Benchmarking LLMs' Mathematical Reasoning with Unseen Random Variables Questions

Zijin Hong[1*], Hao Wu[2*], Su Dong[1], Junnan Dong[3], Yilin Xiao[1], Yujing Zhang[1], Zhu Wang[1]
Feiran Huang[4], Linyi Li[5], Hongxia Yang[1], Xiao Huang[1†]

[1]The Hong Kong Polytechnic University, Hong Kong SAR, China
[2]University of Electronic Science and Technology of China, Chengdu, China
[3]Tencent Youtu Lab, Shanghai, China
[4]Beihang University, Beijing, China
[5]Simon Fraser University, Burnaby, Canada
zijin.hong@connect.polyu.hk, wh.pyjnqd@gmail.com, xiao.huang@polyu.edu.hk

Abstract

Recent studies have raised significant concerns regarding the reliability of current mathematical benchmarks, highlighting key limitations such as simplistic design and potential data contamination that undermine evaluation accuracy. Consequently, developing a reliable benchmark that effectively evaluates large language models' (LLMs) genuine capabilities in mathematical reasoning remains a critical challenge. To address these concerns, we propose **RV-BENCH**, a novel evaluation methodology for **BENCH**marking LLMs with **R**andom **V**ariables in mathematical reasoning. Specifically, we develop question-generating functions to produce random variable questions (RVQs), whose background content mirrors the original benchmark problems, but with randomized variable combinations, rendering them "unseen" to LLMs. Models must completely understand the inherent question pattern to correctly answer RVQs with diverse variable combinations. Thus, an LLMs' genuine reasoning capability is reflected through its accuracy and robustness on RV-BENCH. We conducted extensive experiments on over 30 representative LLMs across more than 1,000 RVQs. Our findings reveal that LLMs exhibit a proficiency imbalance between encountered and "unseen" data distributions. Furthermore, RV-BENCH reveals that proficiency generalization across similar mathematical reasoning tasks is limited, but we verified that it can still be effectively elicited through test-time scaling.

Code — https://github.com/DEEP-PolyU/RV-Bench
Extended version — https://arxiv.org/abs/2501.11790

Introduction

The emergence of LLMs has led to impressive results across a wide range of applications, including machine translation (Zhang, Haddow, and Birch 2023), text summarization (Liu et al. 2024), and question answering (Kamalloo et al. 2023). With advancements in LLMs' reasoning capabilities (Huang and Chang 2023), their performance on complex real-world challenges such as code generation (Chen et al. 2021; Hong et al. 2025), planning (Huang et al. 2024), and especially mathematical reasoning and computation (Romera-Paredes et al. 2024) has become a central focus within the LLM research community (Zhao et al. 2023). Advanced domain-specific studies on LLMs' mathematical reasoning (Shao et al. 2024) further highlight their strong potential for addressing real-world challenges. Consequently, numerous promising methods (Luo et al. 2023) and benchmarks (Zhou et al. 2025d) have been developed to further enhance and comprehensively evaluate LLMs' performance on mathematical tasks (Mirzadeh et al. 2025).

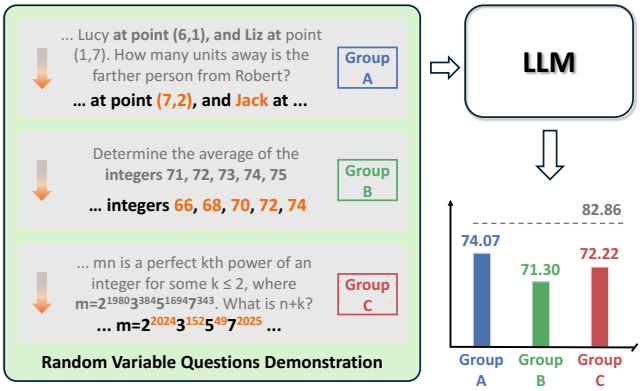

Figure 1: When mathematical problems are presented with "unseen" random variables, LLMs suffer a significant accuracy drop. This discrepancy highlights the limitations of existing evaluations of the mathematical reasoning of LLMs.

However, **are existing benchmarks for LLMs' mathematical reasoning truly reliable?** Figure 1 illustrates a discrepancy in the well-known MATH (Hendrycks et al. 2021b) dataset. In our pilot experiments, powerful LLMs like GPT-4o (Achiam et al. 2023) perform well on MATH problems but suffer a significant accuracy drop when answering questions with identical content but different variable combinations (Mirzadeh et al. 2025), as detailed in the experimental section. This discrepancy raises two critical concerns about current evaluation frameworks: **1) Existing benchmarks**

*Both authors contributed equally to this research.
†Corresponding author.

may be overly simplistic for contemporary LLMs, as they typically evaluate performance on fixed-variable problems with one-step reasoning. LLMs may not truly understand the problems but rather "guess" the correct answers (Dong et al. 2024; Mirzadeh et al. 2025) to achieve high performance; **2) Problems in widely-used benchmarks might be encountered by LLMs during training through data contamination**, enabling models to achieve high accuracy on original benchmark problems (Ni et al. 2025) without genuinely understanding the inherent question patterns. These concerns pose a significant challenge in evaluating the genuine mathematical reasoning capabilities of LLMs (Deng et al. 2024).

Advanced studies present in-depth analyses of LLMs' probabilistic modeling during the reasoning process, obscuring the fact that these models are not genuinely capable of formal reasoning (Shi et al. 2023; Jiang et al. 2024). Additionally, potential issues such as data contamination and overfitting during LLM training have been widely studied (Balloccu et al. 2024; Xu et al. 2024; Mirzadeh et al. 2025), suggesting that LLMs can "reason" simply by memorizing and replicating the same steps. Given that mathematics is a foundational domain applicable across a wide range of semantic scenarios, the growing popularity of publicly available datasets like GSM8K (Cobbe et al. 2021) and MATH (Hendrycks et al. 2021b) increases the risk of data contamination. Although recent studies on contamination detection (Chern et al. 2024; Ni et al. 2025) can signal unreliable results, they fail to reflect LLMs' genuine performance, as data contamination occurs during pretraining and remains non-intervenable (Kapoor and Narayanan 2023).

The abovementioned phenomenon raises a critical issue: existing benchmarks may fail to accurately reflect LLMs' genuine performance on mathematical tasks (Balloccu et al. 2024; Mirzadeh et al. 2025). In this context, **effectively evaluating LLMs' genuine mathematical reasoning capabilities remains a significant challenge**. As a solution, this paper introduces **RV-BENCH**, a new evaluation framework for benchmarking LLMs' mathematical reasoning through **random variable questions (RVQs)**: diverse, "unseen" questions generated with randomized variable combinations and algorithmic problems reformulated into mathematical expressions that remain out-of-distribution for LLMs' training. RV-BENCH provides an effective evaluation methodology that directly addresses the two concerns discussed above.

