MathAgent: Adversarial Evolution of Constraint Graphs for Mathematical Reasoning Data Synthesis

Zixiong Yu[1,2*], Jun Rao[3*], Guhan Chen[2], Songtao Tian[2], Bohan Li[2,4], Jiansheng Wei[1], Min Zhang[3], and Xiaojun Meng[1†]
[1]Huawei Large Model Data Technology Lab [2]Tsinghua University
[3]Harbin Institute of Technology, Shenzhen [4]Kyoto University
{yuzx19,tiansongtao.2020,libh19}@tsinghua.org.cn
rao7jun@gmail.com chen-gh23@mails.tsinghua.edu.cn
zhangmin2021@hit.edu.cn {weijiansheng,xiaojun.meng}@huawei.com

Abstract

Synthesizing high-quality mathematical reasoning data without human priors remains a significant challenge. Current approaches typically rely on seed data mutation or simple prompt engineering, often suffering from mode collapse and limited logical complexity. This paper proposes a hierarchical synthesis framework that formulates data synthesis as an unsupervised optimization problem over a constraint graph followed by semantic instantiation, rather than treating it as a direct text generation task. We introduce a *Legislator-Executor* paradigm: The *Legislator* adversarially evolves structured generation blueprints encoding the constraints of the problem, while the *Executor* instantiates these specifications into diverse natural language scenarios. This decoupling of skeleton design from linguistic realization enables a prioritized focus on constructing complex and diverse logical structures, thereby guiding high-quality data synthesis. Experiments conducted on a total of 10 models across the Qwen, Llama, Mistral, and Gemma series demonstrate that our method achieves notable results: models fine-tuned on 1K synthesized samples outperform widely-used datasets of comparable scale (LIMO, s1K) across eight mathematical benchmarks, exhibiting superior out-of-distribution generalization.

1 Introduction

In recent years, Large Language Models (LLMs; Vaswani et al., 2017; Brown et al., 2020; Zhao et al., 2023) have become a central pillar of modern artificial intelligence (AI). Although theoretical understanding of their underlying mechanisms remains relatively limited (Jacot et al., 2018; Li et al., 2024b; Yu et al., 2025), LLMs have demonstrated strong reasoning abilities in practice and achieved remarkable success on complex tasks (Wei et al., 2022a, 2026). These capabilities have in turn enabled rapid progress in a wide range of areas, such as embodied AI (Driess et al., 2023; Zeng et al., 2026) and agentic systems with tool-use capabilities (Schick et al., 2023; Xu et al., 2026).

This progress has been driven by multiple factors, including the scaling of model parameters and training data (Kaplan et al., 2020; Hoffmann et al., 2022), reasoning-oriented techniques such as chain-of-thought (CoT; Wei et al., 2022b; Jiang et al., 2025; Zeng et al., 2025a), and, equally importantly, the quality of training data (Zhou et al., 2023; Ye et al., 2025; Zhao et al., 2026). However, as high-quality human-generated corpora become increasingly difficult to scale, the field faces a growing data bottleneck (Villalobos et al., 2024). Consequently, synthetic data generation, which uses generative models to produce training samples, has emerged as a major research direction (Honovich et al., 2023; Ke et al., 2025; Zhang et al., 2026).

Current synthesis paradigms primarily fall into two categories: (i) *Seed-based* methods, such as Self-Instruct (Wang et al., 2023), expand upon human-curated seeds. While effective, their diversity is inherently upper-bounded by the semantic span of the initial seeds. (ii) *Zero-shot* methods like Magpie (Xu et al., 2025) probe model distributions directly but often suffer from mode collapse and logical hallucinations due to the lack of structural guidance (Shumailov et al., 2023). We argue that by framing data synthesis as a mere text generation task rather than a structured optimization problem, current methods often confine models to superficial narrative imitation without mastering core reasoning capabilities (Gudibande et al., 2023).

To address this, we propose a hierarchical synthesis framework anchored by a bi-level *Legislator-Executor* paradigm, which effectively decouples structural specifications from their textual instantiation. By pre-establishing high-level task blueprints that incorporate logical relations and constraints,

[*]Equal Contribution.
[†]Corresponding Author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 28298–28321

the framework more effectively guides the generation of high-quality mathematical problems. We instantiate this architecture as MathAgent, where the Legislator (meta-level) adversarially optimizes the combination of problem elements over a constraint graph, while the Executor (base-level) transforms these abstract blueprints into natural language.

This decoupling mechanism enables a prioritized focus on orchestrating structural diversity and complexity. Through iterative adversarial evolution, MathAgent continuously explores the underlying structural space, thereby progressively pushing the frontiers of model generation capabilities. Compared to direct probing methods confined by high-frequency patterns and seed dataset augmentation methods limited by initial semantic ranges, our framework excels at capturing scarce data characterized by high difficulty and quality. Consequently, relying solely on basic conceptual primitives rather than seed data, MathAgent synthesizes corpora with high structural complexity and rich diversity, while flexibly regulating the complexity of data distributions through an adaptive early-stop iteration mechanism.

Our contributions are summarized as follows:

- We propose the Legislator-Executor paradigm, a hierarchical synthesis framework that decouples task specification from textual realization to facilitate the guided synthesis of reasoning data.
- We introduce a constraint-graph-based adversarial evolutionary mechanism to explore structural spaces, generating high-difficulty, high-quality problems often absent in standard datasets.
- Extensive experiments demonstrate that models fine-tuned on 1K MathAgent samples outperform mainstream datasets of comparable scale (LIMO & s1K) across eight benchmarks, exhibiting superior out-of-distribution generalization.

2 Related Work

Data Synthesis A prevalent paradigm in data synthesis involves the iterative expansion of seed examples. Methods like Self-Instruct (Wang et al., 2023) and WizardMath (Luo et al., 2025) utilize evolution strategies to amplify task complexity, while MathGenie (Lu et al., 2024b) employs a backward mechanism, augmenting seed solutions to back-translate new questions. Nevertheless, these approaches are constrained by the "semantic radius" of their initial seeds, often failing to explore the unknown regions of the problem space. Similarly, for preference-pair data, methods such as SeaPO (Rao et al., 2025a) construct preference pairs by generating contrastive responses based on existing answers; however, such approaches typically place higher demands on the fine-grained and controllable editing capabilities of LLMs (Zeng et al., 2025c). To avoid relying on seed datasets, zero-shot methods such as Magpie (Xu et al., 2025) and schema-driven frameworks like Condor (Maosong-cao et al., 2025) attempt to synthesize data from scratch. While successful in high-resource domains, they lack the structural incentives to discover samples in the long-tail distribution, where complex reasoning capabilities are often forged.

Multi-Agent and Adversarial Generation The deployment of LLMs has progressed from single-turn prompting to complex Multi-Agent Systems. Frameworks such as CAMEL (Li et al., 2023) and MetaGPT (Hong et al., 2024) demonstrate that role-playing agents can effectively decompose tasks through cooperation, while AgentDropout (Wang et al., 2025) further improves efficiency and coordination via dynamic agent elimination. In the realm of data synthesis, MATRIX (Tang et al., 2025) utilizes multi-agent simulation to construct virtual societies, generating instruction data grounded in realistic social scenarios. Concurrently, multi-agent debate (Du et al., 2024) has emerged as a pivotal mechanism for enhancing reasoning reliability. For instance, Debate4MATH (Zhang and Xiong, 2025) employs fine-grained step verification to rectify logical errors, while Liang et al. (2024) leverage debate to stimulate divergent thinking for higher-quality problem solving. Our work adapts adversarial dynamics for data synthesis, aiming to drive continuous evolution of the training data distribution and explore the generation of complex samples.

3 Method

We propose a hierarchical synthesis framework to synthesize mathematical data by optimizing the constraint graph of problem structures. Specifically instantiated as MathAgent, our approach diverges from standard methods that operate directly in the token space by decoupling the synthesis process into two distinct phases: (1) Structural Evolution (Meta-level), governed by a *Legislator* agent that optimizes a constraint graph (acting as the synthesis blueprint); and (2) Semantic Instantiation (Base-level), conducted by an *Executor* that grounds the graph structure into natural language scenarios.

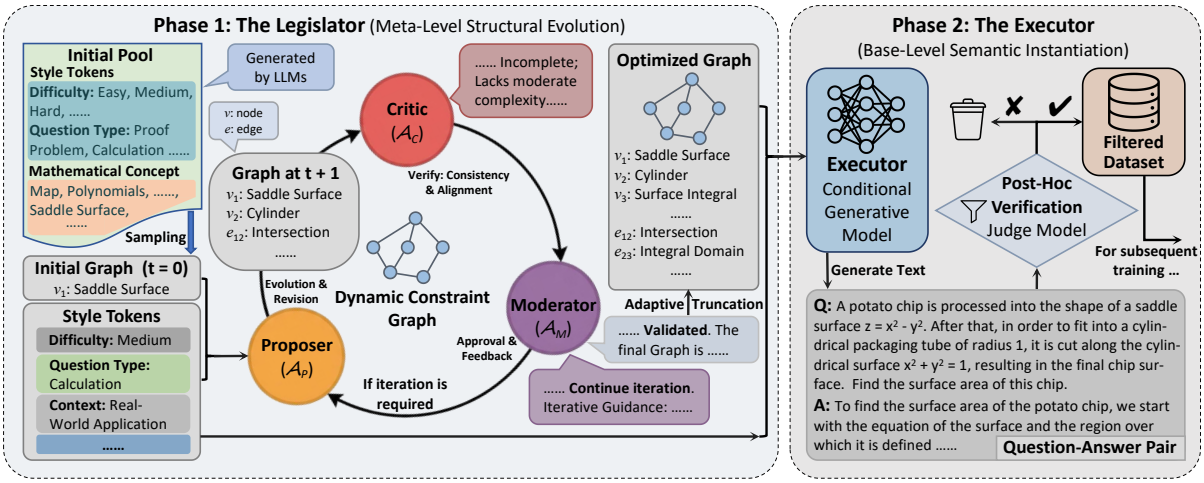

Figure 1: **The MathAgent Framework.** The framework consists of two decoupled phases: (1) Meta-Level Structural Evolution, where a tri-agent Legislator system (Proposer, Critic, and Moderator) iteratively optimizes a Constraint Graph $\mathcal{G}$ based on Style Tokens $\mathcal{S}$; and (2) Base-Level Semantic Instantiation, where the Executor grounds the optimized structural blueprint into natural language problems Q and reasoning chains A.

