MathSmith: Towards Extremely Hard Mathematical Reasoning by Forging Synthetic Problems with a Reinforced Policy

Shaoxiong Zhan[1,3], Yanlin Lai[1], Ziyu Lu[1], Dahua Lin[2], Ziqing Yang[3*], Fei Tang[4*]
[1]Tsinghua University [2]The Chinese University of Hong Kong
[3]SenseTime Research [4]East China Normal University
{zhansx24,laiyl24,luziyu24}@mails.tsinghua.edu.cn
dhlin@ie.cuhk.edu.hk, ziqingyang@gmail.com, ftan@mail.ecnu.edu.cn

Abstract

Large language models have achieved substantial progress in mathematical reasoning, yet their advancement is limited by the scarcity of high-quality, high-difficulty training data. Existing synthesis methods largely rely on transforming human-written templates, limiting both diversity and scalability. We propose MathSmith, a novel framework for synthesizing challenging mathematical problems to enhance LLM reasoning. Rather than modifying existing problems, MathSmith constructs new ones from scratch by randomly sampling concept–explanation pairs from PlanetMath, ensuring data independence and avoiding contamination. To increase difficulty, we design nine predefined strategies as soft constraints during rationales. We further adopts reinforcement learning to jointly optimize structural validity, reasoning complexity, and answer consistency. The length of the reasoning trace generated under autoregressive prompting is used to reflect cognitive complexity, encouraging the creation of more demanding problems aligned with long-chain-of-thought reasoning. Experiments across five benchmarks, categorized as easy & medium (GSM8K, MATH-500) and hard (AIME2024, AIME2025, OlympiadBench), show that MathSmith consistently outperforms existing baselines under both short and long CoT settings. Additionally, a weakness-focused variant generation module enables targeted improvement on specific concepts. Overall, MathSmith exhibits strong scalability, generalization, and transferability, highlighting the promise of high-difficulty synthetic data in advancing LLM reasoning capabilities.

1 Introduction

In recent years, large language models (LLMs) have achieved remarkable progress in reasoning tasks across mathematics, science, and programming (Guo et al., 2025; Jaech et al., 2024). As models continue to scale in both size and architectural sophistication, their capabilities have expanded from solving elementary-level math problems to addressing Olympiad-level challenges (Shao et al., 2024; OpenAI, 2025), gradually revealing their potential for general-purpose intelligence. However, the advancement of reasoning ability now faces a critical bottleneck: the scarcity of high-quality, high-difficulty mathematical problems for training and evaluation limits the upper bound of model performance. Moreover, a lack of diversity in existing problem distributions raises concerns about models memorizing recurring patterns rather than truly reasoning (Huang et al., 2025a), further highlighting the urgent need for diverse and challenging mathematical data to support continued progress.

Most existing approaches to mathematical problem synthesis rely on extracting templates, structures, or conceptual patterns from existing questions (Huang et al., 2025b), followed by rewriting (Yu et al., 2024), augmentation (Toshniwal et al., 2025; Liu et al., 2025), question back-translation (Lu et al., 2024), or evolutionary transformations (Luo et al., 2023). While these methods enhance data diversity to some extent, they remain fundamentally constrained by the distribution and structure of human-authored problems, often lacking generation autonomy and precise difficulty control. As articulated in the Bitter Lesson (Sutton, 2019), sustainable progress in AI is ultimately driven by general purpose, computation-heavy methods rather than handcrafted knowledge. In line with this perspective, we argue that future reasoning agents should be able to autonomously generate high-quality, intellectually challenging problems. To this end, we introduce **MathSmith**, a novel framework that emulates the role of a mathematical blacksmith: it extracts raw materials (i.e., concept and explanation pairs) and progressively

*Corresponding authors.

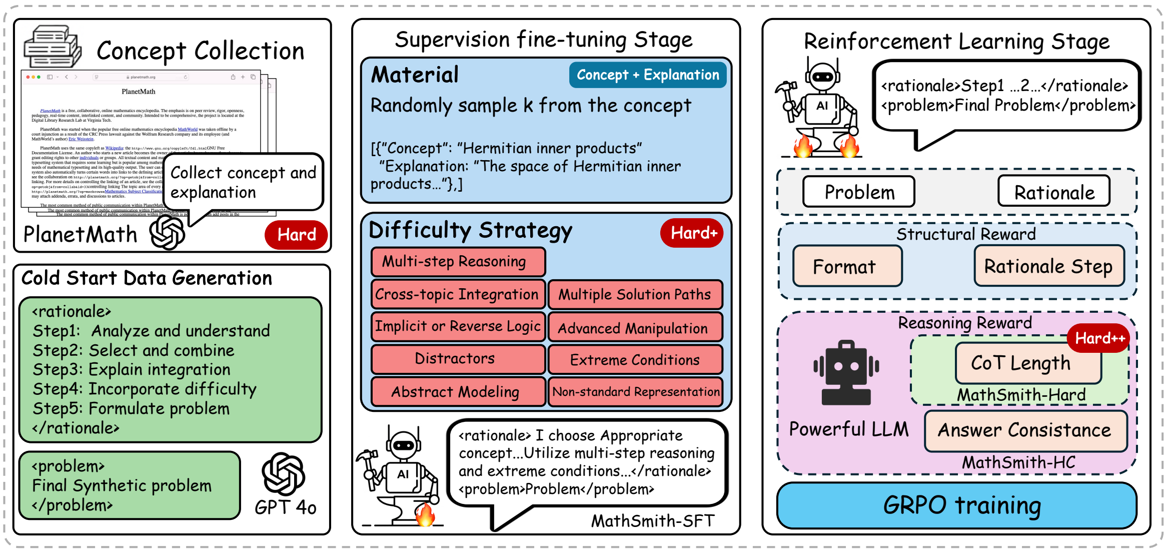
Concept Collection
Collect concept and explanation
PlanetMath
Hard
Cold Start Data Generation
<rationale>
Step1: Analyze and understand
Step2: Select and combine
Step3: Explain integration
Step4: Incorporate difficulty
Step5: Formulate problem
</rationale>
<problem>
Final Synthetic problem
</problem>
GPT 4o
Supervision fine-tuning Stage
Material
Concept + Explanation
Randomly sample k from the concept
[{"Concept": "Hermitian inner products"
"Explanation: "The space of Hermitian inner products..."},]
Difficulty Strategy
Hard+
Multi-step Reasoning
Cross-topic Integration
Multiple Solution Paths
Implicit or Reverse Logic
Advanced Manipulation
Distractors
Extreme Conditions
Abstract Modeling
Non-standard Representation
<rationale> I choose Appropriate concept...Utilize multi-step reasoning and extreme conditions...</rationale>
<problem>Problem</problem>
MathSmith-SFT
Reinforcement Learning Stage
<rationale>Step1 ...2...</rationale>
<problem>Final Problem</problem>
Problem
Rationale
Structural Reward
Format
Rationale Step
Reasoning Reward
Hard++
CoT Length
MathSmith-Hard
Powerful LLM
Answer Consistance
MathSmith-HC
GRPO training

Figure 1: The MathSmith workflow, comprising the phases of concept and explanation collection, supervised fine-tuning, and reinforcement learning.

refines them into complex and coherent mathematical problems.

