ReActR: Reasoning through Error-Activated Reflection for LLM Post-Training

Lina Sun
Shanghai University of Engineering Science, Shanghai, China
{m320124412}@sues.edu.cn

Abstract

Although Large Language Models (LLMs) have demonstrated substantial proficiency in reasoning, current approaches focus disproportionately on scaling correct training samples, underexploring the value of incorrect reasoning trajectories. Motivated by how humans learn from mistakes, we propose **ReActR** (**R**easoning through **E**rror-**Act**ivated **R**eflection), a framework that enhances reasoning by learning reflective behaviors from erroneous trajectories. Specifically, ReActR comprises data construction and training. First, we synthesize multi-turn erroneous reasoning dataset spanning diverse error types and difficult levels via self-generation and targeted error generation. Second, we enhance the model's capabilities through Supervised Fine-Tuning (SFT) on synthesized data and then apply Group Relative Policy Optimization (GRPO) with multiple reward signals to further refine reasoning performance. Extensive experiments across five benchmarks and three LLMs demonstrate that ReActR effectively enhances reasoning performance. Notably, on Llama-3-8B, ReActR achieves an average improvement of 3.5% across the five datasets.

1 Introduction

In recent years, LLMs have made remarkable progress in reasoning and have become widely adopted across diverse problems (Wei et al., 2022). However, they still encounter significant challenges when it comes to mathematical reasoning tasks (Ahn et al., 2024; Yang et al., 2024; Ying et al., 2024), which require complex and rigorous logical reasoning capabilities. Existing methods for improving the mathematical abilities of LLMs mainly follow three directions: (1) distilling high quality reasoning chains from stronger expert models (Magister et al., 2023); (2) synthesizing or augmenting mathematical datasets to increase sample diversity (Luo et al., 2023; Yu et al., 2023); (3) employing self-reflection and correction to refine reasoning (Zhang et al., 2024; Pan et al., 2025). Despite their success, a common limitation is that these methods predominantly train on fully correct reasoning trajectories, thereby overlooking the rich learning signals contained in erroneous ones.

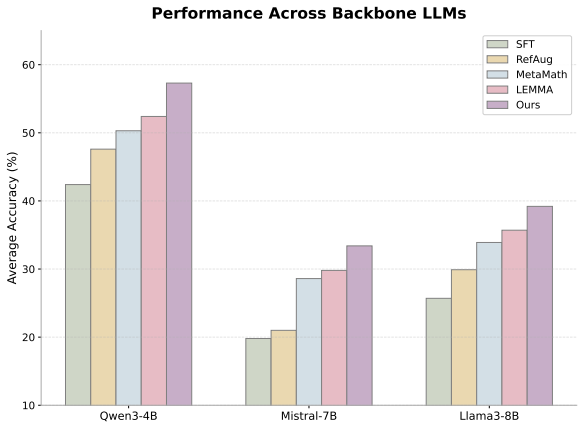

Figure 1: Performance comparison across different backbone LLMs on mathematical benchmarks.

Erroneous trajectories are not merely noise. They not only expose the model's reasoning bias, such as calculation mistakes or formula misuse but also provide valuable supervision for error detection and correction. Learning from such failures is also consistent with human problem-solving, where mistakes often serve as triggers for deeper research. Ignoring erroneous trajectories limits a model's ability to recognize and repair its own reasoning failures.

Recent studies have started to explore the use of erroneous data (Zou et al., 2025; Li et al., 2025) to improve mathematical reasoning in LLMs. For instance, some works constructs incorrect–correct pairs for fine-tuning (Lee et al., 2024), enabling models to iteratively revise their answers. Another works synthesizes common error types based on predefined rules and trains models to reflect on and

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 43017–43028

correct them (Pan et al., 2025). However, these approaches still face limitations: the former are often collected with limited scale and coverage, resulting in insufficient diversity of error types, The latter provide supervised training but lack reinforcement optimization to further strengthen the models' behaviors.

To address these limitations, we propose ReActR (Reasoning through Error-Activated Reflection), a framework that learns reflective and corrective behaviors from error–correction trajectories, and further reinforces them via Group Relative Policy Optimization (GRPO). ReActR has two primary aspects: (1) We generate diverse erroneous trajectories across difficulty levels and error types through a combination of self-generation and targeted augmentation, and construct a multi-turn reflection–correction dataset that captures the full reasoning loop (Problem → Attempt → Reflection → Correction/Confirmation → Summary). (2) We perform supervised fine-tuning (SFT) on the constructed dataset to establish self-reflective and correction behaviors, and then apply GRPO with fine-grained reward signals for reflection and correction accuracy, enabling enhancement of both abilities and finally improve the mathematical reasoning.

Experiments on multiple mathematical benchmarks demonstrate that ReActR achieves strong improvements over competitive baselines. In particular, ReActR boosts the performance of small-scale models on in-domain benchmarks and exhibits strong generalization to out-of-domain evaluations. Further analysis shows that reflection–correction data is crucial for establishing basic reflective and corrective behaviors, while GRPO provides complementary reinforcement that further strengthens both these abilities, enabling strong gains even with relatively limited training data.

2 Related Work

Data Augmentation for Math Enhancing mathematical data has become a pivotal approach for improving the mathematical reasoning capabilities of LLMs. Existing research primarily revolves around two dimensions: improving data quality and expanding data scale. In terms of improving data quality, one mainstream line of work focuses on optimizing the reasoning process. For instance, by performing knowledge distillation from more powerful expert models (Yue et al., 2023; Yu et al., 2024), researchers obtain solutions that are logically rigorous and highly accurate, thereby enhancing the quality of the training data. Regarding the expansion of data scale, researchers have proposed various data synthesis methods. Such work either augments existing problems and solutions (Li et al., 2024b; Zhang et al., 2024), for example, by generating more diverse solutions for difficult questions (Tong et al., 2024), or constructs new questions and answers using techniques like backward reasoning (Yu et al., 2023; Jiang et al., 2024), alternatively, it involves the complete creation of novel problems, such as generating entirely new mathematical questions based on key concepts, seed datasets, or specific examples (Tang et al., 2024; Li et al., 2024a) and leveraging strong models to provide solutions for them.