Specifically, we construct question-generating functions based on original problems from two selected mathematical data sources: the MATH (Hendrycks et al. 2021b) test set and the LeetCode-Math branch. These functions dynamically generate instantiated questions with random variable combinations and corresponding answers. The resulting RVQs are then collected to evaluate LLMs. Unlike existing mathematical benchmarks (Cobbe et al. 2021; Hendrycks et al. 2021b), RV-BENCH comprises questions with a wide range of variable combinations, rather than fixed, static problems. Furthermore, **RV-BENCH provides "unseen" and out-of-distribution questions, enabling LLMs to demonstrate genuine mathematical reasoning capabilities** even if they have previously encountered the original problems (Mirzadeh et al. 2025). To achieve high accuracy on RV-BENCH, an LLM must completely understand the inherent question pattern to correctly answer RVQs, thereby effectively evaluating its genuine mathematical reasoning capabilities. Overall, our contributions are listed as follows:

- We propose RV-BENCH, a leaderboard for comprehensively evaluating LLMs' genuine mathematical reasoning capabilities using four well-designed metrics. Our macroscopic analysis quantifies LLMs' understanding of inherent mathematical question patterns.
- We reveal a significant accuracy drop when LLMs solve RVQs compared to the original problems, exposing the unreliability of existing benchmarks that overlook factors such as data contamination and randomness.
- By analyzing LLMs' accuracy and robustness on RV-BENCH, we suggest LLMs acquire partial mathematical reasoning proficiency, which has limited generalization but can be effectively elicited through test-time scaling.

Related Work

The rapid advancement of LLMs has triggered the development of benchmarks for evaluating their general and domain-specific capabilities (Chang et al. 2024). General-purpose benchmarks like MMLU (Hendrycks et al. 2021a), GLUE (Wang et al. 2018) assess broad tasks such as question answering and natural language understanding, while domain-specific datasets such as GSM8K (Cobbe et al. 2021) and MATH (Hendrycks et al. 2021b) target mathematical reasoning. With the emergence of math-specialized models like Qwen-Math (Yang et al. 2024) and DeepSeek-Math (Shao et al. 2024), performance on existing math benchmarks has reached near-perfect. However, recent advanced studies like GSM-Symbolic (Mirzadeh et al. 2025) and PAL (Gao et al. 2023) reveal that such improvements may arise from pattern replication rather than truly understanding and reasoning. Our proposed RV-BENCH addresses this concern by introducing mathematical questions with random variables, challenging LLMs to beyond memorization and better evaluate their genuine reasoning capabilities. Further related works are in the extended version.

RV-BENCH Construction

Figure 2 provides a sketch workflow for RV-BENCH construction from both the MATH and LeetCode data sources. In this section, we will provide a brief introduction to the data sources and question-generating function form of RV-BENCH, and detail the process in the extended version.

Data Sources & Sampling Strategy

Two selective data sources are used for constructing RV-BENCH: the MATH test set and the LeetCode-Math branch. MATH (Hendrycks et al. 2021b) is a well-known dataset that covers 12,500 challenging mathematics problems targeted at high-school mathematics competitions. Following the preprocessing settings of the MATH by PRM800K (Lightman et al. 2024), we construct 120 question-generating functions based on randomly and uniformly selected problems from

