkai Xu, Joseph E. Gonzalez, Bin Cui, and Shuicheng Yan. 2025c. Supercorrect: Advancing small LLM reasoning with thought template distillation and self-correction. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. 2025. Limo: Less is more for reasoning. *arXiv preprint arXiv:2502.03387*.

Huifeng Yin, Yu Zhao, Minghao Wu, Xuanfan Ni, Bo Zeng, Hao Wang, Tianqi Shi, Liangying Shao, Chenyang Lyu, Longyue Wang, Weihua Luo, and Kaifu Zhang. 2025. Marco-o1 v2: Towards widening the distillation bottleneck for reasoning models. *arXiv preprint arXiv:2503.01461*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, and Xi Victoria Lin. 2022. Opt: Open pre-trained transformer language models. *ArXiv preprint*, abs/2205.01068.

Yongheng Zhang, Qiguang Chen, Jingxuan Zhou, Peng Wang, Jiasheng Si, Jin Wang, Wenpeng Lu, and Libo Qin. 2024. Wrong-of-thought: An integrated reasoning framework with multi-perspective verification and wrong information. In *Proceedings of EMNLP Findings*, pages 6644–6653.

Wangchunshu Zhou, Canwen Xu, and Julian McAuley. 2022. Bert learns to teach: Knowledge distillation with meta learning. *arXiv preprint arXiv:2106.04570*.

Dataset	Train	Test
OpenR1-Math-220k (2025)	10,000	-
GSM-Hard (2023)	-	1,319
MATH500 (2021)	-	500
AIME24 (2024)	-	30
AMC23 (2024)	-	40

Table 3: Data Statistics.

A Appendix

A.1 License

We provide the licenses for the datasets we use. OpenR1-Math-220k and MATH500 are licensed under the Apache License 2.0. AIME24 and AMC23 are not currently labeled with license types. GSM-Hard uses the MIT license. All of these licenses and agreements allow their data for academic use.

A.2 Additional Implementation Details

In this section, we provide the detailed data statistics in ORION. We build our training set by using 10,000 examples randomly sampled from OpenR1-Math-220k (Hugging Face, 2025) as seed data and then expand it to form our final training data. For evaluation, we use the GSM-Hard dataset containing 1,319 examples, MATH500 with 500 examples, AIME24 with 30 examples, and AMC23 with 40 examples. All statistics are shown in Table 3.

A.3 Prompt Templates Used in ORION

We detail the prompts employed across different stages of ORION. As shown in Figure 5, the Error Exposure stage uses prompts that instruct the model to solve problems, from which incorrect cases are collected. These errors are then utilized in the Self-Reflection stage, where the prompt templates in Figure 6 instruct the model to identify and correct solution errors while refining the long-form CoTs. Finally, during the Supervised Fine-Tuning stage, the prompt in Figure 7 leverages the refined trajectories with their corresponding solution errors as training supervision.

A.4 Prompt Templates Used for Evaluating the Quality of ORION

As shown in Figure 8, we present the prompt templates used to evaluate the quality and educational value of the CoTs synthesized by four methods: SFT (Long-CoT), ORION w/o Solution Error, ORION w/o Self-Reflection, and ORION. The evaluation is conducted using GPT-4 (OpenAI et al., 2023) as the evaluator, which assesses each generated reasoning trace based on its clarity, correctness, and pedagogical usefulness. We employ consistent prompt templates across all methods to ensure fair comparison and reproducibility.

A.5 Prompt Templates Used for Solution Error Categorization

As shown in Figure 9, we present the prompt templates used to classify the solution errors generated by GPT-4 (OpenAI et al., 2023). The prompt instructs the evaluator model to classify each incorrect response into one of four predefined categories: Reasoning Error, Understanding Error, Calculation Error, or Other.

A.6 Case Study

In this section, we present a detailed case study across two tables to intuitively demonstrate the core process and the resulting effectiveness of ORION. As shown in Table 4, we illustrate the internal mechanism of our method. It presents the error exposure and self-reflection process. Building on this, Table 5 contrasts the final response generated by the ORION with those from baseline methods, showing the superior quality and clarity of its output.

First, we examine a relatively simple math problem to analyze the behavior of ORION during the error exposure phase. Despite the low comlexity of the problem, sampling multiple responses reveals a diverse range of errors, including a hallucination for general concepts ("lower class soldiers (8) is more than the available (10)"), misunderstandings of the problem statement ("exceeding their respective limits") and basic computational mistakes ("C(8,10)=90"). Therefore, we argue that it is crucial to expose as many diverse erroneous responses as possible for each problem and to use them as guidance to avoid similar pitfalls in the final reasoning. Then, we present a complete example of the Self-Reflection process. Given a mathematical problem, an incorrect solution, and a standard long-form CoT from an RLM, the model generates an Error-Aware Chain-of-Thought that not only analyzes the key mistakes in the solution error but also refines the standard reasoning trace to make it better aligned with the model's own capacity.