Although the aforementioned methods have achieved significant progress in data, their core paradigm remains on having models learn from correct examples, failing to fully exploit the learning value inherent in error data. To address this limitation, recent studies have begun exploring the learning from errors. For example, LLM2LLM (Lee et al., 2024) iteratively synthesizes new data based on data points where the student model answered incorrectly; works like LEMMA (Pan et al., 2025) attempt to synthesize common errors within the data and guide the model to reflect and correct them. However, these initial explorations still have shortcomings. The data collection strategy relied upon by LLM2LLM is relatively inefficient, potentially limiting the scale and diversity of the samples. LEMMA although introduces a reflection and correction mechanism, it lacks subsequent reinforcement training. In contrast to existing works, our study proposes an approach. We not only design a data enhancement pipeline to synthesize high quality data but also structure the reflection and correction process of the model and give subsequent reinforcement training.

Self-Reflection and Correction in LLMs Self-reflection and correction represent a crucial capability for LLMs. Existing research primarily follows two paths: one leverages external feedback-guided correction, utilizing external agents such as verifiers, critic models, or human experts (Lightman et al., 2023; Pan et al., 2024; Scheurer et al., 2023; Du et al., 2023) to provide error signals and correction guidance, the other employs reflection and self-correction (Krause and Stark, 2010; Han et al., 2024; Weng et al., 2023; Li et al., 2025) to

cultivate intrinsic critical thinking abilities. However, the former approach relies either on the availability of explicit error signals, such as code error messages (Chen et al., 2023) tool-use parameters (Gou et al., 2023) or guidance from critical models (Wang et al., 2024), the latter struggles to deeply comprehend the inherent logic of answers, often failing to achieve fundamental self-correction and sometimes even leading to performance degradation (Huang et al., 2023; Zhang et al., 2025a). In this paper, we focus on exploring how to enable models to understand the internal logic and reasoning processes of their own responses in scenarios lacking explicit error signals, thereby achieving more essential reflection and correction.

Group Relative Policy Optimization GRPO (Guo et al., 2025) is a reinforcement learning method that differs from conventional approaches such as PPO (Schulman et al., 2017) and DPO (Rafailov et al., 2023) . GRPO eliminates the need for a separate value network by employing a group-level reward normalization mechanism to fine-tune language models. Specifically, it reconstructs the standard policy gradient objective by introducing relative advantage calculations across multiple response groups corresponding to the same query. This enables effective model training in scenarios with sparse supervision signals, where feedback is only available after the complete generation sequence.

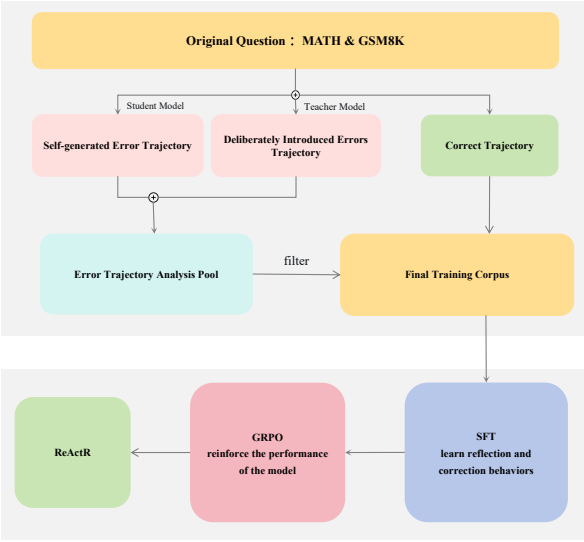

Figure 2: Overview of our framework which consists of two key components: difficulty aware and error types augmentation pipeline to construct diverse error-correction supervision and progressive training that combines SFT with reinforcement learning (GRPO).

3 Methodology

ReActR (**Re**asoning through **E**rror-**Act**ivated **R**eflection) is a post-training framework designed to improve mathematical reasoning by leveraging erroneous trajectories illustrated in Figure 2.

3.1 Error Analysis

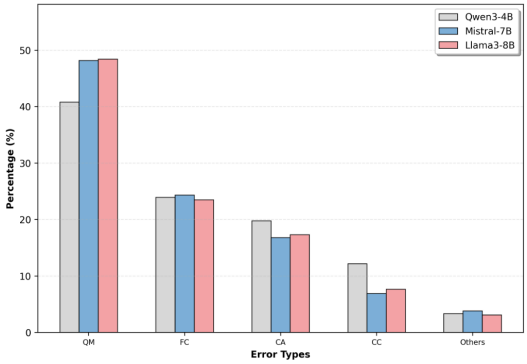

Figure 3: Error type distribution in self-generated erroneous trajectories on the math test set across Qwen3-4B, Mistral-7B, and Llama3-8B, classified by Gemini2.5-Pro, with minor error types consolidated into Others.

To systematically understand the common error types of LLMs in mathematical reasoning, we adopt a revised error taxonomy adapted from Pan et al. (2025); Li et al. (2024c) (Table 1). We analyze the error type distribution of representative small-scale models (Qwen3-4B, Mistral-7B, and Llama3-8B) on the MATH test set, as shown in Figure 3. The results indicate that these models frequently fail due to question misinterpretation (QM), concept confusion (CC) and calculation errors (CA) Pan et al. (2025). These observations motivate our subsequent data augmentation strategy, which explicitly targets more frequency error types.