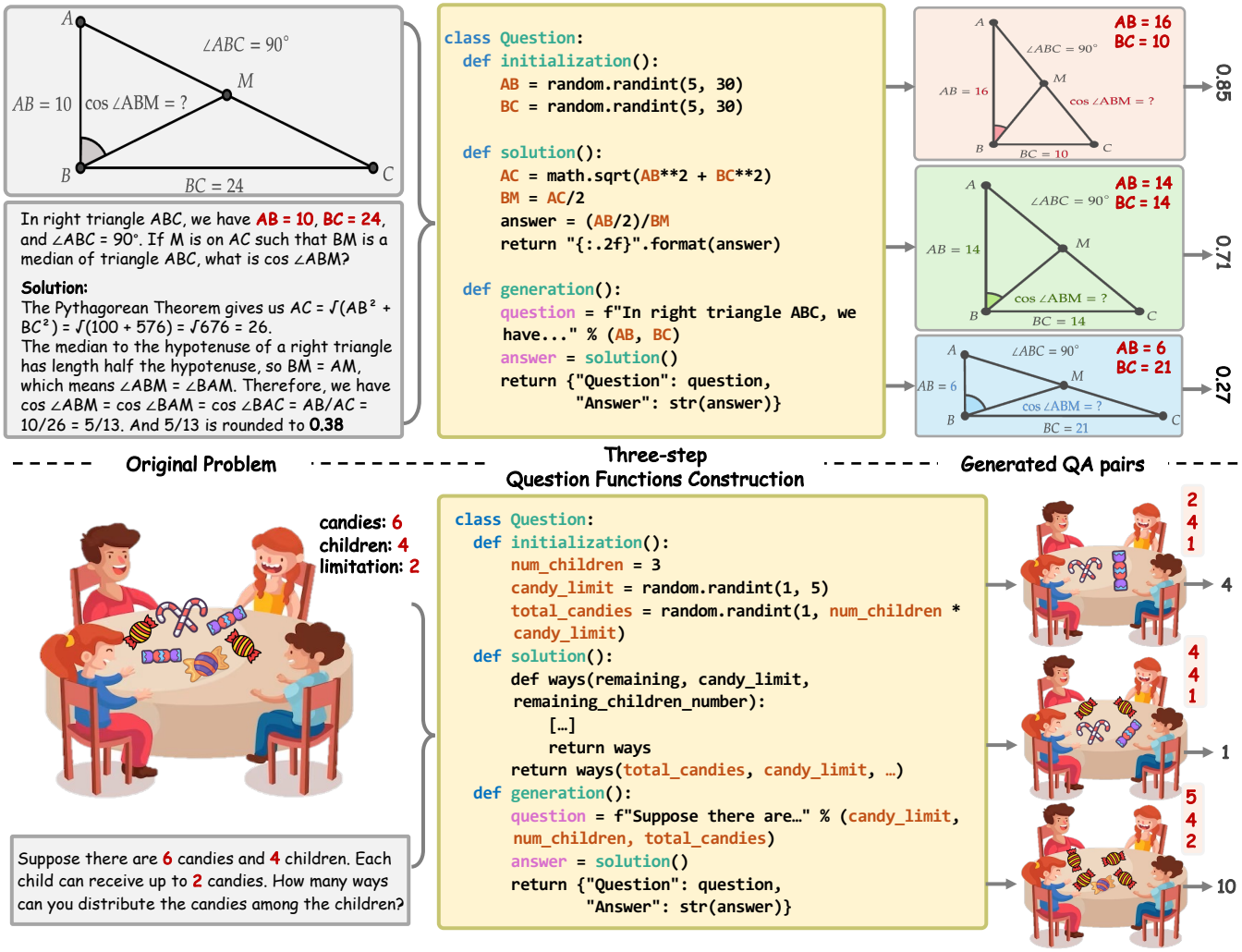

Figure 2: The workflows of RV-BENCH are illustrated for the MATH (top) and LeetCode (bottom) data sources. Each question-generating function (`Question`) comprises three modules: First, the `initialization` (`init`) module randomizes a variable combination. Next, the `solution` (`sol`) module computes the corresponding answer. Finally, the `generation` (`gen`) module outputs the instantiated question and its answer, forming a QA pair for the questions utilized in RV-BENCH evaluation.

the test split. LeetCode is a well-recognized platform providing algorithmic problems for users to practice coding skills (Coignion, Quinton, and Rouvoy 2024). LeetCode-Math is a branch that includes algorithmic problems whose content is designed based on mathematical reasoning and computation. Our motivation for selecting LeetCode as our data source comes from its original focus on coding problems. By transforming these problems into mathematical expressions, we posit that the resulting formulations are unlikely to have been encountered during LLMs' training, rendering them out-of-distribution and effectively "unseen" to the models. Through a careful review of each problem, we construct 130 question-generating functions by manually reformatting the content with random variables, selected uniformly at random. Consequently, the question-generating functions in RV-BENCH are constructed based on the respective data sources, maintaining similar distributions of difficulty, type, and bias as the original dataset.

Question Functions & Difficulty Control

As illustrated in Figure 2, a question function consists of three modules: `init`, `sol`, and `gen`. These modules are responsible for instantiating the random variables, solving the RVQs, and generating the RVQ-A pairs for RV-BENCH, respectively. To enable a fair comparison between RVQs and original problems, we implement a strict difficulty calibration for each question function. Specifically, both the numerical complexity and conceptual difficulty of each RVQ are manually controlled to ensure consistency, since empirical studies (Gao et al. 2023) indicated that increasing numerical magnitude can lead to accuracy drop. However, it primarily reflects the LLMs' limited arithmetic capability, rather than their genuine mathematical reasoning in RVQs. This tailored calibration minimizes confounding effects arising from dif-

ficulty and inconsistency in RVQs. Further details and analysis are also provided in the extended version.

Experimental Setups

Datasets. Overall, RV-BENCH comprises 230 question functions, evenly split between MATH and LeetCode-Math (115 each). To compare LLMs under random variables and original settings, we sample the corresponding original problems for each question function, denoted as **MATH-SP** and **LEETCODE-SP**, respectively. For each question function, we construct an RVQ group with **five** RVQs instantiated using varied variable combinations. In total, 1,150 RVQs are generated from 230 RVQ groups based on the manually designed question functions in RV-BENCH. These RVQs are evenly split by data source into **MATH-RV** and **LEETCODE-RV**, both of which are used throughout the paper. More details are listed in the extended version.

Evaluation Metrics. We define four metrics for the evaluation process. Given a set of RVQ groups $\mathcal{Q}_{\text{RV}} = \left\{\mathcal{G}^{(i)}\right\}_{i=1}^{m}$, let $\mathcal{Q}_{\text{SP}}$ denote the corresponding set of original problem groups. Each RVQ group generated from the i-th question function is denoted as $\mathcal{G}^{(i)} = \left(q_j^{(i)}\right)_{j=1}^{n} \in \mathcal{Q}_{\text{RV}}$, where n is the number of RVQs in the group $\mathcal{G}^{(i)}$. The original problem corresponding to $\mathcal{G}^{(i)}$ is denoted as $q_{\text{SP}}^{(i)} \in \mathcal{Q}_{\text{SP}}$. Let $\hat{a}_j^{(i)}$, $a_j^{(i)}$, $\hat{a}_{\text{SP}}^{(i)}$, and $a_{\text{SP}}^{(i)}$ denote the predicted and ground-truth answers for $q_j^{(i)}$ and $q_{\text{SP}}^{(i)}$, respectively. We further define $N_{\mathcal{G}^{(i)}}$ as the number of correctly answered RVQs associated with $\mathcal{G}^{(i)}$:

1) Exact Match Accuracy (Acc): Measures the correctness of the answer for each RVQ through strict string matching:

$$\text{Acc} = \frac{\sum_i^m \sum_j^{|\mathcal{G}^{(i)}|} \mathbb{1}\left(\hat{a}_j^{(i)} = a_j^{(i)}\right)}{m \cdot n}. \quad (1)$$

2) Group Accuracy@n (GA@n): Indicates that all n generated RVQs are answered correctly in $\mathcal{G}^{(i)}$:

$$\text{GA@}n = \frac{\sum_{i=1}^m \mathbb{1}\left(\forall q_j^{(i)} \in \mathcal{G}^{(i)}, \hat{a}_j^{(i)} = a_j^{(i)}\right)}{m}. \quad (2)$$

3) Complete Ratio (CR): Evaluate whether the original problem is answered correctly and at least 80% of the RVQs are also correct, as a measure of the model's understanding:

$$\text{CR} = \frac{\sum_{i=1}^m \mathbb{1}\left(\hat{a}_{\text{SP}}^{(i)} = a_{\text{SP}}^{(i)} \wedge N_{\mathcal{G}^{(i)}} \geq \lceil 80\% \cdot n \rceil\right)}{m}. \quad (3)$$

4) Original Only Ratio (OOR): Evaluates whether the original problem is answered correctly while at least 80% of the RVQs are incorrect, indicating the proportion of cases where the model solves the original but fails on the RVQs:

$$\text{OOR} = \frac{\sum_{i=1}^m \mathbb{1}\left(\hat{a}_{\text{SP}}^{(i)} = a_{\text{SP}}^{(i)} \wedge N_{\mathcal{G}^{(i)}} \leq \lceil 20\% \cdot n \rceil\right)}{m}. \quad (4)$$

Implementations. Following LLaMA-3 (Dubey et al. 2024), we use 4-shot prompting from Minerva (Lewkowycz et al. 2022) as few-shot examples for inference on MATH-RV and MATH-SP. Similarly, for LEETCODE-RV and LEETCODE-SP, we randomly select 4 problems out of LEETCODE-RV and manually craft step-by-step solutions to serve as the few-shot examples. All the experiments with open-source LLMs are conducted on an NVIDIA server with 8 A100 GPUs, while proprietary LLMs are accessed via APIs provided by their respective official platforms.

