Exploring the Compositional Deficiency of Large Language Models in Mathematical Reasoning Through Trap Problems

Jun Zhao[1*] Jingqi Tong[1*] Yurong Mou[1] Ming Zhang[1]
Qi Zhang[1,2†] Xuanjing Huang[1,2]

[1]School of Computer Science, Fudan University

[2]Shanghai Key Laboratory of Intelligent Information Processing, Fudan University

{zhaoj19,qz}@fudan.edu.cn, jqtong23@m.fudan.edu.cn

Abstract

Human cognition exhibits systematic compositionality, the algebraic ability to generate infinite novel combinations from finite learned components, which is the key to understanding and reasoning about complex logic. In this work, we investigate the compositionality of large language models (LLMs) in mathematical reasoning. Specifically, we construct a new dataset MATHTRAP[‡] by introducing carefully designed logical traps into the problem descriptions of MATH and GSM8K. Since problems with logical flaws are quite rare in the real world, these represent "unseen" cases to LLMs. Solving these requires the models to systematically compose (1) the mathematical knowledge involved in the original problems with (2) knowledge related to the introduced traps. Our experiments show that while LLMs possess both components of requisite knowledge, they do not spontaneously combine them to handle these novel cases. We explore several methods to mitigate this deficiency, such as natural language prompts, few-shot demonstrations, and fine-tuning. Additionally, we test the recently released OpenAI o1 model and find that human-like 'slow thinking' helps improve the compositionality of LLMs. Overall, systematic compositionality remains an open challenge for large language models.

1 Introduction

Humans excel at learning fundamental concepts and skills, systematically combining them to solve new problems. For instance, when a person possesses (a) the knowledge of how to solve quadratic equations with one variable, and (b) the understanding of what integers are, they can combine these two domains of knowledge to tackle the problem "Find the integer solutions of $x^2 + x = 3$." They would first solve the equation, and then determine whether the obtained solutions are integers or not. Fodor and Pylyshyn (1988) had a famous viewpoint that artificial neural networks lack this compositionality, and thus cannot serve as reliable cognitive models. Current LLMs have achieved unprecedented success on tasks requiring complex reasoning [Guo et al.2024, Toshniwal et al.2024]. We wonder whether compositionality still poses a significant challenge for LLMs?[1]

Toward this goal, we construct a new MATHTRAP dataset by introducing carefully designed logical traps into the original problems of the MATH [Hendrycks et al.2021] and GSM8K [Cobbe et al.2021] datasets. For example, by modifying the original problem "Find the solution of the equation $x^2 + x = 3$" to "Find the integer solution of the equation $x^2 + x = 3$," the model needs to combine (a) the knowledge involved in the original problem (how to solve quadratic equations with one variable) and (b) the knowledge about the trap (the definition of integers) to handle these trap problems (in fact, the original equation has no integer solutions). Another reason for evaluating compositionality through trap problems is that these problems rarely appear in the real world, so it is unlikely that LLMs provide the correct answers solely by following the trained reasoning paths.

We conduct comprehensive tests on leading LLMs and recruit 43 undergraduate students from top universities as human controls. We find that LLMs and humans exhibit strikingly different behavioral patterns when dealing with

[1]Equal Contributions.

[2]Corresponding authors.

[3]https://github.com/tongjingqi/MathTrap

[1]Due to space constraints, we provide a detailed description of existing research on LLMs' ability to learn and combine knowledge in Appendix A.1: Related Work.

trap problems. Despite possessing both (a) and (b) knowledge components, LLMs fail to spontaneously compose them to handle trap problems, while humans can. This suggests that tasks requiring compositional generalization remain challenging for current LLMs. Furthermore, the ability of well-aligned LLMs to handle trap problems can be elicited through external interventions, such as natural language prompts, few-shot demonstrations, and supervised fine-tuning. Furthermore, we find that the human-like 'slow thinking' demonstrated by OpenAI's o1 [OpenAI2024] also helps improve the compositionality of LLMs. Nevertheless, systematic compositionality remains an open challenge for current LLMs.

The contributions of this work are threefold: (1) We investigate the compositional generalization of LLMs on mathematical reasoning, and demonstrate their stark under performance compared to humans. (2) An effective method to construct 'unseen problems' by introducing traps into original problems, and a dataset called MATHTRAP that cannot be solved by simply following the reasoning paths seen during training. (3) Comprehensive experiments exploring the impact of model size, the degree of alignment, and external interventions on performance on the MATHTRAP.