To enable models to learn from erroneous samples, we prompt a teacher model with the original problem, the erroneous trajectory and the ground-truth correct trajectory to generate structured data including: (1) the error type label; (2) an explanation of how the error propagates to the error final answer; (3) a corrected reasoning trajectory and the correct answer and (4) a brief lesson summarizing how to avoid similar errors in the future. This design yields a complete learning sequence : original problem → erroneous answer → error analysis → error correction → lessons learned , which serves as the basis for our training data.

Table 1: The error taxonomy is adapted from (Pan et al., 2025; Li et al., 2024c). We exclude some less frequent error types.

Error Type	Definition
Calculation Error (CA)	Error appears during the calculation process.
Formula Confusion Error (FC)	Error appears when applying formula in an inappropriate scenario.
Question Misinterpretation Error (QM)	Error appears because the question is misunderstood, such as ignoring specific constraints in the question.
Missing Step (MS)	Error entails an incomplete generation of the reasoning process, lacking a necessary step.
Confusing Concept Error (CC)	Error occurs because two similar but actually different concepts are mistakenly confused.
Nonsensical Output (NO)	Inconceivable, illogical, or question-irrelevant output.

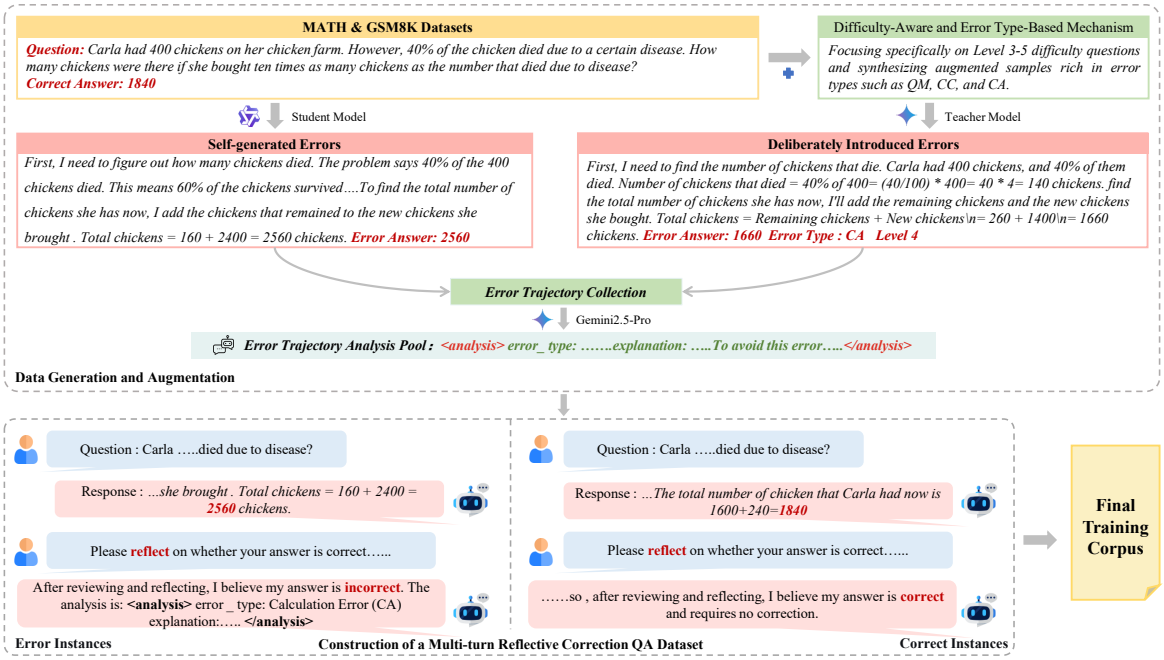

Figure 4: Difficulty-aware and Error-type Augmentation Pipeline: We first prompt the model to self-generate error trajectories, then employ difficulty-aware and error-type augmentation modules to guide the teacher model in targeted generating error trajectories. Both types of error trajectories are merged and processed for analysis. By incorporating correct trajectories, we construct a multi-turn dialogue reflection and correction dataset containing both positive and negative samples, which serves as the final training dataset for the supervised fine-tuning.

3.2 Difficulty Aware and Error Types augmentation pipeline

To define the focus of our augmentation strategy, we evaluate Qwen3-4B, Mistral-7B, and Llama3-8B across difficult levels on the MATH dataset. We report accuracy averaged over 10 inference runs (Figure 5). These models perform reasonably well on easier problems (Levels 1–2), their accuracy drops substantially on harder problems (Levels 3–5), indicating a clear bottleneck. This motivates us to prioritize generating erroneous trajectories from more difficult problems and to increase the coverage of more frequency error types.

Accordingly, We propose a difficulty-aware and error-type augmentation pipeline (Figure 4) with two stages. First, We sampled the MATH and GSM8K datasets using the Qwen3-4B, collecting

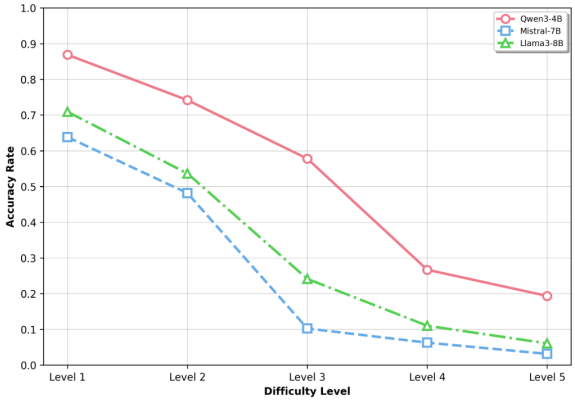

Figure 5: Accuracy Distribution of Qwen3-4B, Mistral-7B, and Llama3-8B on Level 1-5 Problems in the Math test set.