Model Selection. The selected models span a diverse range of LLMs, covering various sizes and families to enable comprehensive evaluation across multiple dimensions. We include open-source LLMs (Dubey et al. 2024), math-specific models (Yang et al. 2024), proprietary LLMs (Achiam et al. 2023), and large reasoning models (LRMs) (Guo et al. 2025) in RV-BENCH evaluation. The detailed list of models is provided in the extended version.

RV-BENCH Learderboard

Table 1 summarizes the performance of various LLMs on the proposed RV-BENCH. As expected from the metric definitions, the performance of a given LLM typically follows the order: Acc $\geq$ CR $\geq$ GA@5. Specifically, higher Acc and GA@5 indicate stronger performance on RVQs and greater consistency across RVQ groups, respectively. A higher CR reflects the model's more complete understanding of the reasoning process underlying both the original problem and its RV variants. Correspondingly, a higher OOR suggests that while the model may answer the original problem correctly, it fails to capture the underlying problem structure, leading to poor generalization on the RVQs.

LLMs are expected to demonstrate higher Acc, CR, and GA@5, and are preferably to have lower OOR. *Models that meet this expectation are recognized as having complete question pattern understanding and possessing genuine mathematical reasoning capabilities.* Furthermore, the generally lower GA@5 suggests that while models can solve individual instances correctly, they struggle to maintain consistency across various variable combinations. To mitigate the potential impact of numerical complexity on the RVQs' difficulty, we compared the distribution of computational errors in the RVQs with those in the original problems, as reported in the extended version. The results show that under the Random Variables setting, LLMs do not experience a significant increase in computational errors. *Accordingly, we ignore the impact of numerical complexity in the subsequent experimental analysis and conclusions in this paper.*

In detail, o3-mini and DeepSeek-R1 lead significantly in performance on RV-BENCH, highlighting their exceptional mathematical reasoning capabilities. Additionally, proprietary LRMs such as o1-mini and GLM-Zero-Preview demonstrate reliable performance. The open-source LRM QwQ-32B also achieves promising accuracy, surpassing that of renowned advanced LLMs such as GPT-4o and Claude-3.5. Large-scale chatLLMs, including Gemini-2.0-Pro, DeepSeek-V3, and Claude-3.5-Sonnet, achieve strong results, further supporting the benefits of scaling model

#	Models	Size	MATH-SP	MATH-RV				LEETCODE-SP	LEETCODE-RV				RV-BENCH
			Acc	Acc	GA@5	CR	OOR (↓)	Acc	Acc	GA@5	CR	OOR (↓)	Acc
1	o3-mini	∼	97.39	**92.52**	82.61	87.83	6.09	**82.61**	**77.57**	**61.74**	**67.83**	6.09	**85.05**
2	DeepSeek-R1	671B	**100.00**	**92.52**	**85.22**	**88.70**	6.09	80.00	72.17	52.17	57.39	**5.22**	82.35
3	o1-mini	∼	90.43	84.00	67.83	80.87	5.22	76.52	66.09	41.74	51.30	6.09	75.05
4	Gemini-2.0-Pro	∼	92.17	84.17	71.30	78.26	8.70	72.17	60.17	34.78	42.61	8.70	72.17
5	DeepSeek-v3	671B	89.57	85.04	72.17	76.52	5.22	66.09	58.26	34.78	37.39	12.17	71.65
6	GLM-Zero-Preview	∼	92.17	83.13	65.22	77.39	6.09	66.96	60.00	35.65	44.35	9.57	71.57
7	QwQ-32B-Preview	32B	91.30	83.83	60.87	79.13	5.22	62.61	58.96	30.43	42.61	7.83	71.40
8	Claude-3.5-Sonnet	∼	88.70	80.35	63.48	73.04	6.09	70.43	61.39	35.65	42.61	8.70	70.87
9	Qwen2.5-Max	∼	88.70	81.39	63.48	74.78	6.96	72.17	58.43	33.04	42.61	12.17	69.91
10	Qwen2.5-72B-It	72B	87.83	81.04	62.61	76.52	6.09	66.09	58.43	29.57	40.00	10.43	69.74
11	Qwen2.5-32B-It	32B	90.43	80.00	61.74	73.91	**4.35**	69.57	55.48	26.09	39.13	12.17	67.74
12	GLM-4-Plus	∼	86.09	77.91	53.91	71.30	6.96	66.96	55.30	26.96	38.26	14.78	66.61
13	o1-preview	∼	80.87	75.83	42.61	59.13	6.96	66.09	54.78	32.17	40.87	9.57	65.31
14	GPT-4o	∼	83.48	76.70	57.39	63.48	6.09	61.74	50.09	20.00	32.17	13.04	63.40
15	Phi-4	14B	77.39	72.00	53.04	61.74	8.70	60.00	54.78	26.96	34.78	9.57	63.39
16	Llama-3.3-70B-It	70B	83.48	74.43	52.17	62.61	9.57	60.00	45.57	18.26	22.61	15.65	60.00
17	Qwen2.5-7B-It	7B	81.74	71.65	52.17	60.00	8.70	53.91	46.78	20.87	26.09	13.04	59.22
18	Qwen2.5-Math-It	7B	87.83	72.70	51.30	62.61	12.17	54.78	37.91	10.43	17.39	14.78	55.31
19	Qwen2.5-3B-It	3B	82.61	67.65	43.48	60.00	8.70	43.48	37.04	12.17	19.13	14.78	52.35
20	Llama-3.1-70B-It	70B	73.04	62.78	39.13	50.43	9.57	57.39	40.35	14.78	23.48	15.65	51.57
21	Gemma-2-27B-It	27B	66.09	59.13	34.78	46.96	6.09	43.48	35.65	13.04	17.39	13.91	47.39
22	Phi-3-medium-4k-It	14B	64.35	53.04	24.35	35.65	11.30	50.43	37.22	8.70	19.13	13.91	45.13
23	Yi-1.5-Chat	34B	64.35	50.96	21.74	31.30	11.30	38.26	33.74	8.70	13.04	13.04	42.35
24	Phi-3-mini-4k-It	3.8B	63.48	50.26	26.09	37.39	14.78	41.74	34.26	9.57	16.52	11.30	42.26
25	Qwen2.5-7B-Base	7B	66.96	53.22	25.22	36.52	13.04	48.70	31.13	7.83	12.17	21.74	42.18
26	Gemma-2-9B-It	9B	65.22	51.30	30.43	36.52	13.04	38.26	29.91	5.22	12.17	13.91	40.61
27	GPT-3.5 Turbo	∼	60.00	48.35	20.87	30.43	11.30	35.65	31.48	9.57	14.78	12.17	39.92
28	Mathstral-7B	7B	59.17	45.22	19.13	29.57	14.78	35.65	28.70	6.96	12.17	11.30	36.96
29	Llama-3.1-8B-It	8B	57.39	46.43	25.22	30.43	16.52	33.04	27.13	6.96	10.43	15.65	36.78
30	DeepSeek-Math-It	7B	59.13	48.17	18.26	33.04	11.30	29.57	24.70	6.09	7.83	12.17	36.44
31	Mixtral-8x7B-It-v0.1	46.7B	44.35	33.22	11.30	13.91	17.39	41.74	27.65	6.09	9.57	20.00	30.44
32	Llama-3.2-3B-It	3B	50.43	36.70	15.65	22.61	14.78	38.26	23.83	5.22	9.57	20.00	30.27
33	Llama-3.1-8B-Base	8B	39.13	24.52	6.09	12.17	17.39	34.78	21.57	5.22	7.83	16.52	23.05