2 Background and Definition

In this section, we provide the definition of compositionality discussed in this paper, based on Hilbert's formal deductive systems [Hilbert1922]:

[Hilbert's Formal Deductive System] This system consists of (1) a syntax G, specifying which derivation statements are legal, (2) a set of inference rules $R^{\dagger}$, clearly stating how new facts (or theorems) can be derived from existing facts (axioms or already proved theorems), and (3) axioms: a predetermined set A of established facts.

[Compositionality in Mathematical Reasoning] Suppose problem sets Q_1, Q_2, Q_3 are described using the same syntax G, and Q_1 and Q_2 can derive final answers through tuple (R_1, A_1), (R_2, A_2) respectively, while Q_3 requires reasoning using $R_1 \cup R_2$ based on $A_1 \cup A_2$ (or a subset) to derive the final answer. If a reasoning engine can solve Q_1 and Q_2, we say it possesses compositionality if it can solve Q_3.

† We refer to the axioms and inference rules as knowledge.

3 The MATHTRAP Dataset

Existing datasets for evaluating compositionality are limited to symbolic reasoning with semantic decoupling [Lake and Baroni2023, Dziri et al.2023]. However, the semantics of language play a crucial role in the reasoning process of LLMs [Tang et al.2024]. Our MATHTRAP dataset aims to evaluate the compositionality of LLMs on semantically rich math word problems. More importantly, the 'unseen' feature of our dataset prevents models from simply following the trained reasoning paths to arrive at solutions.

3.1 Dataset Composition

Sample Composition: As illustrated in Table 1, each sample in MATHTRAP can be viewed as a problem triplet:

1. **Original problem**: Sampled from the MATH and GSM8K datasets. These problems are used to evaluate the model's grasp of math knowledge from these datasets.

2. **Concept Problem**: Manually crafted to assess the model's understanding of the trap concepts to be introduced. These problems are intentionally simple, requiring only knowledge of the trap concept to solve.

3. **Trap Problem**: Created by manually introducing logical traps into the original problems. These are designed to evaluate the model's compositional generalization ability. Solving these problems requires the model to systematically combine knowledge from the original math problem with the introduced trap concept.

The MATHTRAP dataset consists of two subsets: Public and Private. The Public subset contains 105 problem triplets. Using GPT-4, we paraphrase these samples to expand the dataset to 1,000 problem triplets, which are used in all fine-tuning experiments discussed in this paper. We manually verify the quality of the subset. Please refer to Table **??** for the templates used in paraphrasing. We have made this subset publicly available to facilitate community evaluation of compositionality.

The Private subset comprises 155 problem triplets, and the evaluation results presented in this paper are based on this subset. This portion of the data will not be made public to mitigate the risk of data leakage.[‡] See Appendix B for more annotation details about MATHTRAP.

Problem Topic Composition: MATHTRAP comprises problems from two main sources: 15.5% from the GSM8K dataset and 84.5% from the MATH dataset. It covers a diverse range of mathematical topics, including algebra (23.2%), counting and probability (22.6%), geometry (16.1%), prealgebra (12.3%), number theory (7.74%), and precalculus (2.58%).

Trap Categories: We carefully designed five categories of traps for constructing the MATHTRAP dataset. These include:

1. **Concept Undefined**: The reasoning process involves undefined mathematical concepts (such as $\tan 90^\circ$, 0 as a divisor, etc.).
2. **Missing Condition**: Lacking the necessary conditions required to solve the problem.
3. **Direct Contradiction**: Two conditions in the problem description directly contradict each other, which can be discovered without complex calculations.
4. **Indirect Contradiction**: There are indirect contradictions in the problem description, which can only be discovered during the reasoning process.
5. **Violating Common Sense**: The condition or final answer violates common sense.

3.2 Evaluation Protocol

We use accuracy as the evaluation metric for the problem. To measure compositional generalization, we calculate the ratio between the accuracy on trap questions and the accuracy on original questions. A model demonstrating good compositional generalization should exhibit similar performance before and after the introduction of trap questions, rather than showing degradation.

For Original Problems, following prior work [Yu et al.2023, Gou et al.2024, Xi et al.2024, He et al.2024], we calculate accuracy by determining whether the model's final answer matches the standard answer. For Trap Problems and Conceptual Problems, we additionally check the intermediate steps of the model's response to determine whether it correctly identified the trap.