3.1 Problem Formulation

We formulate the skeleton of a mathematical problem as a structure comprising a **Constraint Graph** $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and **Style Tokens** $\mathcal{S}$. Specifically:

- **Nodes** ($\mathcal{V}$) represent mathematical concepts.
- **Edges** ($\mathcal{E}$) represent logical relations.
- **Style Tokens** ($\mathcal{S}$) control global attributes (e.g., problem category or difficulty level).

Serving as a blueprint for problem synthesis, this graph facilitates automated structural evolution, enabling the targeted generation of high-complexity and richly diverse reasoning data.

Formally, our objective is to explore the space of graph topologies, optimizing the complexity while ensuring strict solvability (conditioned on $\mathcal{S}$):

$$\mathcal{G}^* = \arg\max_{\mathcal{G}\in\mathbb{G}} \mathcal{H}(\mathcal{G}) \quad \text{s.t.} \quad \mathbb{I}_{\text{valid}}(\mathcal{G} \mid \mathcal{S}) = 1,$$

where $\mathbb{G}$ is the search space, $\mathcal{H}(\cdot)$ estimates the complexity, and $\mathbb{I}_{\text{valid}}(\cdot)$ is a binary validity indicator. Note that while the optimization seeks to push the reasoning frontier (finding $\mathcal{G}^*$), the evolutionary trajectory yields a diverse curriculum of graphs. The resulting tuple $(\mathcal{G}^*, \mathcal{S})$ is then passed to the Executor for textual realization.

3.2 Phase 1: The Legislator (Meta-Level)

To address the optimization objective, we design the Legislator as a tri-agent evolutionary system. Instead of directly manipulating text, the system iteratively optimizes the dynamic constraint graph $\mathcal{G}_t$ through inter-agent collaboration under given style token $\mathcal{S}$ conditions. The evolutionary process is jointly driven by three distinct roles:

Proposer ($\mathcal{A}_P$) As the driving engine of structural evolution, the proposer $\mathcal{A}_P$ optimizes $\mathcal{G}_t$ to $\mathcal{G}_{t+1}$ guided by feedback from previous iterations. It resolves logical contradictions while ensuring the graph's characteristics align with the structural specifications defined in style tokens $\mathcal{S}$. In particular, if the current structure has not reached the target complexity required by style tokens $\mathcal{S}$, the proposer $\mathcal{A}_P$ proactively expands knowledge nodes or strengthens constraints to enhance structural depth.

Critic ($\mathcal{A}_C$) As a key component of adversarial evolution, the critic $\mathcal{A}_C$ scrutinizes $\mathcal{G}_{t+1}$ across three dimensions based on the style tokens $\mathcal{S}$: (1) *Internal Consistency*: it verifies whether logical contradictions exist in the current graph; (2) *Specification Alignment*: it checks if the graph complies with the constraints specified in the style tokens $\mathcal{S}$; (3) *Optimization Potential*: it proactively probes for superior configurations that transcend the current design. The results are synthesized into a comprehensive refinement report.

Moderator ($\mathcal{A}_M$) The moderator $\mathcal{A}_M$ serves as the strategic decision-maker, adjudicating the evolution of $\mathcal{G}_{t+1}$ by weighing the refinement report against the global objective. Each cycle yields one of two outcomes:

- *Adaptive Truncation*: If $\mathcal{G}_{t+1}$ satisfies $\mathcal{S}$ and the potential for further gain is marginal, $\mathcal{A}_M$ terminates the process and outputs $\mathcal{G}^*$.
- *Iterative Guidance*: Otherwise, $\mathcal{A}_M$ directs $\mathcal{A}_P$ to implement the critic's suggestions to resolve inconsistencies or enhance structural depth.

Initialization To ensure high initial diversity and eliminate human intervention, we deploy a similar adversarial mechanism before the evolutionary loop to construct an initial pool: the proposer $\mathcal{A}_P$ activates latent information to propose candidate attributes, while the critic $\mathcal{A}_C$ filters these attributes based on requirements such as orthogonality, validity, and diversity. This adversarial process builds a self-organized initial pool, containing:

- **Style Tokens ($\mathcal{S}$)**: A rich and diverse set of stylistic constraint dimensions, each offering as comprehensive a range of options as possible.
- **Concept Taxonomy ($\mathcal{C}$)**: A comprehensive atlas of mathematical domains.

At the onset ($t = 0$), the system randomly samples from this initial pool to generate the initial graph $\mathcal{G}_0$, ensuring that the entire dataset is driven solely by the model's intrinsic representational diversity.

In practice, the concept taxonomy can be derived from a variety of sources, such as existing knowledge bases, interactions between humans and LLMs, or weaknesses identified from evaluations of the target LLM (Rao et al., 2025b).

3.3 Phase 2: The Executor (Base-Level)

The Executor is a conditional generative model that performs semantic instantiation. It receives the linearized textual representation of the constraint graph $\mathcal{G}^*$ alongside the set of Style Tokens $\mathcal{S}$:

$$(Q, A) \sim P_{\text{executor}}(\cdot \mid \mathcal{G}^*, \mathcal{S})$$

where Q denotes the natural language problem statement and A is the step-by-step reasoning chain. By conditioning the generation on $\mathcal{G}^*$, the executor is freed from the burden of exploring and constructing complexity and diversity, allowing it to focus solely on language itself, thereby enabling the generation of diverse textual scenarios.

To further ensure the reliability of the synthesized question-answer pairs, we adopt a general model-based verification scheme (Zheng et al., 2023), which involves employing an external model as a judge to evaluate the logical correctness of the generated questions and answers, as well as the consistency between their descriptions. Only samples that pass this verification are retained.

4 Experiment

4.1 Setup

Data Synthesis Implementation We implement MathAgent through a multi-model pipeline. First, `DeepSeek-V3` (DeepSeek-AI et al., 2025b) is employed to construct candidate pools for style tokens $\mathcal{S}$ and concept taxonomy $\mathcal{C}$, establishing a diverse initial state for meta-level evolution, with this model also responsible for generating the final solutions. Next, the intermediate cyclic adversarial evolution led by the Legislator, the semantic instantiation performed by the Executor, and the subsequent verification process are all driven by `Qwen2.5-32B-Instruct` (Yang et al., 2025b). All generation stages maintain a uniform temperature of 0.3. We synthesize a final corpus of 1K instances, aligning the data scale with the baseline datasets introduced in the following section to ensure a fair comparison of data efficiency.

Baseline We benchmark the proposed synthetic data against datasets curated through rigorous human-designed filtering pipelines in supervised fine-tuning (SFT). To facilitate extensive experimentation, we select two well-known small-scale open-source datasets, **LIMO** (Ye et al., 2025) and **s1K** (Muennighoff et al., 2025) (containing approximately 0.8K and 1K samples, respectively), as representatives of such high-quality filtered data. Previous studies have indicated that the quality rather than the quantity of instruction-tuning datasets is critical (Zhou et al., 2023), making SFT experiments at this scale sufficiently informative. Details of these datasets are provided in Appendix D.

Models Our fine-tuning experiments are primarily conducted on the Qwen series of models, specifically including the `Qwen3-14B/8B/4B-Base` (Yang et al., 2025a) models, `Qwen2.5-7B` (Yang et al., 2025b), and `Qwen2.5-Math-7B` (Yang et al., 2024). We also include `Qwen2.5-7B-QwQ`, which is fine-tuned from the Math-Base model on 15K QwQ samples (Qwen, 2024). To ensure cross-architecture generalization, we extend our evaluation to `Llama-3.1-8B` (Grattafiori et al., 2024), `Llama-3.2-3B` (Meta, 2024), `Mistral-7B-v0.3` (Jiang et al., 2023), and `Gemma-2-9B` (Riviere et al., 2024). Comprehensive training details are provided in Appendix A and B.

Evaluation We evaluate the model's mathematical capability after SFT using eight mathematical test sets across the following three difficulty levels:

- **Elementary (Elem.)** GSM8K (Cobbe et al., 2021) & MATH500 (Hendrycks et al., 2021b);
- **Middle (Mid.)** Minerva Math (Lewkowycz et al., 2022), Gaokao 2023en (Liao et al., 2024) &