To enhance both the difficulty and the quality of the synthesized problems, we begin by analyzing the structural and cognitive features of existing high-difficulty questions. This analysis leads to the formulation of nine pre-defined difficulty strategies in Figure 1, including multi-step reasoning, cross-topic integration, implicit or reverse logic, distractor construction, abstract modeling, multiple solution paths, advanced manipulation extreme conditions and non-standard representation, which serve as soft constraints during generation. Additionally, we introduce a reinforcement learning stage to optimize the generation process across three dimensions: structural validity, reasoning complexity, and answer consistency. Drawing inspiration from the long CoT paradigm (Guo et al., 2025), we adopt the reasoning trace length, as produced by models under autoregressive CoT prompting, as an indirect estimation of problem difficulty, and incorporate it into our reward design. We observe that more challenging problems tend to elicit significantly longer reasoning traces, as shown in Figure 2, suggesting a potential link between problem difficulty and reasoning depth. While it remains uncertain whether such problems definitively enhance the reasoning capabilities of LLMs, they offer a promising direction worth exploring. Building on this insight, our method synthesizes problems that induce longer reasoning sequences, enriching the pool of high-difficulty data and fostering deeper reasoning in LLMs.

We categorize widely used benchmarks into two difficulty tiers: easy & medium (GSM8k, MATH-500) and hard (AIME2024, AIME2025, Olympiad). Compared to a variety of existing methods, MathSmith achieves significantly better performance under both short-CoT and long-CoT prompt settings, producing relative improvements of 9.8%–18.1% on the hard benchmarks. In addition, our weakness-focused variant generation mechanism effectively improves model performance on specific underperforming concepts, and the synthesized problems generalize well across different reasoning tasks. Moreover, extended experiments show that MathSmith maintains strong performance as both the number of problems and the model size increase, demonstrating its superiority in reasoning depth, scalability, and effectiveness on larger models.

Our main contributions are as follows: (1) We synthesize mathematical problems by randomly sampling concept–explanation pairs and constructing problems through step-by-step rationale generation, avoiding reliance on real-world templates and minimizing data contamination; (2) We propose the MathSmith framework, which incorporates nine difficulty strategies, a multi-objective reinforcement learning mechanism for optimizing structural integrity, reasoning depth, and solution consistency, as well as a weakness-focused variant generation module for targeted concept-level improvement; (3) We conduct extensive evaluations demonstrating that the synthesized problems significantly enhance model performance on challenging benchmarks such as AIME2024, AIME2025, and Olympiad, particularly under long-chain-of-thought prompting setups.

2 Related Work

Mathematical Reasoning with Large Language Models. Large language models (LLMs) have shown growing capabilities in mathematical reasoning, driven by both training and inference advancements. At the training stage, some works enhance mathematical foundations through continued pre-training on large-scale corpora (Shao et al., 2024; Wang et al., 2024), while others focus on post-training with curated instruction datasets such as OpenMathInstruct (Toshniwal et al., 2025), Mammoth (Yue et al., 2024) and DART-Math (Tong et al., 2024). For inference, Chain-of-Thought (CoT) prompting (Wei et al., 2022) enables step-by-step reasoning, and Program-of-Thought (PoT) (Chen et al., 2023) incorporates the use of tools for solving complex problems. PromptCOT (Zhao et al., 2025) combines concept-driven prompts with multi-step planning to generate Olympiad-level problems and is most relevant to our work, though it still relies on human-selected concepts and lacks deeper reasoning control. Other approaches, such as Llemma (Azerbayev et al., 2024) and MathPrompter (Imani et al., 2023), further explore structured or task-specific alignment. In contrast, our method controls the reasoning structure from the source, targeting logical consistency and difficulty through both supervised and reinforcement training.

Math Instruction Synthesis and Difficulty Control. Recent studies have explored data synthesis methods to improve LLMs on mathematical tasks. MetaMath (Yu et al., 2024) increases the diversity of problems through rewriting, NuminaMath (Li et al., 2024a) resamples the benchmarks with CoT guidance, and GSM-Plus (Li et al., 2024b) increases the robustness with distribution-based augmentation, while others like ScaleQuest (Ding et al., 2024) focus on generating novel questions from scratch. To better control difficulty, JiuZhang3.0 (Zhou et al., 2024) structures prompts tiered by educational stage. PromptCOT (Zhao et al., 2025), MathScale (Tang et al., 2024) and WizardMath (Luo et al., 2023) introduce concept planning, graph-based concept combination, and reinforcement-driven evolution to guide generation. Key-point-driven methods (Huang et al., 2025b) further support multi-concept integration. However, most methods still rely on seed prompts derived from real human-written math problems, limiting their generative autonomy. Moreover, their difficulty control typically depends on prompt-level labels assigned by language models, which lack objective justification. Our framework constructs problems from randomly sampled concept–explanation pairs, bypassing existing problems entirely. With structural, complexity, and consistency-aware rewards, MathSmith generates high-quality, verifiable problems that push LLMs toward stronger mathematical reasoning.

3 Methodology

As illustrated in Figure 1, the MathSmith framework generates challenging mathematical problems through a core three-stage process: (1) **Concept-Explanation Collection**: collecting challenging "Concept + Explanation" pairs from PlanetMath; (2) **Supervised Fine-Tuning Stage**: employing SFT on a seed dataset generated by GPT-4o to equip the model with an initial reasoning ability for formatted problem generation; and (3) **Reinforcement Learning Stage**: implementing RL to refine problem difficulty, where a reward function combines signals from format, solution complexity, and answer consistency.

Furthermore, the traceability of MathSmith-synthesized problems to their source concepts enables the **Weakness-Focused Improvement Pipeline**, a module designed for the targeted enhancement of model weaknesses, as shown in Figure 3.

3.1 Concept and Explanation Collection

We construct a dataset that contains 11,000 mathematical concepts and their explanations. The data for this dataset is sourced from PlanetMath (PlanetMath Community, 2024), a repository known for its extensive coverage of advanced mathematics and theoretically deep concepts. This choice ensures our resulting collection of concepts is inherently challenging. We first crawl the mathematics-related pages from its website and filter out entries that focus on biographical or historical content to ensure a clear focus on the mathematical concepts themselves. Subsequently, we utilize the GPT-4o API to automatically summarize the core concept of each page, which generates a collection of "Concept +