Table 1: The **RV-BENCH** leaderboard for various LLMs includes RVQs from **MATH-RV** and **LEETCODE-RV**. Rankings are based on **RV-BENCH Acc (overall accuracy)**, which measures the overall exact match accuracy across all RVQs in both MATH-RV and LEETCODE-RV. For intuitive comparison, we also report the accuracy on the original problems, listed under MATH-SP and LEETCODE-SP. The best and second-best results in each column are highlighted in bold and underlined, respectively. A tilde (∼) in the "Size" column indicates that the model is proprietary and its size is not publicly disclosed. For the proposed evaluation metrics: **Acc**, **GA@5**, and **CR**, higher values are better; and for **ORR** (↓), lower values are better.

size. In contrast, other open-source LLMs tend to exhibit mediocre accuracy. In summary, the performance of LLMs on RV-BENCH remains closely correlated with their underlying mathematical reasoning capabilities.

Macroscopic Analysis of RV-BENCH

We further advanced the analysis from a macroscopic perspective, considering the LLMs' accuracy on both RVQs from MATH-RV and LEETCODE-RV, as well as the original problems from MATH-SP and LEETCODE-SP. The CR and OOR metrics, reported in Table 1, evaluate the model's understanding of question patterns by verifying the consistency of accuracy. Specifically, a higher accuracy correlates with a higher CR. Leading models, such as o3-mini and DeepSeek-R1, achieve nearly 90% CR, demonstrating that they fully comprehend most of the inherent question patterns that can effectively handle the associated RVQs. For well-performing models, CR and Acc remain largely consistent, suggesting that only a small subset of question patterns eludes complete understanding. Conversely, LLMs with lower performance and fewer parameters exhibit a higher OOR and a greater variance between CR and Acc.

Since CR and OOR indicate both complete and insufficient understanding behaviors based on accuracy inconsistencies, we further quantify the degree of the LLMs' patterns understanding for comprehensive analysis when LLMs correctly answer the original problems. Specifically, we assign a pattern understanding score S to each RVQ group $\mathcal{G}^{(i)}$:

$$S_{\mathcal{G}^{(i)}} = \begin{cases} 1, & N_{\mathcal{G}^{(i)}} \geq \lceil 0.8 \cdot n \rceil \\ 0, & N_{\mathcal{G}^{(i)}} \leq \lceil 0.2 \cdot n \rceil \\ 0.5, & \text{otherwise} \end{cases}. \tag{5}$$

Different values of $S_{\mathcal{G}^{(i)}}$ reflect varying degrees of LLMs'

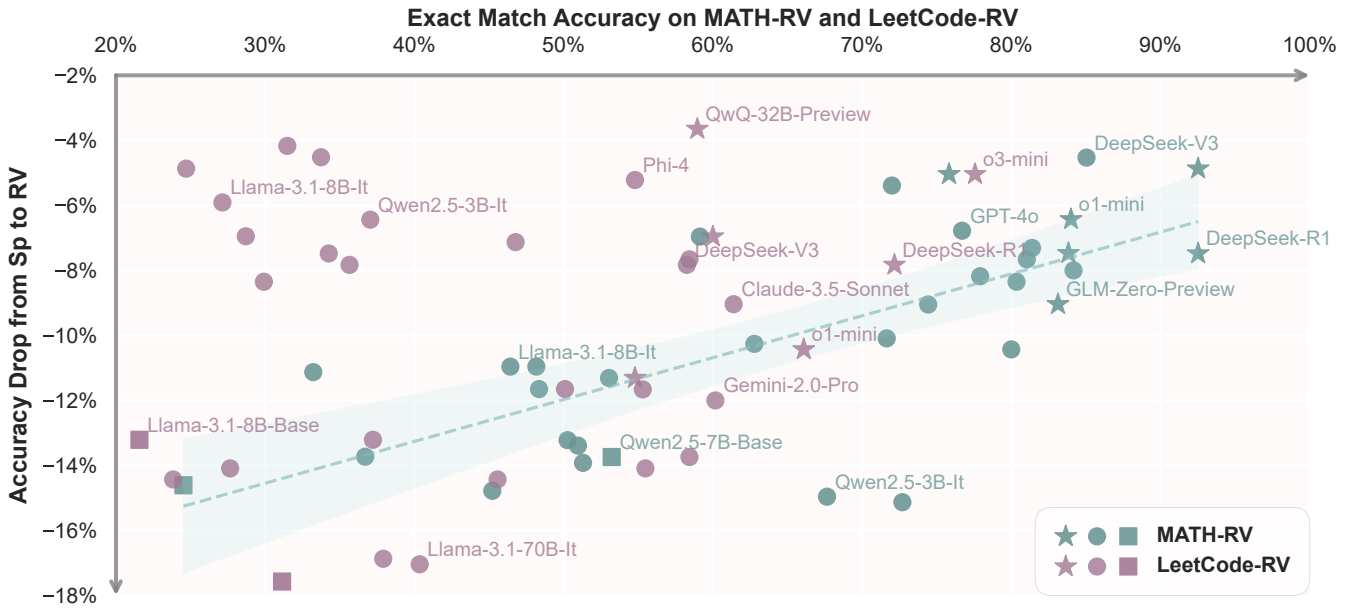

Figure 3: The accuracy drop from original problems to their corresponding RVQs is illustrated. Green data points represent the drop in accuracy from MATH-SP to MATH-RV, while pink data points represent the drop from LEETCODE-SP to LEETCODE-RV. All vertical axis values are computed as the direct difference in accuracy between the original problems and their corresponding RVQs. Different types of LLMs are indicated by distinct marker shapes, and several representative models are annotated by name for clarity. A dotted regression line is fitted using the MATH-RV data points, and the shaded region denotes the corresponding 95% confidence interval. The Pearson correlation coefficient for the green data points is $r_{\mathrm{M}} = -0.72$, indicating a strong negative correlation, while that for the pink data points is $r_{\mathrm{L}} = -0.14$ with no clear relationship.

understanding of the question pattern corresponding to $q_{\mathrm{SP}}^{(i)}$. These degrees are categorized as **complete understanding** ($S_{\mathcal{G}^{(i)}} = 1$), **partial understanding** ($S_{\mathcal{G}^{(i)}} = 0.5$), and **collapsed understanding** ($S_{\mathcal{G}^{(i)}} = 0$) of $q_{\mathrm{SP}}^{(i)}$, respectively.