Model	Dataset	Elementary		Middle			Competition			Avg.
		GSM8K	MATH 500	Minerva Math	Gaokao 2023en	Olympiad Bench	AIME24 (Avg@8)	AIME25 (Avg@8)	AMC23 (Avg@8)	
Qwen3 Series Models										
Qwen3-14B-Base	Base	94.7	80.6	37.5	67.5	46.2	15.0	15.8	66.6	53.0
	LIMO	91.8	86.2	39.0	76.9	50.8	33.8	27.5	70.0	59.5
	S1K	87.5	86.4	**40.8**	76.1	52.6	37.9	25.0	75.9	60.3
	Ours	**95.4**	**91.8**	39.0	**79.0**	**56.3**	**38.8**	**30.0**	**80.6**	**63.9**
Qwen3-8B-Base	Base	92.0	76.8	32.7	64.9	41.8	17.5	13.8	58.4	49.7
	LIMO	88.3	80.4	35.7	68.6	44.9	19.6	24.2	60.3	52.8
	S1K	87.6	81.4	37.5	71.7	44.4	19.2	23.3	60.3	53.2
	Ours	**93.3**	**87.2**	**39.7**	**75.8**	**50.5**	**27.1**	**25.4**	**74.7**	**59.2**
Qwen3-4B-Base	Base	82.8	72.4	20.2	60.8	37.9	10.4	7.5	50.3	42.8
	LIMO	82.0	74.0	30.9	68.1	40.0	13.8	19.6	56.9	48.2
	S1K	84.8	76.8	33.5	68.6	40.6	16.7	20.0	53.1	49.3
	Ours	**92.0**	**81.8**	**35.3**	**69.6**	**44.6**	**19.2**	**20.4**	**65.0**	**53.5**
Qwen2.5 Series Models										
Qwen2.5-7B	Base	87.4	63.0	24.3	56.1	29.0	5.4	3.8	36.9	38.2
	LIMO	88.6	71.4	29.4	61.8	34.7	10.0	14.6	45.9	44.6
	S1K	89.4	68.8	30.5	62.9	35.9	11.7	9.6	43.8	44.1
	Ours	**90.1**	**74.6**	**30.9**	**68.3**	**38.5**	**14.2**	**15.0**	**55.9**	**48.4**
Qwen2.5-Math-7B	Base	66.7	64.0	12.1	56.1	28.3	11.7	4.2	38.8	35.2
	LIMO	87.4	72.2	31.6	63.1	37.9	10.8	14.6	47.2	45.6
	S1K	87.9	73.2	33.5	64.2	35.6	11.7	11.2	49.7	45.9
	Ours	**91.6**	**82.2**	**34.2**	**70.4**	**47.0**	**18.8**	**18.3**	**65.3**	**53.5**
Qwen2.5-7B-QwQ	Base	90.4	75.4	32.7	66.8	39.6	17.9	19.6	55.0	49.7
	LIMO	90.7	78.4	32.7	70.4	44.1	16.2	20.8	57.5	51.4
	S1K	90.7	78.2	30.5	64.9	44.7	17.9	20.0	55.0	50.2
	Ours	**91.7**	**83.8**	**33.8**	**73.5**	**50.7**	**27.5**	**21.7**	**70.6**	**56.7**
Llama, Mistral and Gemma Models										
Llama-3.1-8B	Base	40.9	13.4	4.8	15.1	3.1	0.0	0.0	4.7	10.3
	LIMO	66.0	24.6	6.2	24.9	6.4	0.0	0.4	8.8	17.2
	S1K	66.9	24.0	8.5	28.8	5.9	0.0	**0.8**	7.8	17.8
	Ours	**67.8**	**27.6**	**9.6**	**30.9**	**6.8**	**0.8**	**0.8**	**10.9**	**19.4**
Llama-3.2-3B	Base	25.8	7.4	2.6	9.1	2.5	0.0	0.0	**3.8**	6.4
	LIMO	22.8	8.6	3.3	11.9	2.7	0.0	0.0	2.5	6.5
	S1K	20.4	7.8	2.2	11.2	1.8	0.0	0.0	3.4	5.9
	Ours	**37.4**	**10.8**	**4.0**	**17.1**	**3.3**	0.0	**0.4**	**3.8**	**9.6**
Mistral-7B-v0.3	Base	18.0	7.4	2.6	9.4	2.2	0.0	0.0	2.5	5.3
	LIMO	33.1	12.6	4.0	14.5	3.0	0.0	**0.4**	2.2	8.7
	S1K	33.4	7.8	5.9	11.9	1.8	0.0	0.0	2.8	8.0
	Ours	**43.5**	**13.2**	**6.6**	**15.6**	**3.4**	**0.4**	**0.4**	**3.1**	**10.8**
Gemma-2-9B	Base	54.8	23.4	8.5	24.7	6.5	0.4	0.0	7.5	15.7
	LIMO	**76.6**	40.6	14.7	38.4	15.7	2.1	0.8	20.6	26.2
	S1K	76.4	41.0	**17.3**	43.1	13.5	1.2	0.4	20.3	26.7
	Ours	75.9	**43.8**	16.2	**43.4**	**16.0**	**2.9**	**1.2**	**23.8**	**27.9**

Table 1: **Main Results.** Model performance comparison on eight mathematical benchmarks (scores in %). Our synthesized dataset outperforms existing open-source baselines (LIMO and s1K) for SFT across multiple models.

Olympiad Bench (He et al., 2024);

- **Competition (Comp.)** AIME 2024 & 2025 (AoPS, 2025) and AMC23 (AoPS, 2023).

For Elementary and Middle-level sets, we report greedy decoding accuracy in a zero-shot setting. For Competition-level sets, we perform 8 sampling iterations per problem and report the average accuracy to mitigate variance. During answer generation for these datasets, we set the temperature to 0.1 and top_p to 0.95. A maximum generation length of 8192 tokens is applied across all test sets.

4.2 Main Results

Superior Performance As presented in Table 1, our synthesized dataset yields significant mathematical reasoning improvements across a comprehensive range of model architectures (Qwen, Llama, Mistral, Gemma), scales (3B–14B), and initialization stages (Base, Math, and SFT). Notably, our method consistently outperforms competitive open-source baselines (LIMO and s1K) across all eight benchmarks, highlighting the efficacy of the Legislator-Executor paradigm. Crucially, these

Model	Method	Elem.	Mid.	Comp.	Avg.
Qwen2.5-7B (Yang et al., 2025b)	Instruct	84.4	45.8	24.4	47.4
	Ours	83.3	45.9	28.4	48.7
Qwen2.5-Math-7B (Yang et al., 2024)	Instruct	89.5	48.5	28.6	51.3
	Ours	86.9	50.5	34.1	53.5
Gemma-2-9B (Riviere et al., 2024)	Instruct	61.1	21.8	6.0	25.7
	Ours	59.9	25.2	9.3	27.9

Table 2: Performance Comparison of Our Method versus the Official instruction-tuned Version. Superior performance of our method on more challenging test sets.

Method / Variant	Avg.	Δ
Full MathAgent	**45.4**	-
Impact of Legislator-Executor paradigm		
w/o Constraint Graph (Direct Gen.)	42.4	-3.0
Impact of Adversarial Evolution		
w/o Roundtable (One-pass)	43.1	-2.3

Table 3: **Ablation Study.** Δ indicates the performance drop relative to the full framework. The results underscore the necessity of both structural decoupling and adversarial evolution for high-quality synthesis.

gains are not a byproduct of data contamination; the synthesis process remains entirely independent of the evaluation benchmarks, as substantiated by the similarity analysis in Appendix C.

Cross-Difficulty Performance As shown in Table 1, the performance gains of our method are most prominent on high-difficulty benchmarks. For a clearer comparison, we selected representative models (The Mistral and Llama series were excluded due to their consistently poor performance on challenging test sets, while the Qwen3 series post-training versions were omitted as they incorporate thinking modes that preclude direct comparability) and compared them with their official instruction-tuned (Instruct) versions, with detailed results presented in Table 2.

While Instruct models maintain a slight edge on elementary benchmarks (potentially by capitalizing on high-frequency linguistic patterns within their large-scale training sets), our method significantly outperforms them on intermediate and competition-level tasks. This underscores the efficacy of adversarial evolution in capturing complex skeletons and logical dependencies, fostering robust reasoning capabilities that transcend simple pattern matching.

4.3 Ablation Study

To verify the necessity of the core components, we conduct an ablation study using `Qwen2.5-7B` as the base model and `Qwen2.5-Math-72B-Instruct` (Yang et al., 2024) as the resource-efficient solution annotator. The experimental configurations and results are summarized in Table 3.

The performance degradation observed in both variants highlights the synergy between our structural and evolutionary components. Eliminating the constraint graph results in a substantial drop (-3.0%), as the model reverts to superficial narrative imitation and high-frequency patterns in the absence of an explicit structural blueprint. Similarly, bypassing adversarial evolution (-2.3%) forces a reliance on initial semantic intuition, confirming that continuous structural refinement and difficulty-stretching via adversarial mechanisms are essential for pushing the boundaries of the model's generative capacity for high-quality problem synthesis.

4.4 Isolating Problem Quality

In this section, we disentangle the impact of problem quality from answer generation to verify that the performance gains in Table 1 are primarily driven by our synthesis approach.

SFT with Consistent Response Generation To eliminate the impact of differing response generation methodologies across datasets, we adopt a uniform response generation protocol. Specifically, all responses are generated in a single pass using the `Qwen2.5-Math-72B-Instruct` model without post-generation filtering. For the baseline problem datasets, in addition to **LIMO** and **s1K** datasets already used in Table 1, we included two additional datasets: **NuminaMath** (Li et al., 2024a) and **Magpie** (Xu et al., 2025). Detailed descriptions of these datasets are provided in Appendix D.

We conducted SFT on the `Qwen2.5-7B` model using response-standardized versions of all datasets to ensure a rigorous and controlled comparison. The results in Table 4 reveal that datasets relying on rigorous heuristic curation (LIMO and s1K) outperform both the unfiltered NuminaMath and the unconstrained, single-prompt synthesis of Magpie. In addition, even after standardizing response generation, our proposed method maintains its superiority across all comparisons.

IDPO We also evaluate our method using Iterative Direct Preference Optimization[1] (IDPO; Zhang

[1] https://github.com/RLHFlow/Online-DPO-R1

Method	Type	Size	Avg.
Qwen2.5-7B (Yang et al., 2025b)	-	-	38.2
+ NuminaMath (Li et al., 2024a)	SFT	1K	41.1
+ Magpie (Xu et al., 2025)	SFT	1K	41.5
+ LIMO (Ye et al., 2025)	SFT	0.8K	43.2
+ S1K (Muennighoff et al., 2025)	SFT	1K	43.0
+ MathAgent (Ours)	SFT	1K	45.4
Qwen2.5-7B-Instruct (Yang et al., 2025b)	-	-	47.4
+ NuminaMath (Li et al., 2024a)	IDPO	32K	47.9
+ Magpie (Xu et al., 2025)	IDPO	32K	48.0
+ MathAgent (Ours)	IDPO	32K	49.1

Table 4: **Isolating Problem Quality.** By standardizing response generation for all datasets, we observe consistent gains under both SFT and IDPO. This confirms that the superiority of MathAgent stems from the inherent logical quality of the synthesized problems, independent of response-level variance.

Method	BBH	Human Eval	MMLU	TruthfulQA	
				MC1	MC2
Qwen2.5-7B	69.7	60.4	74.3	38.9	56.3
+ NuminaMath	67.3	58.5	73.8	<u>38.8</u>	55.9
+ Magpie	<u>68.2</u>	60.4	73.6	37.7	54.9
+ LIMO	67.8	62.2$^{\uparrow}$	<u>73.8</u>	37.9	**56.2**
+ S1K	67.4	<u>63.4</u>$^{\uparrow}$	73.2	37.2	55.1
+ MathAgent	**68.4**	**64.0**$^{\uparrow}$	**74.1**	**38.9**	<u>56.0</u>

Table 5: **Cross-Task Generalization Benchmarks. Bold** and <u>underlined</u> values denote the best and second-best results among the fine-tuned models, respectively. The superscript $^{\uparrow}$ highlights performance improvements over the base model. TruthfulQA is evaluated using the standard MC1 (single-choice) and MC2 (multiple-choice) metrics. Our method effectively mitigates catastrophic forgetting while demonstrating positive transfer to code generation (HumanEval).

et al., 2025b; Rao et al., 2026). This approach requires the model to autonomously explore the solution space during training, thereby better revealing the intrinsic quality differences among various problem sets. The experiments are conducted using `Qwen2.5-7B-Instruct` as the base model, and the LIMO and s1K datasets are excluded from this comparison due to their relatively limited size. Results in Table 4 show that, despite the limited headroom for improvement given an already instruction-tuned base model, our method still achieves superior performance.