We employ GPT-4 as the judge for the checks. In experiments where GPT-4 might exhibit bias, we supplement our evaluation with results using Claude-3.5-Sonnet as an additional judge. We provide the prompts used for the evaluation in Table 6 in the Appendix.

4 Results and Discussion

4.1 The Compositionality of LLMs

We evaluate the compositionality of LLMs on the MATHTRAP dataset, with results shown in Table 2. Proprietary LLMs achieve over 70% accuracy on these Conceptual Problems, with OpenAI o1 even reaching 96.2%. This indicates that LLMs possess the knowledge required to identify most traps. However, when comparing LLM performance on original versus trap problems, we observe a significant decline. Most proprietary LLMs achieve less than half their original accuracy on trap problems. This indicates that even advanced, well-aligned LLMs struggle to apply trap-related knowledge flexibly to novel reasoning paths. Notably, o1-preview (Web) achieved a ratio of 77.4, significantly higher than GPT-4's 51.2. This suggests that o1's test-time scaling, akin to human 'slow thinking', effectively improves LLM compositionality. However, it still falls short of the human ratio of 85.9. For detailed information on the human evaluation, please refer to Section 4.2.

Open-source models' ratios below 20 indicate that focusing solely on GSM8K and MATH accuracy doesn't truly enhance LLM reasoning abilities. Nevertheless, extensive pre-training (Llemma-MetaMath-7b vs. MetaMath-7b) and larger model scales (from 7b to 70b) still yield better compositional generalization effects (ratio increasing from 5.84% to 19%).

Table 1: Overview of the MATHTRAP Dataset. The first column represents the five introduced trap types and their percentages in the dataset. The yellow highlighted text emphasizes the difference in problem descriptions before and after introducing traps. Additionally, we annotate Conceptual Problems to test whether models possess trap-related knowledge. We hope that if a model can accurately answer both the Original Problems and the Conceptual Problems, it will also be able to accurately answer the Trap Problems. Appendix section 3.1 provides definitions of the trap types, and Table 7 offers explanations for these 5 example traps. We have included GPT-4-0125-preview's responses to selected problem from the table in Appendix Tables **??**–**??**.

Trap Type (%)	**Original Problem**	**Trap Problem**	**Conceptual Problem**
Concept Undefined (16%)	In right triangle XYZ with $\angle YXZ = 90°$, $XY = 24$ and $YZ = 25$. Find $\tan Y$.	In right triangle XYZ with $\angle YXZ = 90°$, $XY = 24$ and $YZ = 25$. Find $\tan X$.	Does $\tan 90°$ exist?
Missing Condition (6%)	Natalia sold 48 clips in April and half as many clips in May. How many clips did Natalia sell altogether in April and May?	Natalia sold 48 clips in April and half as many clips in May. How many clips did Natalia sell altogether in April and June?	Given the sales figures for May and June, can the sales figures for April and June be calculated?
Direct Contradiction (24%)	An equilateral triangle has a perimeter of 30 centimeters. Calculate its area.	An equilateral triangle has a perimeter of 30 centimeters and a height of 10 centimeters. Calculate its area.	Can the height of an equilateral triangle be equal to its side length?
Indirect Contradiction (38%)	Find the solution of the equation $x^2 + x = 3$.	Find the integer solution of the equation $x^2 + x = 3$.	Is $\sqrt{13}$ an integer?
Violating Common Sense (15%)	Max picks 2 different cards without replacement from a standard 52-card deck. What is the probability that the cards are of different suits?	Max picks 5 different cards without replacement from a standard 52-card deck. What is the probability that the cards are of different suits?	Is it possible to pick five different suits of cards from a standard deck of playing cards?

4.2 The Compositionality of Human

As a control experiment, we evaluate human performance on the MATHTRAP dataset. Specifically, we recruit 43 undergraduate students majoring in science and engineering from top universities to participate in the experiment. Each student is randomly assigned two problems: one original problem and one trap problem.[§] During the answering process, participants are not aware that the assigned problems might contain traps. To prevent participants from discovering the traps by comparing the two problems, the original problem and the trap problem assigned to each participant are completely different. The results are shown in Table 3. Humans achieve an accuracy of 83.8% on the trap problems. In terms of the ratio of accuracy on trap problems to original problems, humans attained an accuracy ratio of $83.8/97.6 = 85.9\%$, far surpassing all existing LLMs. This indicates that humans demonstrate strong compositional reasoning ability on the MATHTRAP dataset. For cases where participants fail to correctly identify the traps, we further inform them that the problems might contain traps and ask them to answer again. After receiving this hint, the human accuracy rate increases from 83.8% to 95.1%, almost identical to their performance on the original problems.