Due to space considerations, the detailed visualization of the average frequency of various understanding behaviors and the average pattern understanding score of all the LLMs is provided in the extended version. We directly turn to the corresponding conclusion. Based on that analysis, we identify that models with an average score below 0.6 demonstrate significant inconsistency in their question pattern understanding. In other words, these models do not perform genuine mathematical reasoning capability on RV-BENCH.

What can be concluded from the previous observation is that: the performance of nearly all LLMs on MATH-RV is significantly better than their performance on LEETCODE-RV. One possible reason for this discrepancy is the higher task difficulty and complexity of LEETCODE-RV and LEETCODE-SP. Beyond this, we introduce another explanation based on our findings: ***the mathematical reasoning capabilities of LLMs partially depend on the data distribution involved in their training, which does not generalize well across mathematical reasoning tasks.*** Following our motivation for selecting LeetCode-related data, which is primarily utilized for enhancing coding skills and kept "unseen" for mathematical reasoning tasks. For questions in MATH-RV, although these RVQs remain new to the LLMs, it is highly likely that they have encountered MATH training sets with same-source problems within the same distribution to enhance their mathematical reasoning capabilities. Through this, LLMs can develop specific proficiency in MATH-related data. However, such proficiency is relatively scarce on LeetCode. Deducing from the performance variance, this proficiency does not generalize well, even when directly applied to similar mathematical reasoning tasks.

Accuracy Dropping in RVQs

Figure 3 illustrates the accuracy drop of various LLMs when transitioning from answering the original problems in MATH-SP and LEETCODE-SP to solving the RVQs in MATH-RV and LEETCODE-RV. Each data point in the scatter plot represents the accuracy drop of a specific LLM on a particular data distribution. Significantly, all models exhibited varying degrees of accuracy drop introduced by random variable setting, ranging from 4% to 16%. The prevalence of this accuracy dropping supports our previous concern: the existing benchmark design is overly simplistic for current LLMs. We consider that ***matching a single answer only for a fixed problem is unreliable, as it may neglect influences such as data contamination and inherent randomness, potentially introducing bias into the final results.*** In our proposed RV-BENCH, replacing variables in mathematical problems can lead to significant accuracy deviations.

For the observed data points corresponding to different question distributions, we intuitively fit a line using the MATH-related data points. When computing the correlation

coefficient between the accuracy on MATH-RV and the accuracy drop (measured as the difference between MATH-SP and MATH-RV), we obtain $r_{\mathrm{M}} = -0.72$, indicating a strong negative correlation: models with lower performance tend to exhibit larger accuracy drops. In other words, higher accuracy on MATH-RV implies better robustness and consistency across varying variable combinations in RVQs. By contrast, the correlation coefficient calculated for the LeetCode-related data points is $r_{\mathrm{L}} = -0.14$, suggesting no clear relationship between the model's accuracy on LEETCODE-RV and its robustness or consistency.

Similarly, we observe that the consistency and robustness of LLMs on RVQs in LEETCODE-RV are significantly weaker than those in MATH-RV. Beyond the possible explanation of varying task difficulty, we extend the hypothesis introduced at the end of the previous section: ***the robustness and consistency of LLMs in mathematical reasoning are also partially dependent on data distribution.*** Proficiency within a specific data distribution does not generalize well in terms of robustness and consistency to other, even similar, mathematical reasoning tasks. In conclusion, we unify these observations into a potential underlying explanation for the inconsistency: ***LLMs obtain certain proficiency in mathematical reasoning through training, but this proficiency is partially distribution-dependent. While it may apply to similar questions within the same distribution, it does not generalize reliably across others.*** This raises an important question: does such distribution-dependent proficiency truly reflect genuine mathematical reasoning capability?

Test-time Scaling Elicits Proficiency

The previous section introduced two potential explanations for the inconsistency in model accuracy and robustness across different data distributions: (1) variation in task difficulty, and (2) the distribution-dependent proficiency of LLMs. This kind of proficiency presents a critical issue: LLMs tend to replicate reasoning patterns encountered within a familiar distribution, even when faced with problems from a different, "unseen" distribution (Mirzadeh et al. 2025), leading to biased single-turn reasoning. To further investigate this phenomenon, we extend our experimental setting by applying test-time scaling, which enables LLMs to attempt each question multiple times (Brown et al. 2024). Specifically, we evaluate the LLMs' performance using pass@k metrics (Chen et al. 2021). For each mathematical question, the LLM generates P independent answers. For $1 \leq k \leq P$, the pass@k metric is defined as:

$$\text{pass@}k = \mathbb{E}_{\text{Questions}}\left[1 - \frac{\binom{P-c}{k}}{\binom{P}{k}}\right], \tag{6}$$

where c denotes the number of correctly answered questions. We set $P = 256$ and re-evaluate two selected LLMs: Llama-3.2-3B-It and GPT-3.5 Turbo using the pass@k metric.