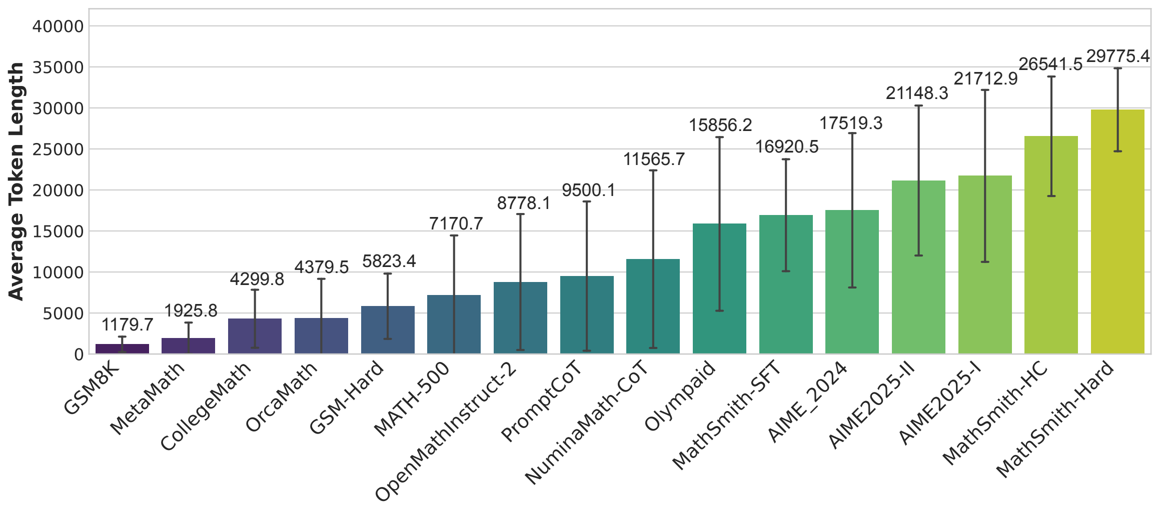

Figure 2: Average reasoning trace token length under thinking mode (Qwen3-30B-A3B) across various open-source math datasets. Problems synthesized by MathSmith variants elicit significantly longer reasoning, reflecting higher complexity.

Explanation" pairs.

3.2 Supervise Fine-tuning Stage

We adopt Qwen3-8B as the base model for problem generation and use GPT-4o to synthesize cold-start training data. Specifically, we randomly sample five concepts and their corresponding explanations from the constructed concept collection and provide them to the model as seed inputs, prompting it to generate math problems grounded in the given instructions and aligned with the sampled concepts.

Each generated sample is structured into two components: a rationale section that outlines the problem construction process, and a problem section that presents the final question. As illustrated in the "Cold Start Data Generation" module of Figure 1, the rationale consists of exactly five reasoning steps. This format serves both as a pedagogical scaffold and as a structural constraint during training, ensuring consistency in problem synthesis.

To further enhance the difficulty of the synthesized problems and encourage advanced mathematical reasoning, we incorporate insights from existing challenging Olympiad problems. Based on these insights, we design a set of nine predefined *difficulty strategies*, also detailed in Figure 1. Each generated problem is required to incorporate at least two of these strategies to ensure sufficient complexity. Following this process, we generate approximately 8,000 cold-start samples. These samples are used to fine-tune the Qwen3-8B model. The resulting model is referred to as **MathSmith-SFT**.

3.3 Reinforcement Learning Stage

We design a composite reward to guide the policy model toward generating valid, challenging, and consistent mathematical problems, comprising structural, complexity, and consistency components.

(1) Structural Reward. We assess whether the output contains both "rationale" and "problem" segments using a binary reward $r_{\text{format}} \in \{0, 1\}$. We further compute a step count reward r_{step} based on the number of reasoning steps N_{step} in the rationale:

$$r_{\text{step}} = \begin{cases} \frac{N_{\text{step}}}{5}, & N_{\text{step}} \le 5 \\ \max\left(1 - \frac{N_{\text{step}}-5}{5},\ 0\right), & \text{otherwise} \end{cases}$$

$$r_{\text{structure}} = \alpha_{\text{format}} \cdot r_{\text{format}} + \alpha_{\text{step}} \cdot r_{\text{step}}. \quad (1)$$

Here, α_{format} and α_{step} are weighting coefficients. The reward is maximized when the rationale contains exactly five steps, aligning with the prompt template.

(2) Reasoning Complexity Reward. To evaluate the difficulty of each generated problem, we utilize the teacher model Qwen3-30B-A3B to generate solutions. The complexity is estimated by measuring the token length of its reasoning trace.

Let $\ell_{\text{cot}}^{(i)}$ be the token length of the i-th reasoning trace among K independent samples. The complexity reward is computed as:

$$r_{\text{complexity}} = \frac{1}{K \cdot T_{\max}} \sum_{i=1}^{K} \ell_{\text{cot}}^{(i)}, \quad r_{\text{complexity}} \in [0, 1], \tag{2}$$

where $T_{\max}$ is a normalization constant (i.e., the maximum allowed CoT length).

Motivation for Reasoning-based Reward. We adopt reasoning trace length as a heuristic measure of problem complexity. Intuitively, more challenging problems tend to require deeper and more structured reasoning, leading to longer CoT traces under autoregressive prompting. While longer traces do not directly imply better generalization, they often contain low-entropy intermediate tokens (Wang et al., 2025), which are shown to provide more informative supervision signals during training. Thus, we encourage the synthesis of problems that induce longer reasoning as a proxy for higher cognitive complexity and better transferability.

(3) Answer Consistency Reward. To evaluate solution consistency, we sample K answers $\mathcal{A} = \{a_1, \ldots, a_K\}$ from the teacher model. If a majority answer exists, we assign a reward of 1; otherwise, 0:

$$r_{\text{consistency}} = \begin{cases} 1, & \exists a \in \mathcal{A} : \text{count}(a) > K/2 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

This encourages the policy model to generate problems that elicit deterministic reasoning behavior from the teacher model, indicating clarity and unambiguity in problem formulation.

(4) Final Reward. We combine the complexity and consistency objectives into a unified reasoning-oriented reward:

$$\begin{aligned} r_{\text{reasoning}} = \beta_{\text{complexity}} \cdot r_{\text{complexity}} \\ + \beta_{\text{consistency}} \cdot r_{\text{consistency}} \end{aligned} \tag{4}$$

where $\beta_{\text{complexity}}$ and $\beta_{\text{consistency}}$ are weighting coefficients.

The overall reinforcement signal provided to the policy model is the sum of structural and reasoning-based components:

$$r_{\text{total}} = r_{\text{structure}} + r_{\text{reasoning}}. \tag{5}$$

This reward guides the policy model to generate mathematically valid, non-trivial and verifiable problems by combining structural alignment, reasoning complexity, and answer consistency. We refer to the resulting model trained with both complexity and consistency components as **MathSmith-HC**, while the variant that excludes the consistency term and uses only the complexity reward is referred to as **MathSmith-Hard**.

We employ Group Relative Policy Optimization (GRPO) (Shao et al., 2024) to optimize the policy model π_θ, maximizing the expected final reward. For each input c, consisting of a set of five sampled concepts and their corresponding explanation, the policy model is prompted to generate a group of G math problems $\{o_i\}_{i=1}^{G}$. We then evaluate each problem using our composite reward function Eq. (5) to obtain a scalar reward R_i. The advantage of the i-th problem is calculated by normalizing the token-level rewards:

$$\hat{A}_{i,t} = \frac{R_i - \text{mean}(\{R_j\}_{j=1}^{G})}{\text{std}(\{R_j\}_{j=1}^{G})}. \tag{6}$$

GRPO then maximizes the clipped objective:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{c,\{o_i\}\sim\pi_{\theta_{\text{old}}}} \left[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \mathcal{L}_{i,t} - \beta \mathbb{D}_{KL}(\pi_\theta \| \pi_{\text{ref}}) \right], \tag{7}$$

where

$$\mathcal{L}_{i,t} = \min\Big(r_{i,t}(\theta)\hat{A}_{i,t}, \text{clip}(r_{i,t}(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_{i,t}\Big), \tag{8}$$

$$r_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t}|c, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|c, o_{i,<t})}. \tag{9}$$

In these equations, π_θ is the policy before the update, π_{ref} is the reference policy (MathSmith-SFT). The hyperparameters ϵ and β control the clipping threshold and the KL penalty term, respectively. By maximizing this objective, we aim to iteratively improve the model's capability to generate well-structured and inherently challenging problems that demand complex multi-step reasoning.