As shown in Table 5, we contrast the final responses to illustrate the superior reasoning capability of the full ORION model. The ORION w/o Self-Reflection provides a direct solution but lacks any analysis of potential pitfalls, resulting in an

incorrect response. The ORION w/o Solution Error produces an analysis of possible error, however, due to the absence of error examples in its training, it ultimately fails to deliver a correct answer. In stark contrast, the complete ORION model exhibits self-awareness by first identifying a potential failure point, noting that the solution could be flawed ("because the expression for n was not properly simplified or checked for integer solutions."). Building on this crucial analysis, it then presents a clear, refined solution, demonstrating its ability to anticipate and preempt common errors.

Instruction for Error Exposure

Please reason step by step, and put your final answer within boxed{}.

Problem: {Problem}

Figure 5: The Prompt Templates Used for Error Exposure.

Instruction for Reasoning Refinement

You are given a math problem, an incorrect solution, and a long reasoning trace. Your task has two steps:

1. *Error Analysis*
- Identify the key mistake in the incorrect solution.
- Explain briefly why this mistake is wrong.

2. *Refined Reasoning*
- Based on the provided long reasoning trace, extract only the *essential and correct steps*.
- Keep the reasoning clear, concise, and logically coherent.

Problem:: {Problem}

Incorrect Solution:: {Error}

Long Reasoning Trace (to refine):: {Long_Reasoning_trace}

Output Format (strict)

Error Analysis
- [Briefly explain the key error]

Correct Reasoning
[Step-by-step refined reasoning, clear and concise, avoiding the previous error]
Final Answer: \\boxed{{...}}

Figure 6: The Prompt Templates Used for Reasoning Refinement.

Instruction for Supervised Fine-Tuning

Instruct: Please reason step by step, and put your final answer within boxed{}.

Input: *Problem*: {Problem} + *Incorrect Solution*: {Error}

Output: {Error-Aware Chain-of-Thought}

Figure 7: The Prompt Templates Used for Supervised Fine-Tuning.

Instruction for GPT-Based Score

You are given three different reasoning chains that attempt to solve the same math problem. Your task is to evaluate these four reasoning chains based on the following three criteria and chose the best reasoning chain:

1. *Correctness*: Whether the reasoning is logically valid and leads to the correct final answer.
2. *Clarity*: Whether the reasoning is clear, well-structured, and easy to follow.
3. *Completeness*: Whether the reasoning covers all necessary steps and avoids unjustified leaps.

Only output the final choice without explanations, in the following format:

Best Reasoning Chain: {A/B/C/D}

Figure 8: The Prompt Templates Used in GPT Preference Rate.

Instruction for Error Type Categorization

You are given a mathematical problem, and an incorrect solution process. Your task is to analyze the incorrect solution process and categorize the type of error made. The possible error types are:

1. *Understanding Error* – The model misunderstood the problem or misinterpreted the instructions.
2. *Reasoning Error* – The model made an error in logical reasoning or approach.
3. *Calculation* – Arithmetic mistakes or errors in calculations.
4. *Others* – Any other type of mistake that doesn't fall into the categories above.

Based on the analysis, output only the most appropriate error type. Do not include any additional explanations or content.

Problem: {problem}
Incorrect Solution: {Incorrect Solution}

Figure 9: The Prompt Templates Used for Error Type Categorization.