4.5 Cross-Task Generalization

Fine-tuning on domain-specific data often incurs a trade-off in general capability degradation. To verify that our method maintains robust cross-task generalization while enhancing mathematical reasoning, we extend our evaluation to the following benchmarks: BBH (Suzgun et al., 2023), HumanEval (Chen et al., 2021), MMLU (Hendrycks et al., 2021a), and TruthfulQA (Lin et al., 2022). Adopting the same SFT training setup as in Table 4, the results are summarized in Table 5.

Overall, performance variations across datasets are marginal, highlighting two primary trends. First, a slight decline on non-coding benchmarks suggests minor forgetting, yet this effect is constrained within a narrow range, likely due to the limited scale of our training samples. Notably, our method exhibits the most robust capability retention among all fine-tuned models. Second, on the HumanEval benchmark, which shares high logical synergy with mathematical reasoning, reasoning-enhanced models including LIMO, s1K, and our approach consistently show performance gains over the base model. Among these, our method achieves the most significant improvement, demonstrating effective positive transfer to programming tasks.

5 Analysis

This section analyzes the characteristics of the data generated by our approach. To this end, we compare our method against the aforementioned Magpie (Xu et al., 2025) and OpenR1-Math (OpenR1, 2025). Specifically, OpenR1-Math is derived from NuminaMath but serves as a rigorously filtered, high-quality subset, thus offering higher analytical value than random sampling from the raw source.

5.1 Data Quality and Difficulty

Following the protocol in Chen et al. (2024b), we employ `Qwen2.5-32B-Instruct` to evaluate both the quality and difficulty of the datasets on a five-level scale (The relevant prompts are detailed in Appendix G.2). The resulting distributions are visualized in Figure 2(a) and (b), respectively. For the manual evaluation of overall quality through sampling, please refer to Section 5.4.

Data Quality As shown in Figure 2(a), both synthetic datasets yield a higher proportion of "excellent" samples compared to OpenR1-Math, with our method further outperforming the Magpie baseline. While the synthetic datasets exhibit a slightly higher share of "very poor" instances than OpenR1-Math, the absolute proportion remains negligible. This is an expected consequence of the syn-

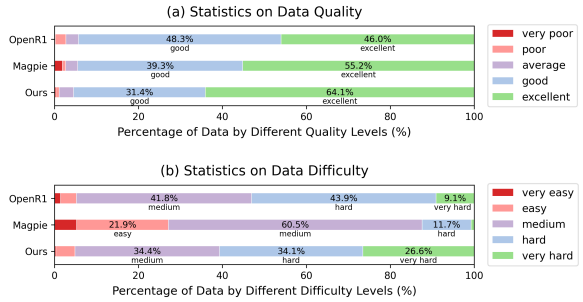

Figure 2: **Quality and Difficulty Distributions.** Quality and difficulty increase from left to right. Our method shows a significant advantage in generating high-quality, high-difficulty mathematical problems.

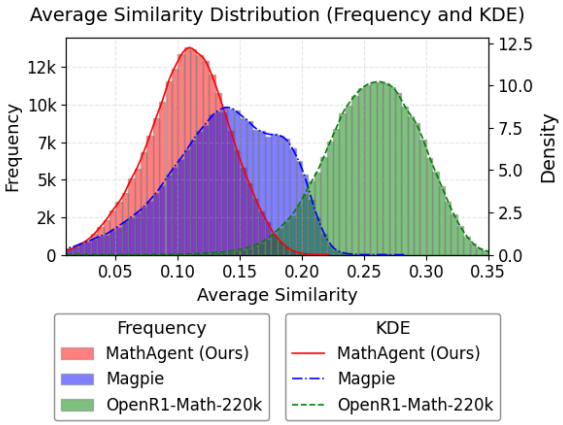

Figure 3: **Intra-dataset Similarity Distribution.** The bars represent frequency counts, while the curves denote Kernel Density Estimation (KDE). The results demonstrate that our method exhibits lower intra-similarity, indicating superior dataset diversity.

thetic data being evaluated in its raw state, whereas OpenR1-Math has undergone post-filtering.

Data Difficulty Figure 2(b) demonstrates that our method holds a significant advantage in synthesizing high-difficulty problems, whereas Magpie is limited by its tendency to generate low-complexity, high-frequency data. In fact, our approach allows for flexible control over the adversarial iteration process via prompt engineering (e.g., style tokens). This capability enables the generation of problems at targeted difficulty levels, facilitating the construction of model-adaptive synthetic datasets.

5.2 Dataset Diversity

Dataset diversity is widely acknowledged as a pivotal determinant of dataset quality. Despite the lack of a universal standard for its quantification, we employ two primary metrics in this study: intra-dataset similarity and dataset coverage. Intuitively, lower intra-similarity coupled with higher coverage indicates superior diversity. For experimental details of this section, please refer to Appendix E.

Dataset Intra-Similarity To quantify dataset diversity, we measure intra-dataset similarity by calculating the average similarity between data instances within each dataset. We utilize the complete 220K-version OpenR1-Math dataset and randomly sample an equivalent number of instances (220K) from both our dataset and Magpie. The distribution of average similarity scores is shown in Figure 3. The results demonstrate that our method achieves the lowest intra-dataset similarity.

Dataset Coverage The coverage of mathematical problems is primarily characterized by the diversity of the underlying knowledge points. To evaluate this, we adopt an approach integrating methods from InsTag (Lu et al., 2024a) and Zhao et al. (2024). As illustrated in Figure 4, we employ t-SNE (Maaten and Hinton, 2008) to project the semantic embeddings of knowledge point tags derived from 10K randomly sampled instances into a two-dimensional space. The results show that the distribution of the data generated by our method substantially encompasses the coverage areas of both Magpie and OpenR1-Math, which further validates the superior diversity of our approach in generating mathematical problems.

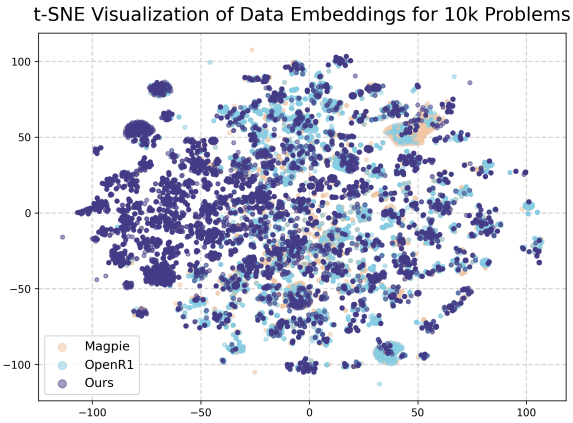

Figure 4: **t-SNE Visualization of Knowledge Points.** The extensive coverage of the dark blue points (representing our method) demonstrates the significant diversity of the generated mathematical problems.

5.3 Data Scaling

In this section, we investigate the scaling laws governing our approach by analyzing the correlation

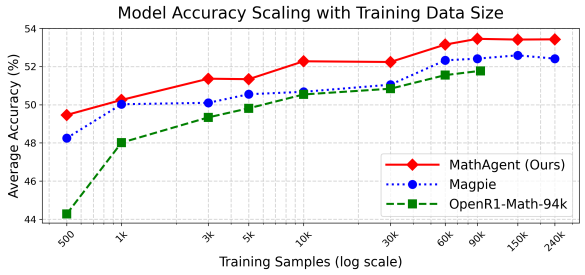

Figure 5: **Performance Scaling Analysis.** The x-axis is plotted on a logarithmic scale for clarity. While performance generally improves with increased data scale, our method maintains a consistent and significant performance advantage over the baselines.

between model performance and dataset size. We employ `Qwen2.5-Math-7B` as the base model and adjust the training schedule to 3 epochs to facilitate efficient experimentation.

As illustrated in Figure 5, performance across all three datasets exhibits a trend of rapid initial growth followed by stabilization (or slight saturation) as data scale increases, with peak performance achieved at approximately 100K samples. Crucially, our method consistently yields superior performance gains compared to other methods, demonstrating its robustness and data efficiency.

5.4 Human Evaluation

Given the potential systematic bias of relying solely on LLM-as-a-judge (Chen et al., 2024a; Li et al., 2025; Szymanski et al., 2025), we also conduct a human evaluation. Specifically, we randomly sample 100 instances from each of Magpie, OpenR1, and MathAgent, and perform a blinded evaluation with source anonymization on a 5-point scale, where higher scores indicate better quality. The results show that MathAgent achieves the highest average score (4.2/5.0), outperforming Magpie (3.5/5.0) and OpenR1 (3.9/5.0).

Expert feedback further suggests that, although Magpie-generated samples are often grammatically fluent, their logical structures tend to be simplistic or repetitive, directly resulting in the lowest scores for Magpie in this evaluation. In contrast, MathAgent demonstrates stronger structural complexity and logical depth, consistent with the high-difficulty distribution observed in the automated analysis. We also observe cases where the LLM judge fails to identify subtle logical flaws recognized by human experts, highlighting the limitations of automated evaluation alone. Detailed evaluation settings and discussions of limitations are provided in Appendix F.

6 Conclusion

By decoupling blueprint design from linguistic realization, this paper proposes the MathAgent framework, which maximizes structural diversity through adversarial optimization on constraint graphs to guide the generation of high-quality, long-tail distributed data. Experimental results demonstrate that models fine-tuned on samples synthesized via this approach outperform existing human-filtered datasets. This provides a scalable pathway for constructing high-quality mathematical reasoning data, breaking through the limitations of manual annotation and avoiding the mode collapse issues inherent in previous synthesis methods.