3.4 Weakness-focused Improvement Pipeline

Given that each MathSmith-generated problem is explicitly linked to a concept set, we design a pipeline to enhance model performance on identified weaknesses.

Practice Set Q: We generate $|Q| = 1000$ problems using the MathSmith generator, covering a broad range of concepts. For each $q \in Q$, we sample 32 completions from Qwen3-30B-A3B, and select the most frequent answer as the reference

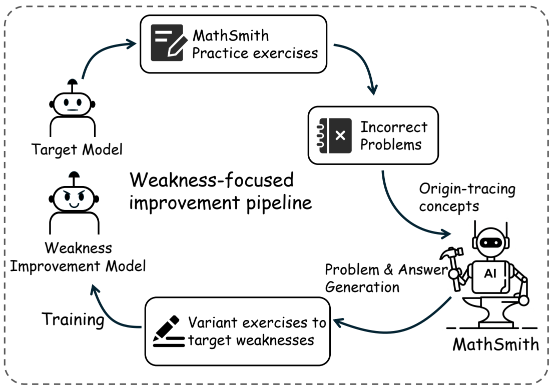
MathSmith Practice exercises
Incorrect Problems
Target Model
Weakness-focused improvement pipeline
Origin-tracing concepts
Weakness Improvement Model
Problem & Answer Generation
Training
Variant exercises to target weakness
MathSmith

Figure 3: The MathSmith weakness-focused improvement pipeline. Problems are traced to concept explanations, which serve as a basis for generating targeted variants to strengthen model weaknesses.

solution. After filtering, we retain 923 high-quality items.

Variant Set Q': For each $q \in Q$ with concept set c, we generate a variant problem $q' \sim \mathcal{G}(c)$, where $\mathcal{G}(c)$ denotes the MathSmith generator conditioned on concept c. The resulting set Q' forms a variant set aligned with concepts.

We fine-tune the model on a small subset of Q' using supervised learning. Let $\mathcal{M}_{\text{base}}$ denote the initial model and $\mathcal{M}_{\text{imp}}$ the fine-tuned model. We iteratively update $\mathcal{M}_{\text{base}} \to \mathcal{M}_{\text{imp}}$ until:

$$\text{Acc}_Q(\mathcal{M}_{\text{imp}}) \geq \tau, \quad (10)$$

where τ is a predefined accuracy threshold on Q. The resulting model $\mathcal{M}_{\text{imp}}$ is referred to as the improved model.

4 Experimental Setups

4.1 Datasets and Evaluation Metrics

To evaluate mathematical reasoning, we adopt five representative benchmarks categorized into two difficulty tiers: easy & medium and hard. The former includes (1) **GSM8K** (Cobbe et al., 2021), focusing on math word problems, and (2) **MATH** (Lightman et al., 2023), covering high school-level reasoning. The hard tier comprises three competition-style benchmarks: (3) **AIME 2024**[1], (4) **AIME 2025**[2], and (5) **OlympiadBench** (He et al., 2024), which require symbolic and multi-step reasoning to solve advanced mathematical problems.

[1] https://huggingface.co/datasets/Maxwell-Jia/AIME_2024

[2] https://huggingface.co/datasets/opencompass/AIME2025

4.2 Baseline

We compare with four representative math problem generation approaches. (1) **OpenMathInstruct** (Toshniwal et al., 2025) synthesizes new problems by prompting LLMs with in-context examples, relying on solution extrapolation without explicit control over difficulty. (2) **NuminaMath** (Li et al., 2024a) reformulates seed problems through CoT-guided sampling, aligning generated problems with existing math benchmarks. (3) **MetaMath** (Yu et al., 2024) increases problem diversity through structured rewriting techniques, including inversion, rephrasing, and reverse construction. (4) **PromptCOT** (Zhao et al., 2025) targets the difficulty level of the Olympiad by conditioning on the mathematical concepts sampled and guiding the generation through multistep rational planning. For each baseline, we sample 50K problems from its official dataset and regenerate solutions using a unified teacher model, Qwen3-30B-A3B, in non-thinking mode for short-CoT and thinking mode for long-CoT, ensuring consistency across methods. We evaluated all methods in both **short-CoT**, where reasoning traces are appended directly before the final answer (Wei et al., 2022), and **long-CoT**, where models are guided to perform detailed reasoning, including intermediate planning and possible self-reflection, before producing the final chain-of-thought (Guo et al., 2025). For evaluation, we use Qwen2.5-7B-Instruct and Qwen3-8B (non-thinking mode) in the short-CoT setting, and Qwen3-8B (thinking mode) and DeepSeek-R1-Distill-Qwen-7B in the long-CoT setting, representing state-of-the-art models within their parameter scale.

4.3 Implementation Details

For the MathSmith problem generation model based on Qwen3-8B, we conduct supervised fine-tuning on 8K cold-start samples generated by GPT-4o. We apply LoRA with rank 16 and train for 5 epochs on 8×H100 GPUs. This stage ensures the model learns the fundamental rationale structure and format patterns of problem synthesis, providing a stable foundation for subsequent reinforcement learning.

During reinforcement learning, we adopt the verl library (Sheng et al., 2024) for policy optimization. For the structural reward defined in Eq. (1), we set weights $\alpha_{\text{format}} = 0.7$ and $\alpha_{\text{step}} = 0.3$ to emphasize format consistency. For the reasoning

Table 1: Baseline performance under equal data and training conditions. MathSmith achieves consistently better generalization on challenging problems.