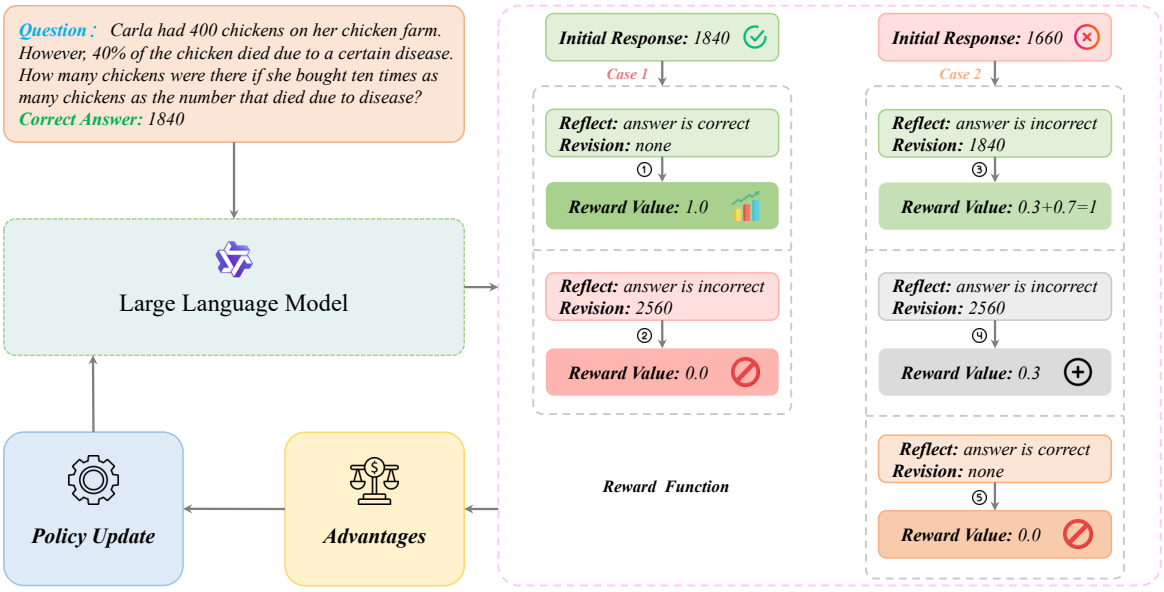

Figure 6: The reward functions of GRPO. For cases where the initial answer is correct: (1) if the reflection deems it correct, award 1.0 point; (2) if the reflection deems it incorrect and provides an erroneous correction, award 0.0 points. For cases where the initial answer is incorrect: (3) if the reflection deems it incorrect and provides a correct correction, award 0.3+0.7=1.0 point; (4) if the reflection deems it incorrect but the correction is wrong, award 0.3 points for the reflection; (5) if the reflection deems it correct, award 0.0 points.

erroneous reasoning trajectories and filtering out degenerate outputs (e.g., nonsensical or question irrelevant generations), which called self-generated errors. Second, we use the teacher model (Gemini 2.5 Pro) to generate targeted erroneous trajectories that covering more difficult problems (Levels 3–5) and more frequency error types (QM, CC, CA) and generate corresponding corrected reasoning trajectories and explanations. This augmentation strategy improves the training data along problem difficulty and error-type diversity, providing high quality supervision for subsequent supervised fine-tuning(SFT).

3.3 Self-Reflection and Correction Dataset

To teach the self-reflection and correction capabilities of the model, we organize the training data in a multi-turn dialogue format. Given a problem, the model first produces an initial solution and answer. It is then prompted to critically review its reasoning. If an error is detected, the model revises the solution and outputs a corrected answer, otherwise, it explicitly confirms the answer. Finally, the model summarizes a short lesson on how to avoid similar mistakes. This structured format encourages the model to internalize a complete attempt, reflect and revise, serving as the foundation for subsequent reinforcement learning.

3.4 GRPO with Reflection and Correction Rewards

Prior studies have proposed several strategies to enhance the reasoning capability of large language models. For instance, (Madaan et al., 2023) improves model performance through iterative refinement guided by self-feedback, while (Lai et al., 2024) strengthens long-chain reasoning by optimizing the model with step-wise preference datasets. Here, we apply GRPO to our fine-tuned model to enhance its self-reflection and correction capabilities at a fine-grained level, thereby improving overall performance. We categorize model behavior into five outcome types: (1) the initial answer is correct and reflection correctly confirms it; (2) the initial answer is correct but reflection incorrectly rejects it and produces a wrong revision; (3) the initial answer is incorrect but reflection mistakenly confirms it; (4) the initial answer is incorrect and reflection correctly identifies and fixes it; (5) the initial answer is incorrect, reflection detects the error, but the revised answer remains incorrect. Therefore,we design two level reward signals based on these classifications (Figure 6). At the reflection level, the model receives 1.0 rewards for correctly confirming a correct answer and 0.3 rewards for correctly identifying an incorrect one. At the correction level, it receives 0.7 rewards for

correctly revising the incorrect answer. This design aligns reward signals with reflective judgment and corrective accuracy and ultimately improves mathematical reasoning performance of models.

4 Experiments

4.1 Experimental Setup

Dataset. We use the training splits of MATH (Hendrycks et al., 2021) and GSM8K (Cobbe et al., 2021) as seed datasets. We prompt Qwen3-4B (Yang et al., 2025) (temperature $= 0.6$) to generate reasoning trajectories and collect erroneous solutions which called self-generated. Subsequently, the seed datasets were processed through the Difficulty-Aware and Error-Type Augmentation Pipeline to collect, analyze and correct erroneous trajectories with targeted difficult levels and error types. In total, we construct 25K error-correction samples. To balance supervision, we additionally include 25K correct trajectories, resulting in a 50K multi-turn dataset for SFT.

Evaluation Metrics. We evaluate models on in-domain benchmarks GSM8K and MATH. To assess generalization, we additionally report results on DeepMind Mathematics (Saxton et al., 2019), College Math (Tang et al., 2024), and OlympiadBench-Math (He et al., 2024). We report pass@1 accuracy for all datasets: for each problem, the model generates a single response and is counted as correct if the extracted final answer matches the ground-truth answer.