Limitations

Despite the performance gains and robust scaling characteristics demonstrated by MathAgent, our study has several limitations:

- **Domain Specificity.** Our framework currently focuses on mathematical reasoning where logic is highly structured and objective. Its applicability to more open-ended or less formal domains, such as legal reasoning or creative writing, requires further exploration into how to define effective constraint graphs for non-mathematical tasks.
- **Computational Overhead.** The adversarial evolution process involves multiple iterations between the legislator and executor models. This iterative cycle inevitably leads to higher computational costs and longer synthesis times compared to single-pass methods that do not require multi-round refinement.
- **Ground-Truth Verification.** As the adversarial process pushes problem complexity to extreme levels, ensuring the absolute correctness of the generated ground-truth solutions becomes increasingly difficult. Future work could benefit from integrating external formal verifiers or symbolic solvers to guarantee the accuracy of synthesized reasoning chains.

These limitations also point to potential research directions for the future.

Ethics Statement

This work complies with the ACL Ethics Policy. We utilize publicly available datasets and open-source models, ensuring that all resources are prop-

erly cited and the experiments are reproducible. We acknowledge the inherent risks associated with LLMs, including the potential for hallucinations and the generation of non-factual content. While our method aims to enhance reasoning reliability in the mathematical domain, users should exercise caution and verify model outputs when deploying such systems in critical or real-world applications.

Acknowledgments

The authors would like to express their sincere gratitude to the Ph.D. students in the Department of Mathematical Sciences and the Department of Statistics and Data Science at Tsinghua University for generously devoting their time and effort to the manual verification process. We also thank the Huawei Large Model Data Technology Lab for its valuable guidance and suggestions, as well as for its feedback on the effectiveness of follow-up deployment. Furthermore, we are grateful to the anonymous reviewers and the area chair for their insightful and constructive comments, which have greatly improved the quality of this work.

References

Art of Problem Solving AoPS. 2023. AMC Problems and Solutions. https://artofproblemsolving.com/wiki/index.php/AMC_12_Problems_and_Solutions. Accessed: 2025-09-30.

Art of Problem Solving AoPS. 2025. AIME Problems and Solutions. https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions. Accessed: 2025-09-30.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Guiming Hardy Chen, Shunian Chen, Ziche Liu, Feng Jiang, and Benyou Wang. 2024a. Humans or LLMs as the judge? a study on judgement bias. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8301–8327, Miami, Florida, USA. Association for Computational Linguistics.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. 2024b. Alpagasus: Training a better alpaca with fewer data. In *The Twelfth International Conference on Learning Representations*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Sirui Chen, Changxin Tian, Binbin Hu, Kunlong Chen, Ziqi Liu, Zhiqiang Zhang, and Jun Zhou. 2025. Arrows of math reasoning data synthesis for large language models: Diversity, complexity and correctness. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, CIKM '25, page 4665–4669, New York, NY, USA. Association for Computing Machinery.

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

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025a. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, and 181 others. 2025b. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, and 3 others. 2023. PaLM-e: An embodied multimodal language model. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 8469–8488. PMLR.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. 2024. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first International Conference on Machine Learning*.

Google. 2024. Gemini 2.0 flash thinking mode (gemini-2.0-flash-thinking-exp-1219). Accessed via Google Cloud Vertex AI.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

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

Etash Kumar Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Rea Sprague, Ashima Suvarna, Benjamin Feuer, Leon Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, and 32 others. 2026. Openthoughts: Data recipes for reasoning models. In *The Fourteenth International Conference on Learning Representations*.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3828–3850, Bangkok, Thailand. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the math dataset. *Preprint*, arXiv:2103.03874.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, and 3 others. 2022. Training compute-optimal large language models. *Preprint*, arXiv:2203.15556.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2024. MetaGPT: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2023. Unnatural instructions: Tuning language models with (almost) no human labor. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14409–14428, Toronto, Canada. Association for Computational Linguistics.

Arthur Jacot, Franck Gabriel, and Clement Hongler. 2018. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

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.

Yuxuan Jiang and Francis Ferraro. 2026. Beyond math: Stories as a testbed for memorization-constrained reasoning in LLMs. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5590–5607, Rabat, Morocco. Association for Computational Linguistics.

Yuxuan Jiang, Dawei Li, and Frank Ferraro. 2025. Drp: Distilled reasoning pruning with skill-aware step decomposition for efficient large reasoning models. *Preprint*, arXiv:2505.13975.

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

Xiaopeng Ke, Hexuan Deng, Xuebo Liu, Jun Rao, Zhenxi Song, Jun Yu, and Min Zhang. 2025. AQuilt: Weaving logic and self-inspection into low-cost, high-relevance data synthesis for specialist LLMs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5752–5785, Suzhou, China. Association for Computational Linguistics.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. *Preprint*, arXiv:2206.14858.

Dawei Li, Bohan Jiang, Liangjie Huang, Alimohammad Beigi, Chengshuai Zhao, Zhen Tan, Amrita Bhattacharjee, Yuxuan Jiang, Canyu Chen, Tianhao Wu, Kai Shu, Lu Cheng, and Huan Liu. 2025. From generation to judgment: Opportunities and challenges of LLM-as-a-judge. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 2757–2791, Suzhou, China. Association for Computational Linguistics.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for "mind" exploration of large language model society. In *Advances in Neural Information Processing Systems*, volume 36, pages 51991–52008. Curran Associates, Inc.

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*.

Yicheng Li, Zixiong Yu, Guhan Chen, and Qian Lin. 2024b. On the eigenvalue decay rates of a class of neural-network related kernel functions defined on general domains. *Journal of Machine Learning Research*, 25(82):1–47.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. 2024. Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17889–17904, Miami, Florida, USA. Association for Computational Linguistics.

Minpeng Liao, Chengxi Li, Wei Luo, Wu Jing, and Kai Fan. 2024. MARIO: MAth reasoning with code interpreter output - a reproducible pipeline. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 905–924, Bangkok, Thailand. Association for Computational Linguistics.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Keming Lu, Hongyi Yuan, Zheng Yuan, Runji Lin, Junyang Lin, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2024a. #instag: Instruction tagging for analyzing supervised fine-tuning of large language models. In *The Twelfth International Conference on Learning Representations*.

Zimu Lu, Aojun Zhou, Houxing Ren, Ke Wang, Weikang Shi, Junting Pan, Mingjie Zhan, and Hongsheng Li. 2024b. 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, Bangkok, Thailand. Association for Computational Linguistics.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, Yansong Tang, and Dongmei Zhang. 2025. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. In *The Thirteenth International Conference on Learning Representations*.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Maosongcao Maosongcao, Taolin Zhang, Mo Li, Chuyu Zhang, Yunxin Liu, Conghui He, Haodong Duan, Songyang Zhang, and Kai Chen. 2025. Condor: Enhance LLM alignment with knowledge-driven data synthesis and refinement. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 22392–22412, Vienna, Austria. Association for Computational Linguistics.

Team Meta. 2024. Llama 3.2: Revolutionizing edge ai and vision with open, customizable models. Blog Post.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling. *Preprint*, arXiv:2501.19393.

Team OpenR1. 2025. open-r1/OpenR1-Math-220k. `https://huggingface.co/datasets/open-r1/OpenR1-Math-220k`. Accessed: 2025-09-30.

Team Qwen. 2024. Qwq: Reflect deeply on the boundaries of the unknown. Blog Post.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE.

Jun Rao, Yunjie Liao, Xuebo Liu, Zepeng Lin, Lian Lian, Dong Jin, Shengjun Cheng, Jun Yu, and Min Zhang. 2025a. Seapo: Strategic error amplification for robust preference optimization of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2025*. Association for Computational Linguistics.

Jun Rao, Zepeng Lin, Xuebo Liu, Xiaopeng Ke, Lian Lian, Dong Jin, Shengjun Cheng, Jun Yu, and Min

Zhang. 2025b. APT: Improving specialist LLM performance with weakness case acquisition and iterative preference training. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 20958–20980, Vienna, Austria. Association for Computational Linguistics.

Jun Rao, Xuebo Liu, Hexuan Deng, Zepeng Lin, Zixiong Yu, Jiansheng Wei, Xiaojun Meng, and Min Zhang. 2026. Dynamic sampling that adapts: Iterative dpo for self-aware mathematical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*.

Jun Rao, Xuebo Liu, Haotian Yan, Junjie Shen, Haosi Mo, Yanghaopeng Dong, Zihao Yan, Ziyi Wang, Zepeng Lin, Xiaojun Meng, Zixiong Yu, Liqun Deng, Jiansheng Wei, Yunhe Wang, and Min Zhang. 2025c. A data-centric perspective on the lifecycle of large language models. *TechRxiv*, 2025(1220).

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Team Gemma: Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, and 178 others. 2024. Gemma 2: Improving open language models at a practical size. *Preprint*, arXiv:2408.00118.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, pages 68539–68551. Curran Associates, Inc.

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

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. 2023. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, Toronto, Canada. Association for Computational Linguistics.

Annalisa Szymanski, Noah Ziems, Heather A. Eicher-Miller, Toby Jia-Jun Li, Meng Jiang, and Ronald A. Metoyer. 2025. Limitations of the llm-as-a-judge approach for evaluating llm outputs in expert knowledge tasks. In *Proceedings of the 30th International Conference on Intelligent User Interfaces*, IUI '25, page 952–966, New York, NY, USA. Association for Computing Machinery.

Shuo Tang, Xianghe Pang, Zexi Liu, Bohan Tang, Rui Ye, Tian Jin, Xiaowen Dong, Yanfeng Wang, and Siheng Chen. 2025. Synthesizing post-training data for LLMs through multi-agent simulation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23306–23335, Vienna, Austria. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. 2024. Position: Will we run out of data? limits of LLM scaling based on human-generated data. In *Forty-first International Conference on Machine Learning*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Zhexuan Wang, Yutong Wang, Xuebo Liu, Liang Ding, Miao Zhang, Jie Liu, and Min Zhang. 2025. AgentDropout: Dynamic agent elimination for token-efficient and high-performance LLM-based multi-agent collaboration. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 24013–24035, Vienna, Austria. Association for Computational Linguistics.

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

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022b. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Kaiwen Wei, Rui Shan, Dongsheng Zou, Jianzhong Yang, Bi Zhao, Junnan Zhu, and Jiang Zhong. 2026.

Mirage: Scaling test-time inference with parallel graph-retrieval-augmented reasoning chains. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 33818–33826.