Models	Method	Eeay & Medium		Hard			Avg for Hard (Rel. Imp.)
		GSM8k	MATH-500	AIME2024	AIME2025	Olympiad	
Qwen-2.5-7B-Instruct short-CoT	-	**92.2**	72.2	16.7	6.7	38.6	20.7
	MetaMath	82.6	61.0	13.3	0.0	26.3	13.2 (-36.1%)
	NuminaMath-COT	87.2	73.4	**23.3**	3.3	36.9	21.2 (+2.4%)
	OpenMathInstruct-2	88.2	74.6	16.7	6.7	38.4	20.6 (-0.3%)
	PromptCOT	87.6	73.2	**23.3**	6.7	35.9	21.9 (+6.2%)
	MathSmith-HC	91.2	**75.2**	**23.3**	**10.0**	**39.9**	**24.4 (+18.1%)**
Qwen-3-8B short-CoT	-	**93.4**	82.8	30.0	16.7	51.0	32.6
	MetaMath	92.3	81.4	30.0	20.0	51.0	33.7 (+3.4%)
	NuminaMath-COT	92.6	83.2	**33.3**	13.3	50.6	32.4 (-0.5%)
	OpenMathInstruct-2	92.8	84.0	23.3	13.3	51.6	29.4 (-9.7%)
	PromptCOT	92.5	83.8	26.7	**23.3**	49.9	33.3 (+2.3%)
	MathSmith-HC	92.9	**84.4**	**33.3**	**23.3**	**53.1**	**36.6 (+12.3%)**
DS-R1-qwen2.5-7B long-CoT	-	89.3	88.6	43.3	36.7	52.4	44.1
	Numinamath-COT	93.3	91.0	46.7	30.0	53.4	43.4 (-1.7%)
	OpenMathInstruct-2	**93.9**	91.2	46.7	40.0	56.1	47.6 (+7.9%)
	PromptCOT	93.5	91.0	46.7	33.3	55.8	45.3 (+2.6%)
	MathSmith-HC	89.2	**91.6**	**53.3**	**43.3**	**56.5**	**51.0 (+15.6%)**
Qwen3-8B long-CoT	-	94.8	94.4	66.7	63.3	66.2	65.4
	Numinamath-COT	**95.5**	96.0	73.3	63.3	68.1	68.2 (+4.3%)
	OpenMathInstruct-2	**95.5**	95.8	70.0	60.0	67.4	65.8 (+0.6%)
	PromptCOT	95.1	95.4	73.3	63.3	67.1	67.9 (+3.8%)
	MathSmith-HC	95.1	**96.4**	**76.7**	**70.0**	**68.8**	**71.8 (+9.8%)**

reward in Eq. (4), we set $\beta_{\text{complexity}} = 0.7$ and $\beta_{\text{consistency}} = 0.3$ to focus on generating problems that require deeper reasoning. For both complexity and consistency estimation, we set the number of teacher samples $K = 5$. We train the model using the GRPO algorithm, executing both teacher inference and MathSmith policy updates on 20×H100 GPUs. The final model is selected at step 100, where performance is near convergence.

To ensure reproducibility, we adopt full supervised fine-tuning for evaluation baselines using the LlamaFactory (Zheng et al., 2024) training scripts. All models are trained for 5 epochs with a learning rate of $1e{-}5$, and evaluated on 8×H100 GPUs under consistent experimental settings.

5 Results and Analysis

5.1 Overall Performance

The overall results are presented in Table 1, revealing several key findings. Our method achieves state-of-the-art performance across multiple benchmarks, demonstrating the effectiveness of the synthesized data. We divide evaluation problems into two difficulty levels: *easy & medium* and *hard*. As difficulty increases, our method yields notably stronger results, especially under the long-CoT setting, where MathSmith shows significantly larger improvements over baselines, indicating its ability to elicit more complex reasoning. Unlike prior methods that extract concepts from real math problems, MathSmith samples concept sets entirely at random, but still generalizes well to challenging real-world tasks. On certain benchmarks such as GSM8K, however, performance occasionally falls below that of the base model. This trend, also observed in other baselines, probably stems from the nature of GSM8K as a word problem data set, which differs in format and complexity from the competition-style problems emphasized during synthesis. In such cases, excessive reasoning may even hinder performance by introducing unnecessary complexity.

5.2 Effect of Dataset Scaling

We evaluate the scalability of our method using the Olympiad benchmark, selected for its high difficulty, diverse problem types, and sufficient data volume, making it a more reliable indicator of performance. Qwen3-30B-A3B serves as the teacher model to synthesize all training data, while Qwen3-8B is used as the base model. As shown in Figure 4, MathSmith-HC consistently outperforms strong baselines (NuminaMath-COT and OpenMathInstruct-2) across training sizes from 50K to 200K, with the performance gap widening as the data volume increases.

5.3 Effect of Model Scaling

We analyze the impact of model scale using the Qwen3 series, trained on a fixed 50K dataset synthesized by Qwen3-30B-A3B. As shown in Fig-

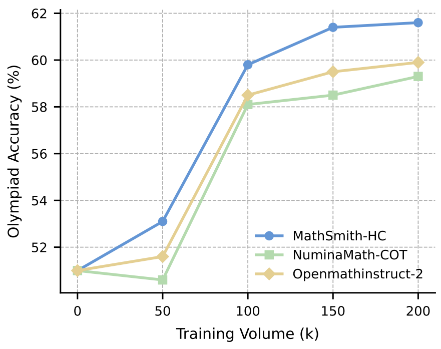

Figure 4: Performance on the Olympiad benchmark under varying training data volumes (50K–200K). MathSmith-HC scales more effectively than baseline methods.

ure 5, MathSmith-HC performs slightly worse on smaller models (e.g., 1.7B, 4B), likely due to limited capacity to learn from complex problems. As model size increases, our method consistently outperforms baselines, suggesting that larger models benefit more from high-difficulty synthetic data by acquiring deeper reasoning abilities.

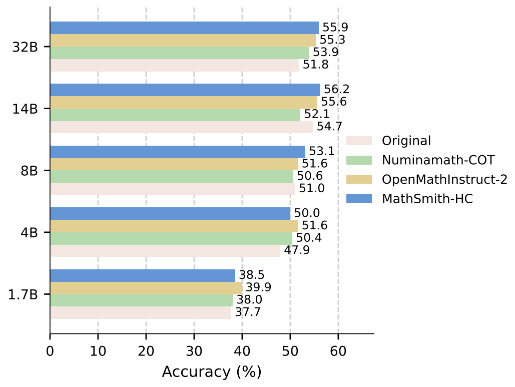

Figure 5: Accuracy on the Olympiad benchmark across Qwen3 model sizes series. MathSmith-HC yields greater gains with larger models.

5.4 Analysis of Problem Difficulty

We assess the relative difficulty of the problems by measuring the average token length of reasoning traces generated in the thinking mode of Qwen3-30B-A3B. Following the assumption that more complex problems induce longer reasoning sequences, we collect 50 problems from each major open-source math dataset, including both training and evaluation sets, and use the model to solve them in long-CoT mode. For datasets containing fewer than 50 problems, all available instances are included. As shown in Figure 2, MathSmith-SFT already produces challenging problems, indicating the effectiveness of synthesizing questions directly from difficult concepts and predefined difficulty strategies. Notably, MathSmith-HC and MathSmith-Hard result in the longest reasoning traces across all datasets, suggesting that our reinforcement learning stage further enhances problem complexity and encourages deeper reasoning behavior.

5.5 Impact of Weakness-Focused Problem Generation

To evaluate the effectiveness of weakness-focused improvement, we first use MathSmith-HC to generate 10 concept-guided variants for each question in the Practice Set, targeting concepts associated with incorrect predictions. We then generate solutions to these variant problems using Qwen2.5-Math-7B, and evaluate the accuracy of a base model (Qwen2.5-Math-1.5B) on both the original and variant problems. As shown in Table 2, the weakness-focused variants yield consistent accuracy improvements over the random sampling baseline with the same number of generated problems (Epoch 1), particularly on harder problems. Furthermore, we observe that accuracy gains on the Practice Set correlate with improved generalization to other math benchmarks, suggesting that MathSmith-generated problems possess strong transferability.