Baselines. We evaluate ReActR on three backbone models: Qwen3-4B, Mistral-7B (Jiang et al., 2023), and Llama3-8B (Dubey et al., 2024). We compare against (1) a standard SFT baseline trained on the combined GSM8K and MATH training set, and (2) representative advanced approaches that leverage data augmentation or reflection mechanisms, including MetaMath (Yu et al., 2023), RefAug (Zhang et al., 2024), and LEMMA (Pan et al., 2025). Additional implementation details are provided in Appendix A.

Teacher Model. We use Gemini 2.5 Pro as the teacher model for targeted error generation and structured data synthesis. Our choice is mainly motivated by the need for strong counterfactual instruction following and stable structured generation (Zhang et al., 2025b; Kumar et al., 2025), as the teacher must intentionally produce plausible reasoning errors of specific types together with coherent analyses and corrections. Gemini 2.5 Pro is well suited to this task and offers a favorable trade-off between generation quality and cost.

4.2 Main Results

Table 2 reports the main results on both in-domain and out-of-domain benchmarks. We summarize four key findings.

Finding 1: ReActR consistently outperforms all competitive baselines across backbones. Our method achieves strong overall performance on all three base models (Qwen3-4B, Mistral-7B, and Llama3-8B) across all evaluation benchmarks. On Qwen3-4B, our method reaches an average accuracy of 57.3% , outperforming LEMMA by 4.94 points, MetaMath by 6.98 points, and RefAug by 9.72 points. On Llama3-8B, our method achieves an average accuracy of 39.2%, surpassing LEMMA by 3.48 points and MetaMath by 5.28 points.

Finding 2: ReActR is robust across backbones with varying initial capabilities. Our method yields consistent improvements regardless of the initial strength of the backbone. For the stronger backbone such as Qwen3-4B, ReActR achieves 91.1% on GSM8K. For weaker backbones such as Mistral-7B and Llama3-8B, our method also delivers substantial gains, relative to their SFT baselines, our method improves the average accuracy of Mistral-7B and Llama3-8B by 13.6 and 13.5 points, respectively.

Finding 3: ReActR yields larger gains on more difficult benchmarks. The improvements are particularly pronounced on challenging datasets such as MATH, College Math. For instance, on Qwen3-4B, ReActR improves MATH accuracy by 5.2 points compared to LEMMA, and similar trends are observed for Mistral-7B and Llama3-8B. ReActR also achieves strong performance on College Math, demonstrating its effectiveness in tackling complex mathematical reasoning problems.

Finding 4: ReActR achieves strong training sample efficiency. Our method achieves strong performance with fewer training samples. With 50K training samples, ReActR outperforms MetaMath trained with 60K samples across all base models. For instance, on Llama3-8B, ReActR surpasses MetaMath by 5.28 points in average accuracy, suggesting that our method can achieve strong performance with fewer train samples.

Table 2: Performance comparison on mathematical benchmarks including MATH, GSM8K, College MATH, DeepMind-Mathematics, OlympiadBench-Math. The best result is highlighted in bold.

Model	# Samples	In-Domain			Out-of-Domain			AVG
		GSM8K	MATH	AVG	DeepMind-Mathematics	College Math	OlympiadBench-Math	
Qwen3-4B								
SFT	15k	79.8	48.6	64.2	47.9	22.5	13.2	42.4
RefAug	30k	83.3	51.9	67.6	58.6	26.0	18.1	47.6
MetaMath	60k	86.5	59.3	72.9	58.2	30.4	17.2	50.3
LEMMA	60k	86.6	60.5	73.5	62.1	29.9	22.7	52.4
Ours	50k	**91.1**	**65.7**	**78.4**	**68.9**	**35.2**	**25.6**	**57.3**
Mistral-7B								
SFT	15k	56.3	12.8	34.5	19.3	8.4	2.2	19.8
RefAug	30k	62.2	13.4	37.8	15.8	10.4	3.1	21.0
MetaMath	60k	71.6	24.7	48.1	26.9	14.6	5.3	28.6
LEMMA	60k	71.9	26.4	49.2	30.8	13.9	5.9	29.8
Ours	50k	**75.8**	**31.2**	**53.5**	**34.1**	**19.7**	**6.2**	**33.4**
Llama3-8B								
SFT	15k	65.6	21.6	43.6	22.5	14.1	4.7	25.7
RefAug	30k	73.2	27.3	50.2	25.9	18.2	4.9	29.9
MetaMath	60k	77.8	34.5	56.1	32.3	19.7	5.6	33.9
LEMMA	60k	76.6	36.4	56.5	37.1	22.3	6.3	35.7
Ours	50k	**81.3**	**39.2**	**60.2**	**41.5**	**26.4**	**7.5**	**39.2**

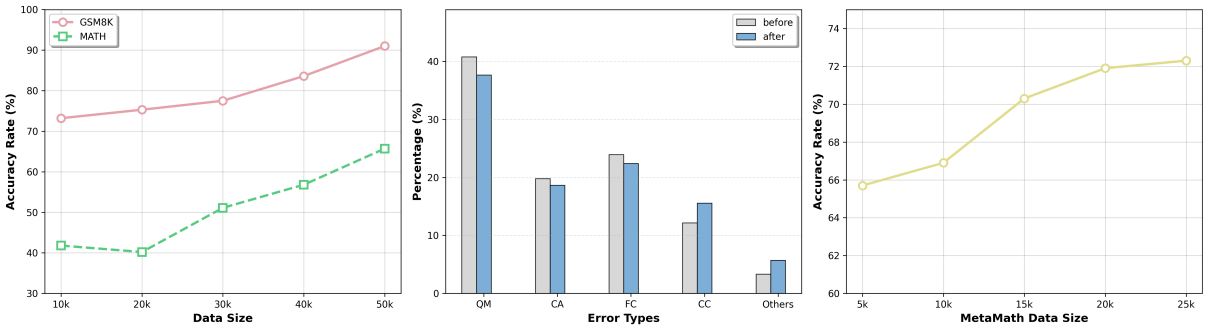

Figure 7: (a):Performance scaling behavior of our model on different sizes of augmented data on Qwen3-4B. (b):Error type changes on math test set before and after fine-tuning. (c): Average performance of the Qwen3-4B models fine-tuned on the combined dataset of Ours and MetaMath with different sizes of sampled data on math test set.