[§] We only assign two problems because once participants discover that a problem contain a trap, they would consciously look for traps in subsequent problems, which would affect the accuracy of the results.

4.3 Mitigating LLMs' Failure on MathTrap

The results in Table 2 show that LLMs have acquired the relevant knowledge needed to solve trap problems, but are unable to spontaneously apply this knowledge to reasoning on trap problems. Therefore, we attempt to mitigate this

Table 2: Accuracy (%) of various models on three types of MATHTRAP problems. 'Conceptual' represents Conceptual problems, 'Original' refers to the original problems, and 'Trap' denotes the trap problems. 'Ratio' refers to the ratio of the accuracy on Trap problems to the accuracy on Original problems. It reflects the degree to which the performance is maintained when facing problems with traps, relative to the original problems.

Model	Conceptual	Original	Trap	Ratio
Gemini-Pro	70.0	36.9	8.30	22.5
Claude3-Opus	87.7	68.5	19.0	27.7
Claude-3.5-Sonnet	93.9	75.0	19.4	25.9
GPT-3.5-turbo-0125	74.6	40.5	7.60	18.8
GPT-4-0125-preview	90.0	70.3	36.0	51.2
o1-preview (API)	96.2	88.3	38.1	43.1
o1-preview (Web)	92.3	87.5	67.7	77.4
Kimi	71.5	46.1	19.6	42.5
Llemma-MetaMath-7B	55.2	41.4	6.40	15.5
MetaMath-7B	43.2	32.5	1.90	5.84
MetaMath-13B	37.8	37.5	3.90	10.4
MetaMath-70B	57.6	34.2	6.50	19.0
Llama3-8B	70.5	33.3	6.45	19.4
Llama3-8B-Base	44.7	33.3	6.45	19.4
Llama3-70B	88.5	61.7	7.74	12.5
Llama3-70B-Base	53.8	37.5	7.74	20.6
Llama3.1-8B	70.8	61.7	13.5	21.9
Llama3.1-70B	88.5	69.2	19.4	28.0

Table 3: Human accuracy (%) on MATHTRAP. "Trap Problem (w/o Notice)" refers to the accuracy of human solutions when unaware that the problems contain traps. "Trap Problem (w/ Notice)" indicates the accuracy of human solutions when informed that the problems contain traps. "Original Problem" refers to the accuracy of human solutions on the original problems.

Condition	Human Accuracy
Trap Problem (w/o Notice)	83.8
Trap Problem (w/ Notice)	95.1
Original Problem	97.6

issue through external interventions, with the specific results presented in Tables 4 and 5. The GPT series models constitute a significant subset of proprietary models in Table 4. To mitigate potential bias in evaluating these models using GPT-4, we supplement our assessment with results using Claude-3.5-Sonnet as a judge.

Natural language prompt: Adding the prompt "Note that this problem may be unsolvable or has no solution" before the problem statement. Refer to Table **??** for the full prompt. Our results show that natural language prompts can guide LLMs to notice the contradictions or traps in the problem descriptions without affecting the accuracy on "Original Problems", especially for those well-aligned closed-source LLMs.

Few-shot demonstration: In the 1-shot setting, a randomly sampled trap problem and its reasoning process are inserted into the context (refer to Table **??** for the full prompt); in the 5-shot setting, 2 original problems and 3 trap problems with their reasoning processes are inserted (refer to Table **??**). The results show that compared to natural language prompts, few-shot demonstrations are more effective in handling trap problems. Additionally, in the 5-shot setting with a mix of original problems, the accuracy on original problems is also improved.

Fine-tuning: We use the MATHTRAP public subset containing 1,000 problem triplets for fine-tuning. Given that GPT-4 was employed for data augmentation, we included additional evaluation results using Claude-3.5-Sonnet in Table 5 to avoid potential assessment bias from using GPT-4 as both an augmenter and judge. The experiments demonstrate that fine-tuning can significantly improve model performance on trap problems without prompt, but it may also reduce the accuracy of solving original problems.