Table 2: Effect of weakness-focused problem generation vs. random sampling

Sample Method	Ave. (Easy & Medium)	Ave. Hard	Acc on Practice
Original	38.2	14.5	23.6
Weakness-focused Epoch 1	69.9	18.8	33.1
Weakness-focused Epoch 2	77.3	21.5	34.6
Weakness-focused Epoch 3	77.6	21.6	34.7
Random Sampling	69.4	15.6	30.0

5.6 Ablation Analysis

We perform ablation studies to assess the effects of different training stages in MathSmith. To quantify the usability of the problem, we define an Available Ratio, the percentage of generated problems that are correctly formatted and solvable by the teacher model with a valid answer.

After generating 50k problems for each variant, we fine-tune Qwen3-8B using the same supervised procedure. As shown in Table 3, MathSmith-HC achieves the highest Available Ratio while maintaining strong performance on hard problems. Although MathSmith-Hard shows slightly better accuracy, its lower usability makes it less suitable for large-scale synthesis. We further evaluate the impact of different teacher models. Using Qwen2.5-32B to generate solutions under the Short-CoT setting, we fine-tune Qwen3-8B on 40K training samples from the questions of each method. Table 4 shows that MathSmith-HC remains consistently superior across all benchmarks.

Table 3: Performance of different training stages. MathSmith-HC offers the best balance of accuracy and usability.

Training Stage	Ave. (Easy & Medium)	Ave. Hard	Available Ratio
MathSmith-SFT	87.7	30.3	71.50%
MathSmith-Hard	**89.25**	**36.6**	84.92%
MathSmith-HC	88.65	**36.6**	**95.38%**

Table 4: Effect of different teacher models. MathSmith-HC outperforms all baselines with Qwen2.5-32B as the solver.

Method	GSM8k	MATH-500	AIME2024	Olympiad
Numinamath	92.1	76.0	**23.3**	**43.8**
OpenMathInstruct	91.3	78.2	13.3	42.3
PromptCOT	90.9	73.8	16.7	41.1
MathSmith-HC	**93.1**	**78.8**	**23.3**	**43.8**

6 Conclusion

We introduced MathSmith, a framework for synthesizing high-difficulty mathematical problems from randomly sampled concept–explanation pairs, guided by predefined difficulty strategies and optimized via reinforcement learning for structural validity, reasoning depth, and answer consistency. By encouraging longer reasoning traces as a heuristic for problem complexity, MathSmith produces diverse and challenging problems that substantially improve LLM reasoning on benchmarks such as AIME 2024, AIME 2025, and Olympiad. These results highlight the potential of scalable synthetic data to drive deeper reasoning capabilities in large models. Moreover, the framework demonstrates strong generalization, showing that fully synthetic problems can effectively complement limited human-authored datasets in advancing mathematical reasoning. Future work will focus on refining difficulty estimation, expanding domain coverage, and exploring adaptive generation strategies to construct richer synthetic datasets to advance high-level mathematical reasoning.

References

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen Marcus McAleer, Albert Q. Jiang, Jia Deng, Stella Biderman, and Sean Welleck. 2024. Llemma: An open language model for mathematics. In *The Twelfth International Conference on Learning Representations*.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. 2023. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Transactions on Machine Learning Research*.

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*.

Yuyang Ding, Xinyu Shi, Xiaobo Liang, Juntao Li, Qiaoming Zhu, and Min Zhang. 2024. Unleashing reasoning capability of llms via scalable question synthesis from scratch. *arXiv preprint arXiv:2410.18693*.

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*.

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.

Kaixuan Huang, Jiacheng Guo, Zihao Li, Xiang Ji, Jiawei Ge, Wenzhe Li, Yingqing Guo, Tianle Cai, Hui Yuan, Runzhe Wang, Yue Wu, Ming Yin, Shange Tang, Yangsibo Huang, Chi Jin, Xinyun Chen, Chiyuan Zhang, and Mengdi Wang. 2025a. MATH-perturb: Benchmarking LLMs' math reasoning abilities against hard perturbations. In *Forty-second International Conference on Machine Learning*.

Yiming Huang, Xiao Liu, Yeyun Gong, Zhibin Gou, Yelong Shen, Nan Duan, and Weizhu Chen. 2025b. Key-point-driven data synthesis with its enhancement on mathematical reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 23, pages 24176–24184.

Shima Imani, Liang Du, and Harsh Shrivastava. 2023. Mathprompter: Mathematical reasoning using large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 37–42.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, and 1 others. 2024a. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13:9.

Qintong Li, Leyang Cui, Xueliang Zhao, Lingpeng Kong, and Wei Bi. 2024b. GSM-plus: A comprehensive benchmark for evaluating the robustness of LLMs as mathematical problem solvers. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2961–2984, Bangkok, Thailand. Association for Computational Linguistics.

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*.

Haoxiong Liu, Yifan Zhang, Yifan Luo, and Andrew C Yao. 2025. Augmenting math word problems via iterative question composing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24605–24613.

Zimu Lu, Aojun Zhou, Houxing Ren, Ke Wang, Weikang Shi, Junting Pan, Mingjie Zhan, and Hongsheng Li. 2024. Mathgenie: Generating synthetic data with question back-translation for enhancing mathematical reasoning of llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2732–2747.

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*.

OpenAI. 2025. Openai o3 and o4-mini system card. `https://openai.com/index/o3-o4-mini-system-card/`. Accessed: 2025-05-21.

PlanetMath Community. 2024. Planetmath: An online mathematics encyclopedia. Accessed: 2025-04-25.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

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

Richard Sutton. 2019. The bitter lesson. *Incomplete Ideas (blog)*, 13(1):38.

Zhengyang Tang, Xingxing Zhang, Benyou Wang, and Furu Wei. 2024. Mathscale: Scaling instruction tuning for mathematical reasoning. In *International Conference on Machine Learning*, pages 47885–47900. PMLR.

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.

Shubham Toshniwal, Wei Du, Ivan Moshkov, Branislav Kisacanin, Alexan Ayrapetyan, and Igor Gitman. 2025. Openmathinstruct-2: Accelerating AI for math with massive open-source instruction data. In *The Thirteenth International Conference on Learning Representations*.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, and 1 others. 2025. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*.

Zengzhi Wang, Xuefeng Li, Rui Xia, and Pengfei Liu. 2024. Mathpile: A billion-token-scale pretraining corpus for math. *Advances in Neural Information Processing Systems*, 37:25426–25468.

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.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng YU, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations*.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2024. MAmmoTH: Building math generalist models through hybrid instruction tuning. In *The Twelfth International Conference on Learning Representations*.

Xueliang Zhao, Wei Wu, Jian Guan, and Lingpeng Kong. 2025. Promptcot: Synthesizing olympiad-level problems for mathematical reasoning in large language models. *arXiv preprint arXiv:2503.02324*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.

Kun Zhou, Beichen Zhang, Zhipeng Chen, Xin Zhao, Jing Sha, Zhichao Sheng, Shijin Wang, Ji-Rong Wen, and 1 others. 2024. Jiuzhang3. 0: Efficiently improving mathematical reasoning by training small data synthesis models. *Advances in Neural Information Processing Systems*, 37:1854–1889.

A Concept and Explanation Examples

We collect a set of approximately 11k concept–explanation pairs from PlanetMath, following the extraction instruction described in Appendix C.1 and using the GPT-4o model. Each entry contains the mathematical concept, its concise explanation, and the associated categories. Table 5 presents several examples.