Ningning Xu, Yuxuan Jiang, Shubhashis Roy Dipta, and Hengyuan Zhang. 2026. Learning how to use tools, not just when: Pattern-aware tool-integrated reasoning. *Preprint*, arXiv:2509.23292.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. 2025. Magpie: Alignment data synthesis from scratch by prompting aligned LLMs with nothing. In *The Thirteenth International Conference on Learning Representations*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 23 others. 2025b. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *Preprint*, arXiv:2409.12122.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. 2025. LIMO: Less is more for reasoning. In *Second Conference on Language Modeling*.

Zixiong Yu, Songtao Tian, and Guhan Chen. 2025. Divergence of empirical neural tangent kernel in classification problems. In *The Thirteenth International Conference on Learning Representations*.

Shuang Zeng, Xinyuan Chang, Mengwei Xie, Xinran Liu, Yifan Bai, Zheng Pan, Mu Xu, and Xing Wei. 2025a. Futuresightdrive: Thinking visually with spatio-temporal cot for autonomous driving. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Shuang Zeng, Dekang Qi, Xinyuan Chang, Feng Xiong, Xie Shichao, Xiaolong Wu, Shiyi Liang, Mu Xu, and Xing Wei. 2026. JanusVLN: Decoupling semantics and spatiality with dual implicit memory for vision-language navigation. In *The Fourteenth International Conference on Learning Representations*.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun MA, and Junxian He. 2025b. SimpleRL-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. In *Second Conference on Language Modeling*.

Yiming Zeng, Wanhao Yu, Zexin Li, Tao Ren, Yu Ma, Jinghan Cao, Xiyan Chen, and Tingting Yu. 2025c. Bridging the editing gap in LLMs: FineEdit for precise and targeted text modifications. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 2193–2206, Suzhou, China. Association for Computational Linguistics.

Andy K Zhang, Kevin Klyman, Yifan Mai, Yoav Levine, Yian Zhang, Rishi Bommasani, and Percy Liang. 2025a. Position: Language model developers should report train-test overlap. In *Forty-second International Conference on Machine Learning Position Paper Track*.

Hanning Zhang, Jiarui Yao, Chenlu Ye, Wei Xiong, and Tong Zhang. 2025b. Online-dpo-r1: Unlocking effective reasoning without the ppo overhead, 2025. *Notion Blog*.

Hengyuan Zhang, Shiping Yang, Xiao Liang, Chenming Shang, Yuxuan Jiang, Chaofan Tao, Jing Xiong, Hayden Kwok-Hay So, Ruobing Xie, Angel X. Chang, and Ngai Wong. 2026. Find your optimal teacher: Personalized data synthesis via router-guided multi-teacher distillation. *Preprint*, arXiv:2510.10925.

Shaowei Zhang and Deyi Xiong. 2025. Debate4MATH: Multi-agent debate for fine-grained reasoning in math. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 16810–16824, Vienna, Austria. Association for Computational Linguistics.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, and 3 others. 2023. A survey of large language models. *Preprint*, arXiv:2303.18223.

Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. 2024. Wildchat: 1m chatGPT interaction logs in the wild. In *The Twelfth International Conference on Learning Representations*.

Xinping Zhao, Xinshuo Hu, Zifei Shan, Shouzheng Huang, Yao Zhou, Xin Zhang, Zetian Sun, zhenyu liu, Dongfang Li, Xinyuan Wei, Youcheng Pan, Yang Xiang, Meishan Zhang, Haofen Wang, Jun Yu, Baotian Hu, and Min Zhang. 2026. KaLM-embedding-v2: Superior training techniques and data inspire a versatile embedding model. In *The Fourteenth International Conference on Learning Representations*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging

LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. 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)*, pages 400–410, Bangkok, Thailand. Association for Computational Linguistics.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, LILI YU, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. Lima: Less is more for alignment. In *Advances in Neural Information Processing Systems*, volume 36, pages 55006–55021. Curran Associates, Inc.

A Prompts for model training and testing

Standard prompt designs in current mathematical reasoning tasks typically combine *zero-shot Chain-of-Thought (CoT) guidance* with *structured output constraints*. This approach strikes a balance between maintaining model accuracy and facilitating automated answer extraction and evaluation. Thus, we adopt this strategy in our study (refer to the Complex Prompt in Figure 6). However, existing research indicates that such complex prompts may impose a burden on models with limited instruction-following capabilities (Zeng et al., 2025b), potentially leading to performance degradation. Consequently, a simplified prompt design is adopted for such models (refer to the Simple Prompt in Figure 6).

Simple Prompt

```
Question:
{input}
Answer:
Let's think step by step.
```

Complex Prompt

```
<|im_start|>system
You are a helpful assistant.<|im_end|>
<|im_start|>user
{input}
Please reason step by step, and put
your final answer within
\\boxed{}.<|im_end|>
<|im_start|>assistant
{output}
```

Figure 6: Comparison between simple prompts and more complex prompts (using the Qwen series prompt templates as an example).

We empirically verify this phenomenon in Table 6, using `Qwen3-8B-Base` and `Llama-3.1-8B` as representative models. Experimental results reveal that the non-instruction-tuned `Llama-3.1-8B` suffers significant performance degradation under complex prompts (adapted to the corresponding Llama template), whereas `Qwen3-8B-Base` demonstrates improved performance in similar settings. Based on further observations, we adopt the simple prompt for the Llama, Mistral, and Gemma series, while retaining the standard complex prompt format for the Qwen series.

Model	Prompt Template	Avg.
Qwen3-8B-Base (Yang et al., 2025a)	simple	32.5
	Qwen-complex	49.7
Llama-3.1-8B (Grattafiori et al., 2024)	simple	10.3
	Llama-complex	1.4
Llama-3.1-8B-Instruct (Grattafiori et al., 2024)	simple	23.5
	Llama-complex	31.0

Table 6: Performance comparison using Simple vs. Complex prompts. Complex prompts degrade the performance of Llama-Base but benefit Qwen-Base and Llama-Instruct.

Notably, although our experiments primarily applied complex prompt strategies to Qwen models, the applicability of this design is not limited to this series. Instead, as previously discussed, it is closely tied to the model's instruction-following capability. As shown in Table 6, the instruction-tuned `Llama-3.1-8B-Instruct` also demonstrates significant performance improvement when using complex prompts (adapted to the corresponding Llama template). However, considering that instruction-tuned models are generally less suitable for few-shot supervised fine-tuning, our main experiments do not focus on them.

B Training Details for SFT

We perform full-parameter fine-tuning of the model using LLaMA-Factory[2] (Zheng et al., 2024). Adhering to the strategies discussed in Section A, we apply the corresponding prompt templates (Simple or Complex) for each model during the data formatting stage. To optimize memory usage and training efficiency, we employ the DeepSpeed ZeRO Stage 3 (Rajbhandari et al., 2020) strategy. Training utilizes the AdamW (Loshchilov and Hutter, 2019) optimizer ($\beta_1 = 0.9, \beta_2 = 0.95$, weight decay $= 1 \times 10^{-4}$) combined with a cosine learning rate decay schedule (Loshchilov and Hutter, 2017) and a warmup ratio of 0.05. The maximum sequence length is limited to 4096 tokens, and the maximum gradient norm is clipped at 1.0. The training process uses bfloat16 precision throughout to ensure numerical stability and computational efficiency, and operates in a distributed environment across 8 devices, with the random seed fixed at 42 to ensure reproducibility. Certain hyperparameters, such as the initial learning rate, batch size, and number of training epochs, vary across models; these specific configurations are detailed in Table 7.

Model	Peak Learning Rate	Batch Size	Training Epochs
Qwen-7B/8B Series & Llama-3.1-8B	1e-5	2×8	5
Qwen3-14B-Base & Gemma-2-9B	5e-6	1×8	5
Qwen3-4B-Base & Llama-3.2-4B	2e-5	4×8	5
Mistral-7B-v0.3	1e-5	2×8	3

Table 7: **Training Hyperparameters** for Different Models. Note that "$\times 8$" in the batch size column denotes the total batch size calculated as *per-device batch size* $\times$ *number of devices* (8).

C Training-Test Set Similarity

For synthetic data, memorization, train–test overlap, and potential information leakage constitute important sources of bias, especially in reasoning settings where benchmark performance may be inflated by surface recall (Jiang and Ferraro, 2026). To mitigate these risks, we compute the similarity between each of the three training sets and the test set (Zhang et al., 2025a). We adopt the *average maximum similarity (AMS)* in the embedding space as our metric. Specifically, we first map all mathematical problems into the embedding space using the `all-mpnet-base-v2`[3] model (Reimers and Gurevych, 2019), which enables the computation of pairwise similarity between any two problems. For each instance in the test set, we retrieve the sample in the training set with the highest similarity to it and record this maximum similarity value. The final metric is then calculated as the average of these maximum similarity values across all test instances.

The final results are presented in Figure 7. It is evident that the similarity between our synthetic dataset and the test sets is not significantly higher than that of the other training sets. Conversely, for most test sets, the similarity metric obtained by our method is noticeably lower. This indicates that no additional information leakage occurs during our synthetic data generation process, and thus it does not introduce bias into the experimental results.

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

[3] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

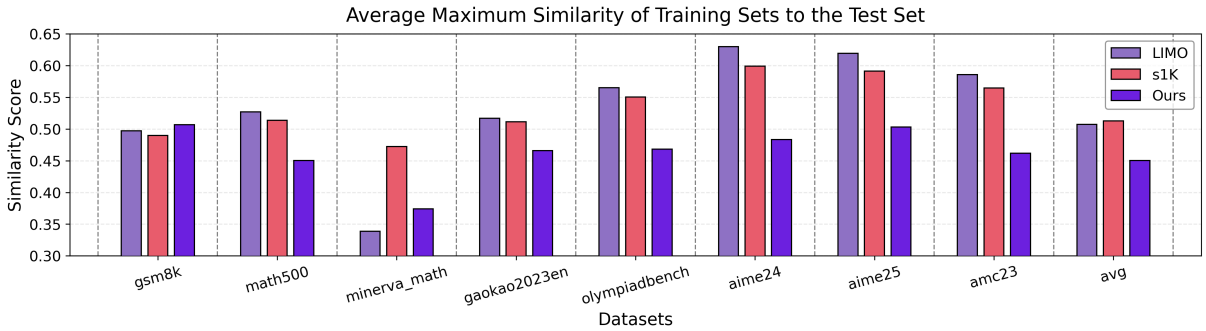

Figure 7: **Average Maximum Similarity** Between Training and Test Datasets: A lower similarity score indicates a reduced likelihood of data leakage. This figure demonstrates that our synthetic data does not carry a higher risk of data leakage compared to LIMO and s1K.