Figure 4 illustrates the pass@k accuracy with multiple attempts. Taking LEETCODE-RV as an example, with a single attempt, Llama-3.2-3B-It achieves an accuracy of 26.67%. However, with up to 10 attempts, its pass@10 accuracy increases substantially to 56.52%. Notably, the upper bounds

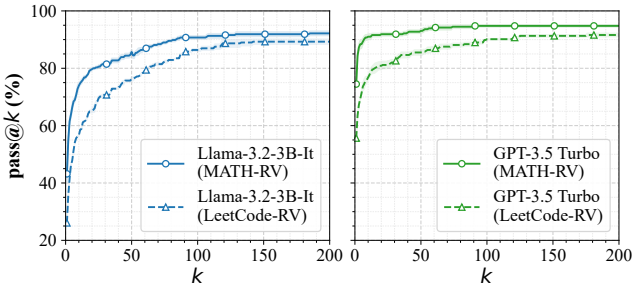

Figure 4: The accuracy evaluated using the pass@k metric. Each line represents test-time scaling performance for a given LLM. The solid line denotes accuracy on MATH-RV, while the dotted line shows accuracy on LEETCODE-RV.

of pass@k performance in LEETCODE-RV align closely with those in MATH-RV, reaching approximately 90% at pass@200. We identify the remaining 10% of questions as likely representing high-difficulty questions (further discussion is provided in the extended version) that LLMs fail to solve due to fundamental limitations in their mathematical reasoning capabilities, possibly constrained in model size. In other words, improving the performance on these high-difficulty questions remains beyond the reach of test-time scaling. Apart from task difficulty, the degree of pass@k improvement in the LeetCode distribution is noticeably greater than in MATH. We refer to this phenomenon as an ***"elicitation of proficiency generalization in mathematical reasoning tasks"***. These findings provide indirect support for our earlier hypothesis: the inconsistency between MATH-RV and LEETCODE-RV performance is more likely caused by ***LLMs' imbalance in proficiency between encountered and "unseen" data distributions. The generalization of proficiency is not well-established across similar mathematical reasoning tasks but can be elicited by test-time scaling.***

Conclusions

Motivated by significant limitations in existing mathematical reasoning benchmarks, such as simplistic design and potential data contamination, we introduce RV-BENCH, a novel evaluation methodology that leverages RVQs to more accurately evaluate LLMs' mathematical reasoning capabilities. Our findings reveal substantial accuracy drops when LLMs encounter RVQs that are "unseen" during their training, highlighting the potential unreliability of existing benchmarks in truly capturing LLM performance. While LLMs acquire partial mathematical proficiency during pre-training and fine-tuning, this proficiency is often tied to specific data distributions and exhibits limited generalization across broader mathematical tasks during evaluation. However, we further demonstrate that test-time scaling can effectively elicit this generalization. Overall, RV-BENCH provides a more reliable and effective methodology for evaluating LLMs in mathematical reasoning, offering valuable insights for advancing LLM-based mathematical reasoning research and its application to real-world challenges.

Acknowledgments

The work described in this paper was fully supported by a grant from the Innovation and Technology Commission of the Hong Kong Special Administrative Region, China (Project No. ITS/263/24FP).

References

Abdin, M.; Aneja, J.; Behl, H.; Bubeck, S.; Eldan, R.; Gunasekar, S.; Harrison, M.; Hewett, R. J.; Javaheripi, M.; Kauffmann, P.; et al. 2024. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*.

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Balloccu, S.; Schmidtová, P.; Lango, M.; and Dusek, O. 2024. Leak, Cheat, Repeat: Data Contamination and Evaluation Malpractices in Closed-Source LLMs. In *European Chapter of the Association for Computational Linguistics (EACL)*.

Brown, B.; Juravsky, J.; Ehrlich, R.; Clark, R.; Le, Q. V.; Ré, C.; and Mirhoseini, A. 2024. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*.

Chang, Y.; Wang, X.; Wang, J.; Wu, Y.; Yang, L.; Zhu, K.; Chen, H.; Yi, X.; Wang, C.; Wang, Y.; Ye, W.; Zhang, Y.; Chang, Y.; Yu, P. S.; Yang, Q.; and Xie, X. 2024. A Survey on Evaluation of Large Language Models. *ACM Transactions on Intelligent Systems and Technology (TIST)*.

Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; Pinto, H. P. D. O.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Chen, Z.; Liu, T.; Tongqing; Tian, M.; Luo, W.; and Liu, Z. 2025. Advancing Mathematical Reasoning in Language Models: The Impact of Problem-Solving Data, Data Synthesis Methods, and Training Stages. In *International Conference on Learning Representations (ICLR)*.

Chern, S.; Hu, Z.; Yang, Y.; Chern, E.; Guo, Y.; Jin, J.; Wang, B.; and Liu, P. 2024. BeHonest: Benchmarking Honesty of Large Language Models. *arXiv preprint arXiv:2406.13261*.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Coignion, T.; Quinton, C.; and Rouvoy, R. 2024. A Performance Study of LLM-Generated Code on Leetcode. In *International Conference on Evaluation and Assessment in Software Engineering (EASE)*.

Deng, C.; Zhao, Y.; Tang, X.; Gerstein, M.; and Cohan, A. 2024. Investigating Data Contamination in Modern Benchmarks for Large Language Models. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Dong, J.; Hong, Z.; Bei, Y.; Huang, F.; Wang, X.; and Huang, X. 2024. CLR-Bench: Evaluating Large Language Models in College-level Reasoning. *arXiv preprint arXiv:2410.17558*.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Gao, L.; Madaan, A.; Zhou, S.; Alon, U.; Liu, P.; Yang, Y.; Callan, J.; and Neubig, G. 2023. Pal: Program-aided language models. In *International Conference on Machine Learning (ICML)*.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2021a. Measuring Massive Multitask Language Understanding. In *International Conference on Learning Representations (ICLR)*.

Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; and Steinhardt, J. 2021b. Measuring Mathematical Problem Solving With the MATH Dataset. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Hong, Z.; Yuan, Z.; Chen, H.; Zhang, Q.; Huang, F.; and Huang, X. 2024. Knowledge-to-SQL: Enhancing SQL Generation with Data Expert LLM. In *Findings of Association for Computational Linguistics (ACL)*.

Hong, Z.; Yuan, Z.; Zhang, Q.; Chen, H.; Dong, J.; Huang, F.; and Huang, X. 2025. Next-Generation Database Interfaces: A Survey of LLM-based Text-to-SQL. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*.

Huang, J.; and Chang, K. C.-C. 2023. Towards Reasoning in Large Language Models: A Survey. In *Findings of Association for Computational Linguistics (ACL)*.

Huang, X.; Liu, W.; Chen, X.; Wang, X.; Wang, H.; Lian, D.; Wang, Y.; Tang, R.; and Chen, E. 2024. Understanding the planning of LLM agents: A survey. *arXiv preprint arXiv:2402.02716*.

Jiang, B.; Xie, Y.; Hao, Z.; Wang, X.; Mallick, T.; Su, W. J.; Taylor, C. J.; and Roth, D. 2024. A Peek into Token Bias: Large Language Models Are Not Yet Genuine Reasoners. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Kamalloo, E.; Dziri, N.; Clarke, C.; and Rafiei, D. 2023. Evaluating Open-Domain Question Answering in the Era of Large Language Models. In *Association for Computational Linguistics (ACL)*.

Kapoor, S.; and Narayanan, A. 2023. Leakage and the Reproducibility Crisis in ML-based Science. *Patterns*.