5 Conclusions

This paper investigates the compositional generalization of LLMs in mathematical reasoning. By introducing traps into the problem descriptions, we construct novel "Trap Problems" that cannot be solved by merely following trained reasoning paths for LLMs. Experiments on MATHTRAP demonstrate that LLMs fail to spontaneously combine their learned knowledge to reason about trap problems. Although this limitation can be mitigated through external interventions, there remains a significant gap compared to the compositional generalization capabilities of humans.

Limitations

The construction of trap problems places high demands on the annotators' abilities, resulting in high annotation costs. Therefore, how to automatically generate high-quality trap problems through automated methods is worthy of further investigation.

References

[Anil et al.2022] Cem Anil, Yuhuai Wu, Anders Andreassen, Aitor Lewkowycz, Vedant Misra, Vinay Ramasesh, Ambrose Slone, Guy Gur-Ari, Ethan Dyer, and Behnam Neyshabur. 2022. Exploring length generalization in large language models. Preprint, arXiv:2207.04901.

[Azerbayev et al.2024] Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen McAleer, Albert Q. Jiang, Jia Deng, Stella Biderman, and Sean Welleck. 2024. Llemma: An open language model for mathematics. Preprint, arXiv:2310.10631.

[Bian et al.2024] Ning Bian, Xianpei Han, Le Sun, Hongyu Lin, Yaojie Lu, Ben He, Shanshan Jiang, and Bin Dong. 2024. Chatgpt is a knowledgeable but inexperienced solver: An investigation of commonsense problem in large language models. Preprint, arXiv:2303.16421.

[Bubeck et al.2023] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. Preprint, arXiv:2303.12712.

[Cobbe et al.2021] 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. CoRR, abs/2110.14168.

[Dziri et al.2023] Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D. Hwang, Soumya Sanyal, Sean Welleck, Xiang Ren, Allyson Ettinger, Zaid Harchaoui, and Yejin Choi. 2023. Faith and fate: Limits of transformers on compositionality. Preprint, arXiv:2305.18654.

[Fodor and Pylyshyn1988] Jerry A Fodor and Zenon W Pylyshyn. 1988. Connectionism and cognitive architecture: A critical analysis. Cognition, 28(1-2):3–71.

[Gou et al.2024] Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Minlie Huang, Nan Duan, and Weizhu Chen. 2024. Tora: A tool-integrated reasoning agent for mathematical problem solving. Preprint, arXiv:2309.17452.

[Guo et al.2024] Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y. Wu, Y. K. Li, Fuli Luo, Yingfei Xiong, and Wenfeng Liang. 2024. Deepseek-coder: When the large language model meets programming – the rise of code intelligence. Preprint, arXiv:2401.14196.

[He et al.2024] Wei He, Shichun Liu, Jun Zhao, Yiwen Ding, Yi Lu, Zhiheng Xi, Tao Gui, Qi Zhang, and Xuanjing Huang. 2024. Self-demos: Eliciting out-of-demonstration generalizability in large language models. arXiv preprint arXiv:2404.00884.

[Hendrycks et al.2021] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. CoRR, abs/2103.03874.

[Hilbert1922] David Hilbert. 1922. Grundlagen der geometrie.

[Hosseini et al.2022] Arian Hosseini, Ankit Vani, Dzmitry Bahdanau, Alessandro Sordoni, and Aaron Courville. 2022. On the compositional generalization gap of in-context learning. In Proceedings of the Fifth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, pages 272–280, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

[Hu et al.2024] Yi Hu, Xiaojuan Tang, Haotong Yang, and Muhan Zhang. 2024. Case-based or rule-based: How do transformers do the math? Preprint, arXiv:2402.17709.

[Kazemi et al.2023] Mehran Kazemi, Najoung Kim, Deepti Bhatia, Xin Xu, and Deepak Ramachandran. 2023. LAMBADA: Backward chaining for automated reasoning in natural language. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 6547–6568, Toronto, Canada. Association for Computational Linguistics.

[Koralus and Wang-Maścianica2023] Philipp Koralus and Vincent Wang-Maścianica. 2023. Humans in humans out: On gpt converging toward common sense in both success and failure. Preprint, arXiv:2303.17276.

[Lake and Baroni2023] Brenden Lake and Marco Baroni. 2023. Human-like systematic generalization through a meta-learning neural network. Nature, 623.