5 Analysis

5.1 Relationship between Augmented Data Size and Performance

We study how the amount of augmented data size supervision affects performance during training. Using Qwen3-4B as the backbone model, we select GSM8K and MATH as our initial training datasets and progressively increase the size of the augmented datasets from 10K to 50K in increments of 10K. Figure 7(a) reports accuracy on GSM8K and MATH.

We observe that ReActR provides limited gains when augmentation data is scarce but yields steadily increasing improvements as the augmented data size grows. A possible explanation is that insufficient structured supervision leads to weaker supervised signals, the model may not reliably learn the reflection-correction abilities, which having difficultly in detecting and correcting the errors.

5.2 Error type Changes before and after Training

To better understand how ReActR affects model's failure modes, we analyze the error type distribution on the MATH test dataset before and after training. We use Gemini2.5-Pro as the evaluator to classify error types. Figure 7(b) shows that after applying ReActR, the occurrence of high frequency error types, such as QM, CA are substantially re-

duced. This shift indicates that error type augmentation enables the model to avoid common failure patterns that frequently occur in LLMs.

5.3 Combine with MetaMath

We further examine whether ReActR can be integrated with other data augmentation pipelines. Specifically, we combine ReActR with MetaMath (Yu et al., 2023) by replacing varying portions of the correct trajectories in our SFT dataset with an equal number of MetaMath samples. Figure 7(c) shows that as the proportion of MetaMath data increases, the model's average performance on the MATH test dataset improves consistently. This suggests that MetaMath provides complementary supervision signals, such as diverse solution paths and backward reasoning strategies, which enrich the correct trajectories.

5.4 Ablation Study

We conduct ablation studies on Qwen3-4B to quantify the contribution of individual components. Results on GSM8K and MATH are reported in Table 3.

Error-type and difficulty-aware augmentation. We ablate two augmentation components: error-type augmentation and difficulty-aware augmentation. Out-Error removes error-type augmentation while keeping difficulty-aware generation, Out-Difficulty removes difficulty-aware augmentation while keeping error-type, and Out-Both removes both, relying only on self-generated erroneous trajectories. Removing either component leads to notable performance drops, while removing both causes the largest degradation, confirming that the two strategies are complementary.

Positive-negative ratio in SFT. We further study the impact of mixing correct and erroneous trajectories during SFT. Training only on erroneous trajectories (All-Error) leads to overly pessimistic reflection behavior, causing performance degradation and ultimately failing to converge or generate effective corrections. As a result, All-Error did not produce meaningful results during training. Training exclusively on correct trajectories (All-Correct) improves standard solution patterns but provides weaker reflective correction capabilities due to the lack of error exposure. The 1:1 mixture used in our method strikes a crucial balance, achieving substantially higher accuracy than either extreme.

Impact of GRPO. Finally, we remove the GRPO stage (Out-GRPO) and evaluate the SFT-only model. Removing GRPO reduces accuracy by nearly 3 points on MATH, demonstrating that reinforcement learning provides complementary gains beyond SFT. We also observe that GRPO benefits more when initialized from a stronger supervised model, suggesting that high quality SFT provides more informative reward signals and stabilizes reinforcement learning.

Table 3: Effect of these models on Qwen3-4B

Method	GSM8K	MATH
Out-Error	84.6	56.2
Out-Difficulty	87.5	60.3
Out-Both	82.6	46.3
All-Correct	79.8	48.6
All-Error	-	-
Out-GRPO	88.4	62.1
Ours	**91.1**	**65.7**

6 Conclusion

In this paper, we propose **ReActR**, a post-training framework that improves mathematical reasoning by learning from erroneous trajectories. ReActR constructs a high quality multi-turn reflection-correction dataset through difficulty-aware and error-type augmentation, enabling models to acquire foundational self-reflection and correction behaviors via supervised fine-tuning. We further introduce GRPO with reward functions to reinforce both reflective judgment and corrective accuracy. Extensive experiments on multiple LLMs demonstrate that ReActR consistently outperforms strong baselines on both in-domain and out-of-domain benchmarks.

7 Limitations

Although our framework effectively enhances the mathematical reasoning capabilities of large language models, it still has some limitations.

First, we rely on a teacher model (Gemini 2.5 Pro) to generate targeted erroneous trajectories along with their analysis and corrections. Although we apply filtering, the generated content may still contain subtle inaccuracies or ambiguities. This means that the upper bound of data quality is constrained by the teacher model's inherent capabilities. Exploring alternative teacher models such as GPT-4 (Achiam et al., 2023) and more robust verification strategies is an important direction for future work.

Second, when a problem involves multiple intertwined error types (e.g., QM accompanied by CA), the model's performance can still degrade noticeably, as the current framework mainly focuses on identifying and correcting the dominant error. In such cases, fixing one error may expose another previously hidden one, making accurate multi-error disentanglement more challenging. Developing iterative multi-turn correction strategies that resolve compound errors layer by layer is an important direction for future work.

Finally, while our current dataset provides high quality supervision, its scale remains limited, which may restrict further improvements. Expanding the dataset and extending ReActR to other verifiable domains, especially coding, are important directions for future work.