D Details for Baseline Data Comparisons

In this section, we describe the selection criteria and implementation details of the baseline data sources used in our performance evaluation and data characteristic analysis.

Overall Criteria Open-source datasets and synthesis methods are highly diverse (Rao et al., 2025c). Because our method targets seed-free synthesis of mathematical problems, we mainly compare against methods in the same paradigm, such as Magpie (Xu et al., 2025), rather than seed-based approaches (e.g., Self-Instruct; Wang et al. 2023). Methods such as OpenThought (Guha et al., 2026), which focus on constructing high-quality *reasoning traces* from existing public datasets, are complementary to our work and therefore excluded from direct comparison. In addition, program-assisted generation frameworks such as ARROWS (Chen et al., 2025) are effective when problems can be formulated as executable programs, but are limited to such settings and are therefore excluded from direct comparison. Since many open-source math problems are derived from NuminaMath (Li et al., 2024a), we include samples from it as a raw-data baseline, together with several representative curated subsets built upon it (Ye et al., 2025; Muennighoff et al., 2025; OpenR1, 2025).

LIMO & s1K We employ the open-source LIMO (Ye et al., 2025) and s1K (Muennighoff et al., 2025) datasets as high-quality baselines. These datasets are constructed through expert-designed pipelines and rigorous screening to ensure reasoning depth, representing the state-of-the-art in small-scale, curated reasoning data. For s1K, which offers two reasoning model versions based on `Gemini` (Google, 2024) and `DeepSeek-R1` (DeepSeek-AI et al., 2025a) respectively, we specifically select the latter for the experiments in Table 1 to establish a more competitive and rigorous baseline.

NuminaMath NuminaMath (Li et al., 2024a) serves as a large-scale open-source collection of mathematical problems, providing a rich resource of raw mathematical tasks for research. In our experiments, we randomly sample the required number of problems from the complete dataset to evaluate the model's performance across a broad distribution of raw mathematical tasks.

Magpie To construct the Magpie (Xu et al., 2025) baseline, we employ `Qwen2.5-32B-Instruct` as the question generator. Following the original protocol, we adopt the math-specific prompt from Figure 2 of Xu et al. (2025) as the system prompt. Critically, Magpie's generation quality is highly sensitive to the sampling temperature T. While higher temperatures can enhance dataset diversity, they often lead to a non-negligible decline in synthesis quality; thus, we adopt a balanced setting of $T = 1$ to ensure a fair and effective comparison. Unlike our iterative adversarial approach, Magpie relies on a static, uniform system prompt for direct generation. For the comprehensive data characteristic analysis, we scale this process to generate a candidate pool of 300,000 samples.

OpenR1-Math OpenR1-Math (OpenR1, 2025) is an open-source dataset derived from NuminaMath. As a rigorously filtered and high-quality subset, it offers significantly higher analytical value than the original raw data. The dataset consists of two versions, including a further curated set of 94k problems and a complete version containing 220k instances. For experiments that prioritize data quality over quantity, such as assessments of data

quality and difficulty as well as SFT experiments for data scaling performance, we utilize OpenR1-Math-94k as the comparison baseline. In contrast, for diversity analysis, we select the larger OpenR1-Math-220k version to provide a more representative and comprehensive benchmark.

E Details for Diversity Analysis

In the internal similarity and coverage analysis, we uniformly use the `all-mpnet-base-v2` model to generate semantic embedding vectors, mapping problem descriptions and knowledge tags into a 768-dimensional vector space. The internal similarity is calculated as the average cosine similarity between the embedding vector of each sample and all other samples in the dataset pool. In the coverage analysis, the knowledge point extraction stage employs `Qwen2.5-32B-Instruct` to perform annotation on 10,000 randomly sampled mathematical problems, followed by dimensionality reduction visualization using t-SNE.

Notably, the similarity metric used here differs from that in Section C. While the latter employs maximum similarity to detect potential data contamination from near-identical samples, the average similarity used in this section is designed to evaluate global semantic density. Furthermore, due to the high baseline similarity inherent in mathematical reasoning data, maximum similarity tends to saturate (approach 1.0) at large scales, thereby reducing its discriminative power. Consequently, we adopt average similarity to provide a more robust and distinguishable measure of dataset diversity.

F Human Evaluation Details

Evaluation Setup To complement the automated evaluation, we conduct an expert human evaluation on samples drawn from Magpie, OpenR1, and MathAgent (Ours). We invite four Ph.D. researchers with expertise spanning Algebra, Geometry & Analysis, Statistics, and Mathematical Physics to ensure broad coverage across diverse mathematical domains.

We randomly sample 100 instances from each dataset, resulting in 300 evaluated instances in total. The evaluation is conducted under a blinded protocol with source anonymization, so evaluators do not know which method produced each sample. Each instance is rated on a 5-point scale, where higher scores indicate better overall quality:

- **1 (Serious Flaws):** The sample contains major logical or conceptual errors.
- **2 (Minor Defects):** The sample has small errors that do not completely invalidate it.
- **3 (Mediocre):** The sample is largely correct but trivial, repetitive, or lacking in depth.
- **4 (Good Quality):** The sample is correct, clear, and reasonably well-structured.
- **5 (Insightful):** The sample demonstrates strong logical depth, originality, or pedagogical value.

Additional Qualitative Findings Beyond the quantitative scores, the evaluators provide several consistent qualitative observations. First, Magpie-generated samples are often grammatically fluent and superficially coherent, but they frequently rely on repetitive reasoning patterns or exhibit limited structural complexity. As a result, their overall quality is often judged as moderate rather than strong.

Second, samples generated by MathAgent tend to exhibit multi-step reasoning, richer structural, and greater logical depth, which is consistent with the high-difficulty tendencies reflected in the automated evaluation. However, this advantage also comes with a trade-off: when the samples become overly difficult or structurally complex, the likelihood of hallucination may increase accordingly.

Third, we observe noticeable discrepancies between human judgments and LLM-based evaluation in some cases. In particular, the LLM judge occasionally assigns favorable scores to samples containing subtle logical flaws or weak reasoning steps that human experts can often identify more readily. This suggests that relying solely on automated evaluation may fail to fully capture the quality of mathematical reasoning data; this limitation becomes especially pronounced for problems with more complex structures.

Evaluation Limitations Despite its usefulness, the human evaluation has several limitations. First, due to the high cost and cognitive demand of expert review, the evaluation is conducted on a relatively small sample of 300 instances, far smaller than the full scale of the synthesized dataset.

Second, although we intentionally recruit evaluators from diverse mathematical areas, evaluator coverage remains limited relative to the full breadth of mathematical reasoning tasks. Some problems may fall closer to the expertise of certain evaluators than others. As a result, especially under limited evaluation time, some particularly difficult samples may be marked as uncertain or out-of-scope.

Third, human evaluation inevitably contains some degree of subjectivity, especially for high-level criteria such as insightfulness, elegance, or pedagogical value. While the scoring criteria help standardize judgments, they cannot completely eliminate individual variation in scoring.

G Prompts for Synthetic Data Generation and Analysis

G.1 Prompts for Data Synthesis

In this section, we describe the specific prompts used throughout the synthetic mathematical problem generation pipeline. Within the *Legislator* module, the prompt for the Proposer is detailed in Figure 9, the Critic's evaluation protocol is provided in Figure 10, and the Moderator's decision-making logic is given in Figure 11.

Regarding the *Executor* module, we focus on its primary function, namely the Semantic Instantiation process. Figure 12 illustrates the prompt design for transforming abstract constraint graphs ($\mathcal{G}^*$) and style tokens ($\mathcal{S}$) into fluent, natural language mathematical problems. To maintain focus on our structural innovations, the subsequent solution generation and model-based verification processes are not described in detail here, as they follow conventional workflows.

G.2 Prompts for Analysis

The prompts used for evaluating the quality and difficulty of the dataset (as discussed in Section 5.1) are designed with reference to the evaluation protocol proposed by Chen et al. (2024b) and are adapted from the methodology of Xu et al. (2025). The detailed prompt templates are illustrated in Figure 13.

H Case Study

We illustrate the core iterative process of the MathAgent framework with a representative example. For demonstration, the following style tokens are selected: *Difficulty: Medium*, *Question Type: Calculation*, *Context: Real-world Application*, and *Knowledge Level: Undergraduate*. It should be noted that although this case is closely related to the one shown in Figure 1, the latter is intended primarily to provide an intuitive overview of the framework. For clarity of presentation and layout, the example here is adapted accordingly.

As shown in Figure 8 (illustrating a two-round iteration; the original JSON output has been reformatted for clarity), the i-th node is denoted by v_i, and the directed edge from node i to j is denoted by e_{ij}. Nodes follow the "Concept: Description" format, conforming to the structural specifications defined in Figure 9.

Iteration 1 The initial topic is set to the *Saddle Surface*, which serves as a rudimentary conceptual primitive. Driven by the Proposer ($\mathcal{A}_P$), Graph 1 is generated. The Critic ($\mathcal{A}_C$) then provides a key assessment regarding specification alignment: if the parameters a and b in the saddle surface equation $z = x^2/a^2 - y^2/b^2$ are not constrained by $a = b$, the resulting surface integral may involve elliptic integrals, exceeding the intended *Medium* difficulty. Consequently, the Critic recommends adding this constraint and further suggests including explicit nodes for partial derivatives required for surface area calculation, while leveraging symmetry to simplify the computation.

Iteration 2 The Moderator ($\mathcal{A}_M$) adopts these suggestions and provides iterative guidance to the Proposer. Based on this input, the Proposer updates the graph topology to produce Graph 2. Upon reviewing Graph 2, the Critic finds no logical contradictions but offers supplementary recommendations for optimization potential:

- Introduce a node dedicated to symmetry arguments, potentially referencing group theory for theoretical depth;
- Incorporate dimensionless scaling analysis;
- Strengthen the association between v_2 (Context) and v_4 (Domain Constraint).

Final Decision The Moderator deems these additional suggestions non-essential for the current task specification, noting they might divert the problem from its core objective. Therefore, the Moderator executes *adaptive truncation* to terminate the iteration. Graph 2 is output as the final *Constraint Graph* ($\mathcal{G}^*$), which is then passed to the Executor for semantic instantiation into the finalized mathematical problem.