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

[Miao et al.2020] Shen-yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A diverse corpus for evaluating and developing English math word problem solvers. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 975–984, Online. Association for Computational Linguistics.

[OpenAI2023] OpenAI. 2023. Gpt-4 technical report. Preprint, arXiv:2303.08774.

[OpenAI2024] OpenAI. 2024. Introducing openai o1. https://openai.com/o1/. 2024-10-02.

[Patel et al.2021] Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 2080–2094, Online. Association for Computational Linguistics.

[Sanyal et al.2022] Soumya Sanyal, Zeyi Liao, and Xiang Ren. 2022. RobustLR: A diagnostic benchmark for evaluating logical robustness of deductive reasoners. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 9614–9631, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

[Tang et al.2024] Xiaojuan Tang, Zilong Zheng, Jiaqi Li, Fanxu Meng, Song-Chun Zhu, Yitao Liang, and Muhan Zhang. 2024. On the paradox of generalizable logical reasoning in large language models.

[Toshniwal et al.2024] Shubham Toshniwal, Ivan Moshkov, Sean Narenthiran, Daria Gitman, Fei Jia, and Igor Gitman. 2024. Openmathinstruct-1: A 1.8 million math instruction tuning dataset. Preprint, arXiv:2402.10176.

[Wu et al.2024] Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. 2024. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. In Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 1819–1862, Mexico City, Mexico. Association for Computational Linguistics.

[Xi et al.2024] Zhiheng Xi, Wenxiang Chen, Boyang Hong, Senjie Jin, Rui Zheng, Wei He, Yiwen Ding, Shichun Liu, Xin Guo, Junzhe Wang, et al. 2024. Training large language models for reasoning through reverse curriculum reinforcement learning. arXiv preprint arXiv:2402.05808.

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

[Zhang et al.2023] Chiyuan Zhang, Daphne Ippolito, Katherine Lee, Matthew Jagielski, Florian Tramer, and Nicholas Carlini. 2023. Counterfactual memorization in neural language models.

[Zheng et al.2024] Tianyu Zheng, Ge Zhang, Tianhao Shen, Xueling Liu, Bill Yuchen Lin, Jie Fu, Wenhu Chen, and Xiang Yue. 2024. Opencodeinterpreter: Integrating code generation with execution and refinement. Preprint, arXiv:2402.14658.

A Related Works

A.1 Investigation on the Limitations of Transformer Capabilities

In recent years, large language models (LLMs) have achieved unprecedented success in various tasks requiring complex reasoning [OpenAI2023], such as coding [Guo et al.2024, Zheng et al.2024] and solving mathematical problems [Luo et al.2023, Toshniwal et al.2024]. Some researchers even view these exciting advancements as sparks of artificial general intelligence [Bubeck et al.2023]. In stark contrast, these models have shown unexpected failures in simple and intuitive tasks [Bian et al.2024, Koralus and Wang-Maścianica2023]. For example, the state-of-the-art GPT-4 only achieve 59% accuracy on three-digit multiplication problems [Dziri et al.2023].

What is the reason behind this stark discrepancy? Recent studies have examined LLMs' ability to composite knowledge from training scenarios to solve more complex problems. Tests covered tasks such as boolean variable assignment [Anil et al.2022], semantic parsing [Hosseini et al.2022], deductive reasoning [Sanyal et al.2022], and arithmetic reasoning [Kazemi et al.2023]. A common trend shows that as problem complexity increases, LLMs' accuracy drops significantly. Our MATHTRAP is based on math word problems, currently the most widely studied task for evaluating LLM reasoning. Unlike previous work that assessed models' compositional generalization by increasing problem complexity, we simply introduced straightforward logical traps into original problems without significantly increasing their complexity.

Researchers have also investigated whether LLMs' impressive reasoning abilities stem from learning general knowledge or merely reciting examples from their vast training corpora. Studies [Wu et al.2024, Zhang et al.2023] on LLMs' performance in factual versus counterfactual scenarios revealed significant performance drops in counterfactual cases, suggesting LLMs often recite answers from common cases in their training data. Recent research [Dziri et al.2023] modeled reasoning tasks as computational graphs, experimentally demonstrating that LLMs reason through subgraph matching rather than developing systematic problem-solving skills. Another study [Hu et al.2024] removed certain samples from the training set and found that LLMs rely on surrounding cases in the training set for mathematical reasoning rather than learning generalizable rules. These findings indicate that LLMs still face challenges in learning knowledge and combining them to solve out-of-distribution problems.