Acknowledgements

The author is deeply grateful to **Prof. Weibing Wan (Shanghai University of Engineering Science)** for his invaluable mentorship, thoughtful guidance, and constructive feedback throughout the course of this research. His supervision and scholarly insights have been instrumental in shaping this work.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. *arXiv preprint arXiv:2402.00157*.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. 2023. Teaching large language models to self-debug. *arXiv preprint arXiv:2304.05128*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first International Conference on Machine Learning*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2023. Critic: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Haixia Han, Jiaqing Liang, Jie Shi, Qianyu He, and Yanghua Xiao. 2024. Small language model can self-correct. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18162–18170.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, and 1 others. 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)*, pages 3828–3850.

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 arXiv:2103.03874*.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2023. Large language models cannot self-correct reasoning yet. *arXiv preprint arXiv:2310.01798*.

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

Weisen Jiang, Han Shi, Longhui Yu, Zhengying Liu, Yu Zhang, Zhenguo Li, and James Kwok. 2024. Forward-backward reasoning in large language models for mathematical verification. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6647–6661.

Ulrike-Marie Krause and Robin Stark. 2010. Reflection in example-and problem-based learning: Effects of

reflection prompts, feedback and cooperative learning. *Evaluation & research in education*, 23(4):255–272.

Sai Adith Senthil Kumar, Hao Yan, Saipavan Perepa, Murong Yue, and Ziyu Yao. 2025. Can llms simulate personas with reversed performance? a benchmark for counterfactual instruction following. *arXiv preprint arXiv:2504.06460*.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. 2024. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*.

Nicholas Lee, Thanakul Wattanawong, Sehoon Kim, Karttikeya Mangalam, Sheng Shen, Gopala Anumanchipalli, Michael Mahoney, Kurt Keutzer, and Amir Gholami. 2024. Llm2llm: Boosting llms with novel iterative data enhancement. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6498–6526.

Chen Li, Weiqi Wang, Jingcheng Hu, Yixuan Wei, Nanning Zheng, Han Hu, Zheng Zhang, and Houwen Peng. 2024a. Common 7b language models already possess strong math capabilities. *arXiv preprint arXiv:2403.04706*.

Chengpeng Li, Zheng Yuan, Hongyi Yuan, Guanting Dong, Keming Lu, Jiancan Wu, Chuanqi Tan, Xiang Wang, and Chang Zhou. 2024b. Mugglemath: Assessing the impact of query and response augmentation on math reasoning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10230–10258.

Jason Li, Lauren Yraola, Kevin Zhu, and Sean O'Brien. 2025. Error reflection prompting: Can large language models successfully understand errors? In *The Sixth Workshop on Insights from Negative Results in NLP*, pages 157–170.

Xiaoyuan Li, Wenjie Wang, Moxin Li, Junrong Guo, Yang Zhang, and Fuli Feng. 2024c. Evaluating mathematical reasoning of large language models: A focus on error identification and correction. *arXiv preprint arXiv:2406.00755*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

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

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36:46534–46594.

Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. 2023. Teaching small language models to reason. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 2: short papers)*, pages 1773–1781.

Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. 2024. Automatically correcting large language models: Surveying the landscape of diverse automated correction strategies. *Transactions of the Association for Computational Linguistics*, 12:484–506.

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*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. 2019. Analysing mathematical reasoning abilities of neural models. *arXiv preprint arXiv:1904.01557*.

Jérémy Scheurer, Jon Ander Campos, Tomasz Korbak, Jun Shern Chan, Angelica Chen, Kyunghyun Cho, and Ethan Perez. 2023. Training language models with language feedback at scale. *arXiv preprint arXiv:2303.16755*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Zhengyang Tang, Xingxing Zhang, Benyou Wang, and Furu Wei. 2024. Mathscale: Scaling instruction tuning for mathematical reasoning. *arXiv preprint arXiv:2403.02884*.

Yuxuan Tong, Xiwen Zhang, Rui Wang, Ruidong Wu, and Junxian He. 2024. Dart-math: Difficulty-aware rejection tuning for mathematical problem-solving. *Advances in Neural Information Processing Systems*, 37:7821–7846.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. 2023. Large language models are better reasoners with self-verification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2550–2575.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, and 1 others. 2024. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

Huaiyuan Ying, Shuo Zhang, Linyang Li, Zhejian Zhou, Yunfan Shao, Zhaoye Fei, Yichuan Ma, Jiawei Hong, Kuikun Liu, Ziyi Wang, and 1 others. 2024. Internlm-math: Open math large language models toward verifiable reasoning. *arXiv preprint arXiv:2402.06332*.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2023. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*.

Ping Yu, Jing Xu, Jason Weston, and Ilia Kulikov. 2024. Distilling system 2 into system 1. *arXiv preprint arXiv:2407.06023*.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2023. Mammoth: Building math generalist models through hybrid instruction tuning. *arXiv preprint arXiv:2309.05653*.

Qingjie Zhang, Di Wang, Haoting Qian, Yiming Li, Tianwei Zhang, Minlie Huang, Ke Xu, Hewu Li, Liu Yan, and Han Qiu. 2025a. Understanding the dark side of llms' intrinsic self-correction. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 27066–27101.

Qinyan Zhang, Xinping Lei, Ruijie Miao, Yu Fu, Haojie Fan, Le Chang, Jiafan Hou, Dingling Zhang, Zhongfei Hou, Ziqiang Yang, and 1 others. 2025b. Inverse ifeval: Can llms unlearn stubborn training conventions to follow real instructions? *arXiv preprint arXiv:2509.04292*.

Zhihan Zhang, Tao Ge, Zhenwen Liang, Wenhao Yu, Dian Yu, Mengzhao Jia, Dong Yu, and Meng Jiang. 2024. Learn beyond the answer: Training language models with reflection for mathematical reasoning. *arXiv preprint arXiv:2406.12050*.

Jiaru Zou, Yikun Ban, Zihao Li, Yunzhe Qi, Ruizhong Qiu, Ling Yang, and Jingrui He. 2025. Transformer copilot: Learning from the mistake log in llm fine-tuning. *arXiv preprint arXiv:2505.16270*.