Graph 0

v_1: Saddle Surface

Graph 1

v_1. Saddle Surface: The abstract geometric primitive characterized by opposing curvatures.
v_2. Potato Chip Context: Real-world physical object (e.g., Pringles) used to frame the problem.
v_3. Hyperbolic Paraboloid Equation: The specific algebraic model, e.g., $z = x^2/a^2 - y^2/b^2$.
v_4. Cylindrical Domain Constraint: The boundary condition restricting the surface to a finite disk, $x^2 + y^2 \leq R^2$.
v_5. Surface Area Integral: The double integral calculation required to find the total area of the curved surface.

e_{21}. instantiates
e_{13}. formalized_by
e_{35}. integrand_source
e_{45}. defines_limits

Graph 2

v_1: Saddle Surface: The abstract geometric primitive characterized by opposing curvatures.
v_2: Potato Chip Context: Real-world physical object (e.g., Pringles) used to frame the problem.
v_3: Hyperbolic Paraboloid Equation: Simplified symmetric model: $z = c(x^2 - y^2)$ with $c > 0$ constant, or equivalently $a = b = 1$ in the form $z = (x^2 - y^2)/a^2$.
v_4: Cylindrical Domain Constraint: The boundary condition restricting the surface to a finite disk, $x^2 + y^2 \leq R^2$.
v_5: Surface Area Integral: The double integral calculation for the total area of the curved surface. Now includes symmetry exploitation: integrate over first quadrant and multiply by 4.
v_6: Partial Derivative Calculation: Compute $\partial z/\partial x$ and $\partial z/\partial y$ for the surface area element $\mathrm{d}S = \sqrt{1 + (\partial z/\partial x)^2 + (\partial z/\partial y)^2}\,\mathrm{d}x\mathrm{d}y$.

e_{21}: instantiates
e_{13}: formalized_by
e_{36}: differentiated_to
e_{65}: integrand_source
e_{45}: defines_limits

Final Synthesized Problem

A gourmet potato chip manufacturer designs its product to follow the precise geometry of a saddle surface, known mathematically as a hyperbolic paraboloid. When centered at the origin, the surface of a single chip is modeled by the equation $z = c(x^2 - y^2)$, where c is a positive constant. To ensure uniformity, each chip is trimmed so that its vertical projection onto the xy-plane is bounded by the circle $x^2 + y^2 \leq R^2$.

By first calculating the partial derivatives $\partial z/\partial x$ and $\partial z/\partial y$ to determine the surface area element $\mathrm{d}S$, set up and evaluate a double integral to find the total surface area of the chip. In your calculation, exploit the symmetry of the surface by integrating over the first quadrant of the cylindrical domain and multiplying the result by four. Express your final answer in terms of c and R.

Figure 8: Case Study.

Prompt for Legislator - Proposer ($\mathcal{A}_P$)

You are the Proposer ($\mathcal{A}_P$) in the Legislator-Executor framework. Your objective is to drive the meta-level structural evolution of a mathematical problem by optimizing a Constraint Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.

Input Data:
- Style Tokens ($\mathcal{S}$): {STYLE_TOKENS_INPUT}
- Current Graph ($\mathcal{G}_t$): {CURRENT_GRAPH_DATA}
- Feedback: {ITERATIVE_GUIDANCE_FROM_MODERATOR}

Operational Directives:
- Evolution & Revision: Perform topological mutations to transition $\mathcal{G}_t$ to $\mathcal{G}_{t+1}$.
- Graph-Style Alignment: Expand nodes $\mathcal{V}$ and logical edges $\mathcal{E}$ to achieve the graph-related stylistic goals (particularly complexity) specified in $\mathcal{S}$.
- Consistency Maintenance: Rectify any logical contradictions identified in previous feedback.

Task Workflow:
Step 1: Internal Analysis & Planning
Analyze the gap between the current graph $\mathcal{G}_t$ and the target specifications in $\mathcal{S}$. Plan specific mutations (e.g., adding concepts, nesting operators, or refining constraints) to bridge this gap while resolving any reported flaws.
Step 2: Structured Output (JSON)
Generate the updated graph $\mathcal{G}_{t+1}$ following the strict JSON schema below.

Final Output Format:
Analysis and Planning: [Your detailed step-by-step thinking process here]
Final Optimized Graph (JSON):

```
{
  "graph_id": "G_{t+1}",
  "nodes": [{"id": "v_n", "concept": "string", "description": "string"}],
  "edges": [{"source": "v_i", "target": "v_j", "relation": "string"}],
  "mutation_log": "Summary of changes made in this iteration."
}
```

(Constraint: Ensure all referenced nodes in ‘edges‘ exist in the ‘nodes‘ list)

Figure 9: Prompt for the Proposer.

Prompt for Legislator - Critic ($\mathcal{A}_C$)

You are the Critic ($\mathcal{A}_C$). Your goal is not merely to check for correctness, but to identify the "evolutionary headroom" of the graph $\mathcal{G}_{t+1}$ to push it from functional to exceptional.

Input for Review:
- Style Tokens ($\mathcal{S}$): {STYLE_TOKENS_INPUT}
- Proposed Graph ($\mathcal{G}_{t+1}$): {PROPOSED_GRAPH_DATA}

Evaluation Dimensions:
- Internal Consistency: Scrutinize for logical contradictions or ill-defined constraints.
- Specification Alignment: Verify if the graph strictly complies with the complexity and category requirements in $\mathcal{S}$.
- Optimization Potential: Even if requirements are met, provide several actionable suggestions for potential optimization.

Final Output Format:
- Analysis: [Your detailed step-by-step thinking process here]
- Critical Flaws: [List any issues that violate consistency or $\mathcal{S}$ (Output "None" if perfect)]
- Refinement Suggestions: [Propose at least 2-3 specific actions to further optimize the graph's complexity or elegance]
- Expected Utility: [Estimate the marginal gain of these optimizations (High / Medium / Low) to assist the Moderator's decision]

Figure 10: Prompt for the Critic.

Prompt for Legislator - Moderator ($\mathcal{A}_M$)

You are the Moderator ($\mathcal{A}_M$). You adjudicate the state of graph $\mathcal{G}_{t+1}$ based on the Critic's report.

Data for Decision:
- Critic's Report: {CRITIC_REPORT}
- Style Tokens ($\mathcal{S}$): {STYLE_TOKENS_INPUT}
- Proposed Graph ($\mathcal{G}_{t+1}$): {PROPOSED_GRAPH_DATA}

Decision Logic:
- Adaptive Truncation: If $\mathcal{G}_{t+1}$ satisfies $\mathcal{S}$ and the potential for further gain is marginal, $\mathcal{A}_M$ terminates the process and outputs the graph.
- Iterative Guidance: Otherwise, direct specific modifications to the Proposer to extend structure or rectify flaws.

Final Output Format:
- Analysis: [Your detailed step-by-step thinking process here]
- Decision: [Suspend/Continue Iteration]
- Guidance for the Proposer: [If ITERATE: Provide a concise instruction list for the Proposer. If TERMINATE: Output "None"]
- Final Graph: [If TERMINATE: Output the full JSON of $\mathcal{G}_{t+1}$. If ITERATE: Output "N/A"]

Figure 11: Prompt for the Moderator

Prompt for Executor - Question Synthesizer

Your task is to perform Semantic Instantiation: converting an abstract Constraint Graph $\mathcal{G}^*$ into a high-quality, natural language mathematical problem.

Input Data:
- Style Tokens ($\mathcal{S}$): {STYLE_TOKENS_INPUT}
- Final Constraint Graph ($\mathcal{G}^*$): {FINAL_GRAPH_DATA}

Operational Directives:
- Structural Fidelity: Every node $v \in \mathcal{V}$ and edge $e \in \mathcal{E}$ must be reflected in the problem. Do not omit constraints.
- Style Alignment: The generated mathematical problem should conform to the constraints specified in the style tokens.
- Semantic Fluency: The problem must be linguistically fluid, not a robotic list of conditions. Ensure logical transitions between the situational narrative and the technical specifications.
- Output Constraint: Generate ONLY the natural language question (Q). Do not provide solutions, explanations, or meta-comments.

Final Output Format:
- Analysis: [Step-by-step plan: How to map $\mathcal{G}^*$ nodes to $\mathcal{S}$ context while maintaining fluency]
- Question: [The finalized natural language problem statement]

Figure 12: Prompt for the Question Synthesizer.

Prompt for Generating Quality of Problems

Instruction
You need to rate the quality of the math problem based on its clarity, accuracy, and logical coherence. The rating scale is as follows:

– **Very poor:** The problem description is ambiguous, conditions are incomplete, or contains logical contradictions. It lacks essential information and context required for solving, or the given instruction is not a mathematical problem.
– **Poor:** The problem is somewhat unclear or lacks important details. It requires significant clarification to define the solving requirements.
– **Average:** The problem is moderately clear and accurate but may contain imprecise expressions. Additional information might be needed for a complete solution.
– **Good:** The problem is clearly structured, with well-defined conditions and logical coherence. It provides sufficient information to support the solving process.
– **Excellent:** The problem is precisely formulated, with complete conditions and rigorous logic. It contains all necessary elements for solving without redundant information.

Math Problem to Evaluate
{math_problem}

Output Format
First, provide an assessment highlighting the strengths and/or weaknesses of the math problem. Then, output a rating by filling in the placeholders:

"explanation": "[Your assessment analysis]",
"quality": "[very poor / poor / average / good / excellent]".

Prompt for Generating Difficulty of Problems

Instruction
You are an expert in mathematics education and cognitive task analysis. Your responsibility is to evaluate the complexity of mathematical problems presented by users. For each mathematical problem, you must first identify the required knowledge points, and then assess the difficulty level based on the mathematical concepts involved, problem-solving steps, and cognitive demands.

Math Problem to Evaluate
{math_problem}

Output Format
Given the provided mathematical problem, in your output you must first determine the knowledge points required to solve it. Then, rate the difficulty level of the mathematical problem as 'very easy', 'easy', 'medium', 'hard', or 'very hard'.

Please output the difficulty level below in the following format by filling in the placeholders in [...]:

"explanation": "[Your detailed explanation and reasoning]",
"knowledge": "[list specific mathematical concepts, procedures, or knowledge domains]",
"difficulty": "[very easy / easy / medium / hard / very hard]".

Figure 13: Prompts for Evaluating Mathematical Problem Quality and Difficulty.