Lewkowycz, A.; Andreassen, A. J.; Dohan, D.; Dyer, E.; Michalewski, H.; Ramasesh, V. V.; Slone, A.; Anil, C.; Schlag, I.; Gutman-Solo, T.; Wu, Y.; Neyshabur, B.; Gur-Ari, G.; and Misra, V. 2022. Solving Quantitative Reasoning Problems with Language Models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Lightman, H.; Kosaraju, V.; Burda, Y.; Edwards, H.; Baker, B.; Lee, T.; Leike, J.; Schulman, J.; Sutskever, I.; and Cobbe, K. 2024. Let's Verify Step by Step. In *International Conference on Learning Representations (ICLR)*.

Liu, S.; Liu, H.; Liu, J.; Xiao, L.; Gao, S.; Lyu, C.; Gu, Y.; Zhang, W.; Wong, D. F.; Zhang, S.; and Chen, K. 2025. CompassVerifier: A Unified and Robust Verifier for LLMs Evaluation and Outcome Reward. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Liu, Y.; Shi, K.; He, K.; Ye, L.; Fabbri, A.; Liu, P.; Radev, D.; and Cohan, A. 2024. On Learning to Summarize with Large Language Models as References. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Luo, H.; Sun, Q.; Xu, C.; Zhao, P.; Lou, J.; Tao, C.; Geng, X.; Lin, Q.; Chen, S.; and Zhang, D. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*.

Mirzadeh, I.; Alizadeh, K.; Shahrokhi, H.; Tuzel, O.; Bengio, S.; and Farajtabar, M. 2025. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. In *International Conference on Learning Representations (ICLR)*.

Ni, S.; Kong, X.; Li, C.; Hu, X.; Xu, R.; Zhu, J.; and Yang, M. 2025. Training on the Benchmark Is Not All You Need. In *Conference on Artificial Intelligence (AAAI)*.

Romera-Paredes, B.; Barekatain, M.; Novikov, A.; Balog, M.; Kumar, M. P.; Dupont, E.; Ruiz, F. J.; Ellenberg, J. S.; Wang, P.; Fawzi, O.; et al. 2024. Mathematical discoveries from program search with large language models. *Nature*.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y.; Wu, Y.; et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Shi, F.; Chen, X.; Misra, K.; Scales, N.; Dohan, D.; Chi, E.; Schärli, N.; and Zhou, D. 2023. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning (ICML)*.

Toshniwal, S.; Moshkov, I.; Narenthiran, S.; Gitman, D.; Jia, F.; and Gitman, I. 2024. OpenMathInstruct-1: A 1.8 Million Math Instruction Tuning Dataset. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *EMNLP 2018 BlackboxNLP Workshop (EMNLP)*.

Wang, P.; Li, L.; Shao, Z.; Xu, R.; Dai, D.; Li, Y.; Chen, D.; Wu, Y.; and Sui, Z. 2024. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. In *Association for Computational Linguistics (ACL)*.

Xiang, Z.; Wu, C.; Zhang, Q.; Chen, S.; Hong, Z.; Huang, X.; and Su, J. 2025. When to use graphs in rag: A comprehensive analysis for graph retrieval-augmented generation. *arXiv preprint arXiv:2506.05690*.

Xu, C.; Guan, S.; Greene, D.; Kechadi, M.; et al. 2024. Benchmark Data Contamination of Large Language Models: A Survey. *arXiv preprint arXiv:2406.04244*.

Yang, A.; Zhang, B.; Hui, B.; Gao, B.; Yu, B.; Li, C.; Liu, D.; Tu, J.; Zhou, J.; Lin, J.; et al. 2024. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

Yuan, Z.; Chen, H.; Hong, Z.; Zhang, Q.; Huang, F.; Li, Q.; and Huang, X. 2025. Knapsack Optimization-based Schema Linking for LLM-based Text-to-SQL Generation. In *International Conference on Data Engineering (ICDE)*.

Zhang, B.; Haddow, B.; and Birch, A. 2023. Prompting large language model for machine translation: a case study. In *International Conference on Machine Learning (ICML)*.

Zhang, Q.; Chen, S.; Bei, Y.; Yuan, Z.; Zhou, H.; Hong, Z.; Chen, H.; Xiao, Y.; Zhou, C.; Dong, J.; et al. 2025a. A survey of graph retrieval-augmented generation for customized large language models. *arXiv preprint arXiv:2501.13958*.

Zhang, Q.; Dong, J.; Chen, H.; Zha, D.; Yu, Z.; and Huang, X. 2024. Knowgpt: Knowledge graph based prompting for large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Zhang, Q.; Xiang, Z.; Xiao, Y.; Wang, L.; Li, J.; Wang, X.; and Su, J. 2025b. FaithfulRAG: Fact-Level Conflict Modeling for Context-Faithful Retrieval-Augmented Generation. In *Association for Computational Linguistics (ACL)*.

Zhao, W. X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Zhong, T.; Liu, Z.; Pan, Y.; Zhang, Y.; Zhou, Y.; Liang, S.; Wu, Z.; Lyu, Y.; Shu, P.; Yu, X.; et al. 2024. Evaluation of openai o1: Opportunities and challenges of agi. *arXiv preprint arXiv:2409.18486*.

Zhou, C.; Wang, Z.; Chen, S.; Du, J.; Zheng, Q.; Xu, Z.; and Huang, X. 2025a. Taming Language Models for Text-attributed Graph Learning with Decoupled Aggregation. In *Association for Computational Linguistics (ACL)*.

Zhou, H.; Du, J.; Zhou, C.; Yang, C.; Xiao, Y.; Xie, Y.; and Huang, X. 2025b. Each Graph is a New Language: Graph Learning with LLMs. In *Findings of Association for Computational Linguistics (ACL)*.

Zhou, H.; Yu, K.; Zhang, Q.; Chen, H.; Zha, D.; Pei, W.; Kong, A.; and Huang, X. 2025c. Self-Monitoring Large Language Models for Click-Through Rate Prediction. *ACM Transactions on Information Systems (TOIS)*.

Zhou, Z.; Liu, S.; Ning, M.; Liu, W.; Wang, J.; Wong, D. F.; Huang, X.; Wang, Q.; and Huang, K. 2025d. Is Your Model Really A Good Math Reasoner? Evaluating Mathematical Reasoning with Checklist. In *International Conference on Learning Representations (ICLR)*.

Zhuang, D.; Zhang, X.; Song, S.; and Hooker, S. 2022. Randomness in neural network training: Characterizing the impact of tooling. In *Machine Learning and Systems (MLSys)*.