B Training Details

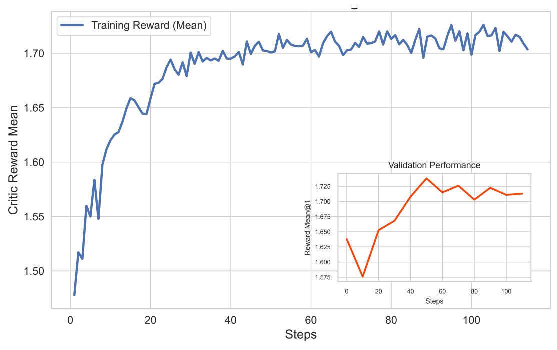

Figure 6: Reward trend during GRPO training of **MathSmith-HC**. The model steadily improves and reaches convergence after about 100 steps.

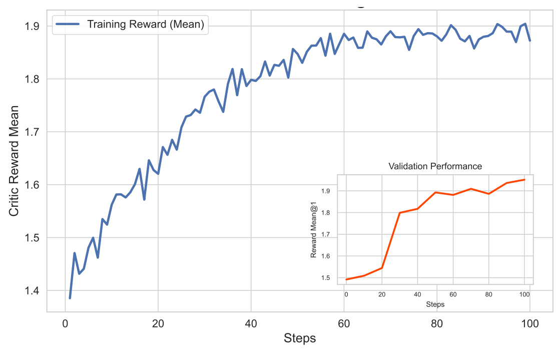

Figure 7: Reward trend during GRPO training of **MathSmith-Hard**, showing similar convergence behavior after roughly 100 steps.

B.1 Reinforcement Learning Training Process

We train **MathSmith-HC** and **MathSmith-Hard** using the GRPO algorithm within the verL framework. The reinforcement learning process optimizes the model through multi-objective rewards that balance structural correctness, reasoning depth, and solution consistency. The reward curves during training are shown in Fig. 6 and Fig. 7. Both models exhibit a steady improvement in the early stages and achieve approximate convergence after around 100 steps, indicating stable reward optimization.

C Instruction Details

C.1 Instruction for Collect Concept and Explanation

Concept and Explanation Extraction Prompt

You are a mathematical assistant.
Given the following markdown content from a mathematical article, extract key concepts and their explanations.

Instructions:

1. If the content **does NOT** define or discuss a math concept, respond with exactly: `Not a math concept`
2. Otherwise, identify all important math concepts mentioned.
   - For each concept, ensure it is explicitly mentioned or defined in the given markdown article, and provide a brief explanation based strictly on the article content.
   - The explanation should be concise and relevant to the concept.
   - Concept and Explanation must be written in LaTeX-compatible format, using proper mathematical notation where appropriate.
   - Additionally, list 1–3 broad mathematical categories related to the content.
3. For each concept, output a JSON format: ["Concept": "...", "Explanation": "...", "Categories": ["...", "..."], ...]

Input: Markdown Math Content (from PlanetMath)

C.2 Instruction for Cold Start Data Generation

C.3 Instruction for Quetion and Solution Generation

D Case Study

We present illustrative examples of the problem generation processes for MathSmith-HC and MathSmith-Hard. MathSmith-HC offers more detailed rationale steps, enhancing interpretability, whereas MathSmith-Hard generates concise rationales that still yield readable problems.

Cold Start Data Generation Prompt with GPT-4o

Given the Concepts and Explanations along with the instructions below, develop a **single challenging mathematics problem** suitable for advanced Olympiads.

Instructions:
A. Select Relevant Concepts

1. Carefully analyze the provided concepts and their explanations.
2. Based on this analysis, choose a suitable subset of concepts (at least two; more are welcome if appropriate) that can be naturally and meaningfully integrated into a cohesive, well-focused mathematics problem.

B. Problem Difficulty Rules Include **at least two** of the following features:

- **Multi-step Reasoning:** Requires multiple sequential logical steps.
- **Cross-topic Integration:** Combines distinct mathematical topics.
- **Implicit or Reverse Logic:** Includes hidden conditions or reverse deduction.
- **Distractors:** Contains misleading or extraneous conditions.
- **Abstract Modeling:** Translates complex scenarios into mathematical form.
- **Multiple Solution Paths:** Allows various non-trivial solving methods.
- **Advanced Manipulation:** Necessitates sophisticated algebraic or geometric transformations.
- **Extreme Conditions:** Focuses on limits or boundary values.
- **Non-standard Representation:** Uses unconventional presentation of familiar concepts.

C. Problem Format Constraints

1. Generate **only one single problem**; no sub-questions or multiple parts.
2. The problem must have a **single, unified solving goal**, clearly answerable by a focused line of reasoning. Avoid phrases like "first..., then..." or numbering subtasks.
3. The problem must admit a **well-defined, verifiable solution**, such as a number, expression, or equation.
4. **Do not include proof-based phrasing** (e.g., "prove that," "show that"). Ask only for specific, verifiable results.
5. Avoid secondary tasks (e.g., "construct an example," "justify your answer") unless construction is the core objective. **Keep the question focused**.

Output Format:
A. Rationale Explain your thought process step-by-step:

1. Analyze and understand the given concepts in depth.
2. Select a suitable combination of concepts that can be meaningfully integrated.
3. Explain how these concepts are woven into a unified mathematical scenario.
4. Identify and describe the difficulty features used.
5. Formulate the final problem statement clearly and concisely.

Wrap your rationale in:

```
<rationale>
[Your construction thought process goes here]
</rationale>
```

B. Problem Present the final problem inside:

```
<problem>
[Your final math problem goes here]
</problem>
```

Given Concept and Explanation: `{CONCEPT_AND_EXPLANATION}`

Concept-Based Question and Solution Generation Prompt

Given the Concepts and Explanations along with the instructions below, develop a **single challenging mathematics problem** suitable for advanced Olympiads.

Instructions:
A. Select Relevant Concepts

1. Carefully analyze the provided concepts and their explanations.
2. Choose a subset of concepts that can be naturally and meaningfully integrated into a single, well-focused problem.

B. Problem Difficulty Rules

- The problem should require deep insight and non-obvious reasoning.
- Include **at least two** of the following difficulty features:
  - **Multi-step Reasoning**: Requires multiple sequential logical steps.
  - **Cross-topic Integration**: Combines distinct mathematical topics.
  - **Implicit or Reverse Logic**: Involves hidden constraints or backward deduction.
  - **Distractors**: Includes misleading or irrelevant elements.
  - **Abstract Modeling**: Converts complex real-world scenarios into mathematical expressions.
  - **Multiple Solution Paths**: Allows for multiple viable solving strategies.
  - **Advanced Manipulation**: Requires sophisticated algebraic or geometric techniques.
  - **Extreme Conditions**: Focuses on limits, edge cases, or extremal values.
  - **Non-standard Representation**: Presents concepts in unconventional forms.