A.2 Math Word Problem Benchmark

Mathematical word problems have long been considered an effective proxy for evaluating the reasoning abilities of large language models (LLMs), garnering widespread attention from the academic community. Numerous benchmark datasets of math word problems have been proposed. ASDiv [Miao et al.2020] is a dataset covering most knowledge types encountered in elementary school, with each problem annotated with its corresponding knowledge component and grade level. SVAMP [Patel et al.2021] comprises 1,000 challenging elementary math word problems carefully designed and curated to assess a system's complex reasoning and multi-step arithmetic capabilities. cobbe2021training introduced GSM8K, a high-quality and diverse evaluation benchmark containing 8.5k math word problems. hendrycks2021measuring presented MATH, a dataset of 12,500 challenging competition math problems. Recently, researchers have also constructed unanswerable math word problems to evaluate hallucinations in LLMs' mathematical reasoning process. However, evaluating LLMs' compositional abilities has been limited to symbolic

reasoning tasks [Lake and Baroni2023, Dziri et al.2023]. tang2024paradox found that semantics play a crucial role in the reasoning process, motivating our MATHTRAP dataset to evaluate LLMs' compositional skills on semantically rich math word problems.

B Annotation Process and Standards of MATHTRAP Dataset

In this section, we provide a detailed explanation covering three aspects: the annotators' background, annotation criteria, and the annotation process.

B.1 Qualified annotators

Our annotation team consists of five students with STEM backgrounds from top universities, with an average math score of 145 out of 150 in their college entrance exams. They possess the necessary math knowledge and passed our qualification test, ensuring their ability to understand and complete our tasks with high quality.

B.2 Clear and Specific Annotation Criteria

- **Adherence to trap problem definition**: We provided clear definitions and examples for five types of traps (as shown in 3.1). Annotators were required to fully understand these definitions and verify that their trap problems align with the corresponding trap definitions.
- **Unambiguity**: We require annotators to mutually verify the semantic clarity of trap problems, ensuring the modified questions are unambiguous.
- **Difficulty**: We set a standard that the trap problems should cause GPT-3.5 to give incorrect responses in at least 1 out of 5 runs, preventing overly simple trap problems.
- **Diversity**: We dynamically monitored the distribution of knowledge points during the annotation process and provided feedback to annotators to adjust their selection of topics.

B.3 Standardized Annotation Process

- **Problem assignment and distribution**: Problems from each knowledge point in the original datasets were equally distributed among five annotators. This approach was adopted to prevent potential bias that might arise if any single annotator were to focus on only a subset of knowledge points.
- **Quality control during annotation**: We required strict adherence to annotation standards and recording of the verification process. A supervisor regularly checked the output for compliance with our standards and provided feedback.
- **Post-annotation quality assessment**: All annotators cross-verified trap problems created by others, filtering out those that didn't meet the annotation standards.

C Evaluation

C.1 Compared Method

We evaluate mainstream proprietary and open-source LLMs with strong mathematical capabilities. Proprietary models include Claude3-Opus, Gemini-pro, OpenAI o1, GPT-4-0125-preview, GPT-3.5-turbo-0125, and Kimi. The open-source models include the 7B, 13B, and 70B versions of MetaMath [Yu et al.2023], which are all based on the Llama-2 model. Additionally, we provide Llemma-MetaMath-7B, which is based on Llemma [Azerbayev et al.2024], a foundation model pretrained specifically for mathematics.

C.2 Prompt Template

Table 6 presents the prompt templates used to evaluate the accuracy of responses from various models using GPT-4. Table **??** displays the prompt templates for enhancing the training set answers with GPT-4. Table **??** shows the prompt templates used to indicate that a model's question might be unsolvable. Table **??** presents the prompt templates for in-context learning in a 1-shot setting. Table **??** presents the prompt templates for in-context learning in a 5-shot setting.

Table 4: The impact of external intervention methods on the accuracy for original problems and trap problems. "w/o Notice" refers to the control experiment without any external intervention. 'w/ Notice' indicates using a natural language prompt to inform the model that the problem description may contain traps. ICL(1/5-shot) refers to adding one or five demonstrations in the context to exemplify how to handle trap problems. The prompt templates employed are presented in Tables **??** in the Appendix.