A Appendix

A.1 Targeted Error Trajectory Generated and Analysis

We generate erroneous trajectories in two ways. First, we sample self-generated erroneous solutions from Qwen3-4B on GSM8K and MATH. Second, we prompt Gemini 2.5 Pro to generate targeted erroneous trajectories on MATH with specified difficult levels and error types. All synthesized trajectories are filtered to remove invalid or degenerate samples.

The prompt to systematically collect erroneous reasoning trajectories and conduct thorough analysis, as shown in Figure 9, Figure 10.

The final format is structured as a reflection and correction process, as shown in Figure 8. For a given problem, the model first produces an initial response. It then performs a reflective evaluation of this response guided by specific prompts. If the reflection identifies errors, the model generates a revised and corrected answer. Finally, the model summarizes the lessons or insights learned from solving the problem.

A.2 Training and Resource Details

we fine-tune a wide range of base models, including Qwen3-4B, LLaMA3-8B, and Mistral-7B. For supervised fine-tuning (SFT), we use LLaMA-Factory with LoRA adapters and train for 4 epochs with a batch size of 4, a learning rate of 1×10^{-5}, and a cosine learning rate scheduler.

After SFT, we further apply GRPO for reinforcement learning using verl. For each problem, we sample five responses as a group and compute advantages through relative comparisons. We adopt a two-level reward design with five outcome cases, and train for 3 epochs with a fixed learning rate of 1×10^{-6}.

For data synthesis, our pipeline processes over 100 million tokens in total to generate, analyze, and correct 25K reasoning trajectories, resulting in the API cost of slightly over $700 USD.

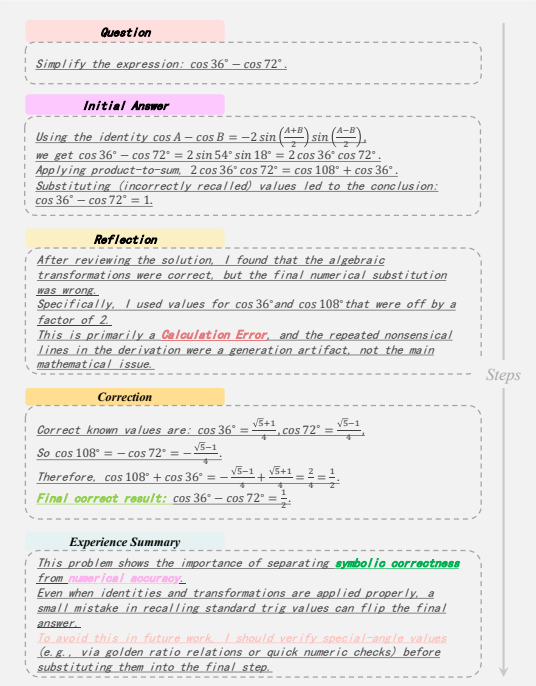

Figure 8: The reflection and correction process for erroneous reasoning trajectories.

A.3 Evaluation

We evaluate our models using the following five benchmarks:

GSM8K (Cobbe et al., 2021) contains 8,792 elementary school math problems, with 7,473 for training and 1,319 for testing.

MATH (Hendrycks et al., 2021) consists of 12,500 problems from high school math competitions, split into 7,500 for training and 5,000 for testing. The problems in MATH are divided into 7 categories and span 5 difficult levels.

DeepMind Mathematics (Saxton et al., 2019) includes 1,000 problems covering algebra, arithmetic, calculus, and probability.

College Math (Tang et al., 2024) comprises 2,818 problems. These problems cover 7 major mathematical areas: Algebra, Precalculus, Calculus, Vector Calculus, Probability, Linear Algebra, and Differential Equations.

OlympiadBench-Math (He et al., 2024) includes 675 problems of Olympiad-level difficulty.

We briefly introduce the three baseline methods used for comparison below.

MetaMath (Yu et al., 2023) enhances mathematical reasoning by constructing an augmented training corpus through answer augmentation, question rephrasing, and backward reasoning strategies. These techniques diversify both problem formulations and solution trajectories, providing richer supervision for SFT.

RefAug (Zhang et al., 2024) strengthens reflective reasoning by appending a reflection section to each solution. The reflection includes proposing the alternative solution and solving the similar problem, encouraging the performance of the model.

LEMMA (Pan et al., 2025) improves mathematical reasoning via SFT with an error-type based data augmentation strategy and reflection and self-correction, enabling the model to better recognize common reasoning mistakes and learn more reliable solution behaviors.

Key Prompts for Targeted Generation of Erroneous Trajectories

Instruction

You are a careless student. I will provide you with a problem and the ground-truth solution. Your goal is to propose a new solution that contains specific errors.

you should:

Understand the Problem and Solution: Carefully read the problem and the ground-truth solution.

Introduce Specific Errors: Intentionally introduce an error into the reasoning steps that an intelligent student will make, not blatant or superficial.

Continue Reasoning: Continue reasoning after the error step until you reach a wrong answer .

Format Answer: Give your final number in format : The final answer is [your answer]. The final answer should be the str after #### . Stop at the final wrong answer.

Figure 9: Key Prompts for Targeted Generation of Erroneous Reasoning Trajectories.

Key Prompts for Analysis Erroneous Trajectories

Instruction

You are an expert at analyzing mathematical reasoning errors. Below is a problem, a correct solution , and a model's incorrect response.

Task:

1. Compare the model's response with the correct solution
2. Explain why this error led to the wrong answer
3. Provide the correct reasoning for the problematic step and correct answer
4. Suggest how to avoid similar errors in future

Output your brief analysis in the following format within tags:

<analysis>error type: [error type]explanation: [Your explanation here]</analysis>

Figure 10: Key Prompts for analysis Erroneous Reasoning Trajectories.