Case 1: The Case of Error Exposure via Response Sampling.
Question: The Greek army contained two types of soldiers: the upper class and the lower class soldiers. If there were a total of 5 upper class soldiers, and 10 lower class soldiers in a certain part of Athens, and the battle of Thermopylae demands a force of 4 upper class soldiers and 8 lower class soldiers, how many different battalions can be sent? **Ground-Truth:** 225 **Error Response[0]:** To solve this problem, we need to determine how many different battalions can be formed given the constraints on..., Since the requirement for **lower class soldiers (8) is more than the available (10)**, it is impossible to...**So the final answer is boxed{0}.** **Error Response[1]:** To solve this problem, we need to determine how many different battalions can be formed given the constraints on..., Since both the upper class and lower class soldiers allow us to form exactly one battalion without ...**exceeding their respective limits...So the final answer is boxed{1}.** **Error Response[2]:** To solve this problem, we need to determine how many different battalions can be formed given the constraints on the number of upper class and lower class soldiers available... For lower class soldiers, the number of ways to choose 8 from 10 is mistakenly calculated as **C(10,8)=90**, Then, the total number of battalions is $5 \times 90 = 450$. **So the final answer is boxed{450}.**
Case 2: The Case of Reasoning Refinement via Error-Aware Self-Reflection.
Question: Let \$a, b, c\$ be distinct numbers such that the equations $x^2 + ax + 1 = 0$ and $x^2 + bx + c = 0$ have a common real root, and the equations $x^2 + x + a = 0$ and $x^2 + cx + b = 0$ also have a common real root. Compute the sum $a + b + c$. **Ground-Truth:** -3 **Error Response:** To solve for $a + b + c$, we need to analyze the given conditions step by step. Let the common real root of the equations $x^2 + ax + 1 = 0$ and $x^2 + bx + c = 0$ be r; substituting r into both equations gives $r^2 + ar + 1 = 0$ (1) and $r^2 + br + c = 0$ (2). Subtracting (1) from (2) yields $(b - a)r + (c - 1) = 0$, so $r = \frac{1-c}{b-a}$ (3). Similarly, let the common real root of $x^2 + x + a = 0$ and $x^2 + cx + b = 0$ be s; substituting s gives $s^2 + s + a = 0$ (4) and $s^2 + cs + b = 0$ (5). Subtracting (4) from (5) yields $(c - 1)s + (b - a) = 0$, so $s = \frac{a-b}{c-1}$ (6). Since $r = s$, equating (3) and (6) gives $\frac{1-c}{b-a} = \frac{a-b}{c-1}$; cross-multiplying gives $(1 - c)(c - 1) = (b - a)^2$, or $-(c - 1)^2 = (b - a)^2$. Since a, b, c are distinct, $(b - a)^2 \neq 0$, so $(c - 1)^2 = (b - a)^2$, meaning $c - 1 = \pm(b - a)$. Considering $c = b - a + 1$ and substituting back, we find $r = -1$ from (3). Substituting $r = -1$ into $x^2 + x + a = 0$ gives $1 - 1 + a = 0$, so $a = 0$. Then $c = b + 1$; checking consistency with $x^2 + cx + b = 0$ confirms validity, giving $b = -1$ and $c = 0$. **Thus, the final answer is boxed{-1}.** **Reasoning Refinement:** *Error Analysis* : The incorrect solution **incorrectly assumed that the common roots α and β must be the same.** leading to a contradiction with the requirement that a, b, c are distinct. Specifically, it derived $\beta = \frac{1-c}{a-b}$ and $\alpha = \frac{c-1}{b-a}$, then concluded $\alpha = \beta$, leading to $\beta = 1$, $\alpha = 1$, $a = -2$, and $b = -1 - c$. This made a, b dependent on c, conflicting with the distinctness condition. *Correct Reasoning* : Let's start by carefully defining the problem to avoid confusion. We need to find $a + b + c$ where a, b, c are distinct numbers, with two pairs of quadratic equations sharing common real roots. First, let's clarify which roots are common—they don't have to be the same across both pairs, **which was a key mistake in the incorrect solution.** First, define the common roots properly: Let the common real root of $x^2 + ax + 1 = 0$ and $x^2 + bx + c = 0$ be α; this means α satisfies both equations. Separately, let the common real root of $x^2 + x + a = 0$ and $x^2 + cx + b = 0$ be β, so β satisfies these two equations. Note that α and β might not be equal—this distinction is crucial. Next, use Vieta's Formulas to relate the roots and coefficients of each quadratic equation. For a quadratic equation $x^2 + px + q = 0$, the sum of roots is $-p$ and the product is q... Now, let's connect these equations using the fact that α and β satisfy their respective pairs of equations. For α in both quadratics, subtracting the equations gives $(b - a)\alpha + (c - 1) = 0 \implies \alpha = \frac{c-1}{b-a}$. For β, subtracting its pair gives $(1 - c)\beta + (a - b) = 0 \implies \beta = \frac{b-a}{c-1}$. Notice that $\alpha \cdot \beta = 1$—interesting, but let's focus on α. Let's simplify by testing specific values for α. Suppose $\alpha = 1$: substituting into $x^2 + ax + 1 = 0$ gives $1 + a + 1 = 0 \implies a = -2$. For $x^2 + bx + c = 0$, $1 + b + c = 0 \implies b = -1 - c$. Now check the second pair with $a = -2$: $x^2 + x - 2 = 0$ factors to $(x + 2)(x - 1) = 0$, so β is 1 or -2. If $\beta = 1$, substituting into $x^2 + cx + b = 0$ gives $1 + c + b = 0$, which holds since $b = -1 - c$. If $\beta = -2$, we get $c = 1$ and $b = -2$, but this makes $a = b$, violating distinctness—so $\beta = 1$ is valid. Ensuring a, b, c are distinct (e.g., $c \neq 1, -2$), the sum is $a + b + c = -2 + (-1 - c) + c = -3$. **Thus, the final answer is boxed{-3}.**

Table 4: Case Study. Text in **red** indicates errors and their causes in the response, while text in **blue** marks key errors identified by ORION and the corresponding correct solutions.