C. Output Format Strictly format your response as:

```
<rationale>
Step-by-step reasoning describing:
  - How the selected concepts are connected;
  - How they contribute to the structure and difficulty of the problem;
  - Which difficulty features (from B.2) are used, and how they are embedded.
</rationale>

<problem>
[Write a single self-contained, high-level Olympiad mathematics problem here.]
</problem>
```

Given Concept and Explanation: `{Sampled_Concept_and_Explanation_Set}`

Solution Prompt Instruction:

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

Sampled Concept and Explanation:

1.**Quadratic Resolvent**: The quadratic resolvent of the cubic equation $y^3 + py + q = 0$ is the auxiliary equation $z^2 + qz - \left(\frac{p}{3}\right)^3 = 0$, which helps to find roots $z_1 = u^3$ and $z_2 = v^3$, where $y = u + v$ satisfies the original cubic equation.

2.**GCD condition**: The condition $\gcd(a, b) = 1$ can be added to the conclusion of Dirichlet's approximation theorem.

3.**Partial Map**: A partial map on a set X is an application defined from a subset of X into X, denoted as $\mathfrak{F}(X)$.

4.**Minimum Element**: A minimum element of a set S is the smallest element in S. If S is a finite subset of $\mathbb{R}$, then it contains a minimum element $s \in S$.

5.**Lattices with Operators**: Lattices that include additional operations beyond the standard meet and join.

Rationale:

Step 1: Analyze and understand the given concepts in depth.

The concepts provided span algebra (quadratic resolvent of cubic equations), number theory (GCD conditions), set theory (partial maps and minimum elements), and abstract algebra (lattices with operators). Among these, quadratic resolvent and minimum element naturally lend themselves to integration with algebraic structures, while GCD conditions and partial maps introduce constraints or mappings that can add complexity.

Step 2: Select a suitable combination of concepts.

I chose quadratic resolvent and minimum element as the primary concepts, integrating them with constraints inspired by GCD conditions. This combination allows for a rich interplay between algebraic manipulation and optimization under specific conditions.

Step 3: Weave concepts into a unified scenario.

The problem is constructed around finding roots of a cubic equation under constraints tied to GCD conditions and minimum elements. By embedding implicit conditions and requiring advanced algebraic manipulation, the problem challenges competitors to navigate multiple layers of reasoning.

Step 4: Incorporate difficulty features.

–**Multi-step reasoning**: Solving the cubic equation requires sequential steps involving resolvent construction, GCD constraints, and optimization.

–**Implicit logic**: Hidden conditions tied to GCD and minimum elements force competitors to deduce constraints indirectly.

–**Advanced manipulation**: Algebraic transformations of the cubic resolvent and interaction with GCD constraints demand sophisticated handling.

–**Extreme conditions**: The problem focuses on minimizing a specific expression under strict constraints, pushing competitors to consider boundary cases.

Step 5: Formulate the final problem statement.

The problem is presented concisely, ensuring competitors focus on a single, unified goal without auxiliary subtasks.

Final Problem:

Let $y^3 - 3py + q = 0$ be a cubic equation with integer coefficients p and q, where $p > 0$ and $q < 0$. The quadratic resolvent of this equation is given by $z^2 + qz - \left(\frac{p}{3}\right)^3 = 0$, with roots z_1 and z_2. Suppose $y = u + v$ satisfies the original cubic equation, where $u^3 = z_1$ and $v^3 = z_2$. Additionally, let u and v satisfy the following conditions: 1. $\gcd(u, v) = 1$, 2. $u + v$ is the minimum positive real root of $y^3 - 3py + q = 0$. If $p = 9$ and $q = -27$, determine the exact value of $u^2 + v^2$.

Figure 8: Case study of MathSmith-HC problem generation.

Sampled Concept and Explanation:

1. **Analytic Polyhedron:** An analytic polyhedron is defined as the set $\Omega := \{z \in W \mid |f_j(z)| < 1,\ j = 1, \dots, k\}$ which is relatively compact in an open set W of a domain $G \subset \mathbb{C}^n$, where $f_1, \dots, f_k$ are holomorphic functions.
2. **Modulus of a Complex Number:** The modulus of a complex number is the distance from the origin in the complex plane, denoted as r.
3. **Infinitesimal:** An infinitesimal is a number that is greater than zero but smaller than any positive real number, defined such that for all positive integers n, $|\varepsilon| < \frac{1}{n}$.
4. **Graph Isomorphism:** Graph isomorphism refers to a one-to-one correspondence between the vertex sets of two graphs that preserves the edge connectivity.
5. **n-th Derivative of a Determinant:** The n-th derivative of the determinant of a matrix A, where A is a $d \times d$ matrix with entries as functions of t, is expressed using a sum over multinomial coefficients and permutations, involving the derivatives of the matrix entries.

Rationale:
Step 1. Analyze the given concepts:
The concepts involve analytic geometry, complex numbers, infinitesimals, graph theory, and matrix calculus. These are diverse, allowing for cross-topic integration.

Step 2. Select suitable concepts:
Combine Modulus of a Complex Number and n-th derivative of a determinant, as they can naturally interact in a problem involving complex-valued matrices and constraints.

Step 3. Integration:
Construct a scenario involving a complex matrix whose entries are constrained by modulus conditions, requiring advanced manipulation of determinants and derivatives.

Step 4. Difficulty features:
Incorporate multi-step reasoning, cross-topic integration, and implicit logic by embedding hidden modulus constraints and derivative conditions.

Step 5. Formulate problem:
Design a focused problem requiring determinant analysis under complex modulus restrictions and derivative evaluation.

Final Problem:
Let $A(t)$ be a 3×3 complex matrix whose entries are given by

$$a_{ij}(t) = t^{i+j-2} + \frac{\omega_{ij}}{1 + |t|^2},$$

where ω_{ij} are distinct complex numbers with $|\omega_{ij}| = 1$ for all $i, j \in \{1, 2, 3\}$, and $t \in \mathbb{C}$ satisfies $|t| < 1$.
Define $D(t) = \det(A(t))$, the determinant of $A(t)$. If the second derivative $D''(t)$ vanishes at exactly two distinct values $t_1, t_2 \in \mathbb{C}$ with $|t_1| < |t_2| < 1$, determine the sum

$$S = |D(t_1)| + |D(t_2)| + \sum_{i,j=1}^{3} |\omega_{ij}|^3.$$

Figure 9: Case study of MathSmith-Hard problem generation.

Table 5: Examples of Concept and Explanation pairs.

Concept	Explanation	Categories
Von Neumann ordinals	Von Neumann ordinals are defined as sets of all natural numbers less than a given number, establishing a foundation for defining natural numbers through set theory.	Set Theory; Ordinal Numbers; Foundations of Mathematics
Dynkin system	A Dynkin system on a set Ω is a collection $\mathcal{D} \subset \mathcal{P}(\Omega)$ satisfying: 1. $\Omega \in \mathcal{D}$; 2. if $A, B \in \mathcal{D}$ and $A \subset B$, then $B \setminus A \in \mathcal{D}$; 3. if $A_n \in \mathcal{D}$ with $A_n \subset A_{n+1}$ for $n \geq 1$, then $\bigcup_{k=1}^{\infty} A_k \in \mathcal{D}$.	Set Theory; Measure Theory; Probability Theory
Inverse image of a function	For a function f and a subset $B \subseteq Y$, the inverse image of B under the restriction $f\|_A$ is $$(f\|_A)^{-1}(B) = A \cap f^{-1}(B).$$	Functions; Set Theory; Mathematical Analysis