Model	Judge	Original Problem		Trap Problem	
w/o Notice	w/ Notice	w/o Notice ICL(1-shot)	w/ Notice ICL(5-shot)	ICL(1-shot)	ICL(5-shot)
Gemini-pro	GPT-4	36.9	37.8	32.4	50.0
8.3	14.1	8.94	27.7		
Claude3-Opus	GPT-4	68.5	65.8	68.5	82.5
19.0	40.7	29.0	56.1		
Claude3-Opus	Claude-3.5	67.6	64.0	68.5	70.8
16.1	48.4	23.9	47.1		
GPT-3.5-turbo-0125	GPT-4	40.5	40.5	45.9	51.7
7.74	12.2	12.2	23.9		
GPT-3.5-turbo-0125	Claude-3.5	37.8	29.7	44.1	46.7
7.10	13.5	11.0	12.9		
GPT-4-0125-preview	GPT-4	70.3	72.1	65.8	77.5
35.5	50.3	41.9	48.4		
GPT-4-0125-preview	Claude-3.5	65.8	67.6	64.9	76.7
24.5	40.6	36.8	35.5		
Kimi	GPT-4	46.1	41.4	41.4	55.0
19.6	37.1	26.4	31.0		
Llemma-MetaMath-7B	GPT-4	41.4	42.7	32.2	33.3
6.36	9.86	7.93	7.74		
MetaMath-7B	GPT-4	32.5	32.5	30.0	35.0
1.94	3.23	5.16	17.4		
MetaMath-13B	GPT-4	37.5	37.5	34.2	40.7
3.87	3.87	2.58	12.3		
MetaMath-70B	GPT-4	34.2	30.8	35.8	38.4
6.45	7.74	4.52	16.8		
Llama3-8B	GPT-4	33.3	36.7	38.3	35.8
6.45	13.5	4.52	35.5		
Llama3-8B-Base	GPT-4	14.2	16.7	15.8	24.2
3.23	3.87	5.16	20.0		
Llama3-70B	GPT-4	61.7	61.7	54.2	61.7
7.74	26.5	14.8	39.4		
Llama3-70B-Base	GPT-4	37.5	30.0	34.2	41.7
7.74	9.03	7.74	25.8		

Table 5: The impact of fine-tuning data configurations on the accuracy for original and trap problems. We use Llemma as the foundation model. The parentheses indicate the judge model used.

Dataset	Original	Trap (GPT-4)	Trap (Claude-3.5)
GSM8K+MATH	20.8	1.01	0.65
GSM8K+MATH+MathTrap1K	13.3	12.4	34.2
MetaMath395K	41.4	6.36	1.94
MetaMath395K+MathTrap1K	33.3	29.1	11.6

Table 6: Prompt template used for evaluating the Trap Problem across various Large Language Models (LLMs) using GPT-4.

Following is a problem with no solution or can't be solved, and a reference answer about how to find the contradiction. There is also an answer you need to make a verdict on whether it finds the same contradiction as the reference answer does. Please output a short comment and end with [0] or [1] ([1] means the answer finds the same contradiction and explicitly states it). #The Problem: {input} #The Reference Answer: {ref} #The Answer Needs Verdict: {answer} #Your Verdict:

Table 7: Explanation of examples of trap problems for each category. The sections highlighted in yellow delineate the distinction between original problems and trap problems.

Type	Trap problem	Explanation
Concept Undefined	In right triangle XYZ with $\angle YXZ = 90°$, we have $XY = 24$ and $YZ = 25$. Find $\tan X$.	$\angle YXZ = 90°$, so $\tan X = \tan(\pi/2)$ is undefined.
Missing Condition	Natalia sold 48 clips in April and half as many clips in May. How many clips did Natalia sell altogether in April and June?	We don't know anything about June, so it's impossible to calculate the sum of the sales for April and June.
Direct Contradiction	An equilateral triangle has a perimeter of 30 centimeters and a height of 10 centimeters. Calculate the area of the triangle.	The height of the equilateral triangle and its side length are both 10 centimeters, which is contradictory and impossible.
Indirect Contradiction	Find the integer solution of the equation $x^2 + x = 3$.	The 2 solutions of this quadratic equation is $\frac{-1\pm\sqrt{13}}{2}$, so there is no integer solution.
Violating Common Sense	Max picks 5 different cards without replacement from a standard 52-card deck. What is the probability that the cards are of different suits?	There are only 4 suits in a deck, so it's impossible for 5 cards to be of different suits.