Case 3: Comparison of Responses from Different Baseline Methods.
Question: Please reason step by step, and put your final answer within boxed{}. If m and n are integers such that $3m + 4n = 100$, what is the smallest possible value of $\lvert m - n \rvert$?
Ground-Truth: 3
Vanilla Response: To find the smallest possible value of $\lvert m - n \rvert$ given $3m + 4n = 100$, we solve for m in terms of n: $m = \frac{100-4n}{3}$. For m to be an integer, $100 - 4n$ must be divisible by 3. Since $100 \equiv 1 \pmod 3$ and $-4n \equiv -n \pmod 3$, **we require $n \equiv 0 \pmod 3$.** Let $n = 3k$. Substituting gives $m = \frac{100-12k}{3} = 33 - 4k$, so both m and n are integers: $m = 33-4k, n = 3k$. Then $\lvert m-n \rvert = \lvert(33-4k)-3k\rvert = \lvert 33-7k \rvert$. Minimizing $\lvert 33 - 7k \rvert$ reduces to choosing k nearest to $33/7 \approx 4.714$. For $k = 4$, $(m, n) = (17, 12)$ and $\lvert m - n \rvert = 5$. For $k = 5$, $(m, n) = (13, 15)$ and $\lvert m - n \rvert = 2$. **Hence, the smallest possible value is boxed {2}.**
ORION w/o Self-Reflection: Hmm, okay, let's try to express n in terms of m: $n = \frac{100-3m}{4}$. For n to be an integer, $100 - 3m$ must be divisible by 4. Since $100 \equiv 0 \pmod 4$, we need $3m \equiv 0 \pmod 4$, and because 3 and 4 are coprime, m must be a multiple of 4. Let's set $m = 4k$, giving $n = 25 - 3k$. Then $\lvert m - n \rvert = \lvert 4k - (25 - 3k)\rvert = \lvert 7k - 25 \rvert$... To minimize this, k should be near $25/7 \approx 3.57$. Checking $k = 3$ gives $(m, n) = (12, 16)$ and $\lvert m - n \rvert = 4$; checking $k = 4$ gives $(m, n) = (16, 13)$ and $\lvert m - n \rvert = 3$. **Both are small, but the smaller one seems to be 4 when $k = 3$... Thus, the final answer is boxed{4}.**
ORION w/o Solution Error: ## Error Analysis: The main mistake in the original solution was incorrectly simplifying the equation $3m + 4n = 100$ when attempting to express n in terms of m, leading to non-integer values for m and n. ## Correct Reasoning: We want to minimize $\lvert m - n \rvert$ given $3m + 4n = 100$. Solving for m, $m = \frac{100-4n}{3}$. Suppose n is chosen to make m an integer. Trying $n = 3k$, we get $m = \frac{100-12k}{3} = 33 - 4k$. Approximating, we take $m \approx 33 - 4k$. Then $\lvert m - n \rvert \approx \lvert 33 - 7k \rvert$. **If $k = 5$, this gives $m = 13, n = 15$, so $\lvert m - n \rvert = 2$. Thus, the smallest possible value is boxed {2}.**
ORION: ## Error Analysis: The possible mistake was the incorrect handling of $3m + 4n = 100$ when solving for m, n. The substitution and simplification produced a wrong expression for n, leading to non-integer values. The error arose **because the expression for n was not properly simplified or checked for integer solutions.** ## Correct Reasoning: We aim to minimize $\lvert m - n \rvert$ given $3m + 4n = 100$. Expressing n in terms of m, $n = \frac{100-3m}{4}$. For n to be integer, $100 - 3m$ must be divisible by 4. Since $100 \equiv 0 \pmod 4$, this requires $3m \equiv 0 \pmod 4$, so $m = 4k$. Substituting gives $n = 25 - 3k, \quad m = 4k$. For non-negative integers, $0 \leq k \leq 8$. Then $\lvert m - n \rvert = \lvert 4k - (25 - 3k)\rvert = \lvert 7k - 25 \rvert$. Evaluating: $k = 3 \Rightarrow \lvert 21 - 25 \rvert = 4$, $k = 4 \Rightarrow \lvert 28 - 25 \rvert = 3$, others yield larger values. **Thus the minimum is boxed {3}.**

Table 5: Case Study. Text in **red** indicates errors and their causes in the response, while text in **blue** marks key errors identified by ORION and the corresponding correct solutions.