Not All LLM Reasoners Are Created Equal

Arian Hosseini
Mila
arian.hosseini9@gmail.com

Alessandro Sordoni
Mila, Microsoft Research

Daniel Toyama
Google DeepMind

Aaron Courville
Mila

Rishabh Agarwal
Mila, Google DeepMind

Abstract

We study the depth of problem-solving capabilities of LLMs, and to what extent they perform mathematical reasoning in a compositional manner. To this end, we create a new benchmark by composing pairs of existing math word problems together so that the answer to the second problem depends on correctly answering the first problem. We measure the difference between the performance of solving each question independently and solving the compositional pairs as the reasoning gap of a model. Our findings reveal a significant reasoning gap in most frontier LLMs. This gap is more pronounced in smaller and more cost-efficient models. The objective of this study is not to introduce yet another benchmark, but rather to provide a case study aimed at gaining deeper insights into current models' reasoning abilities, and to reassess existing established training methods and benchmarks.

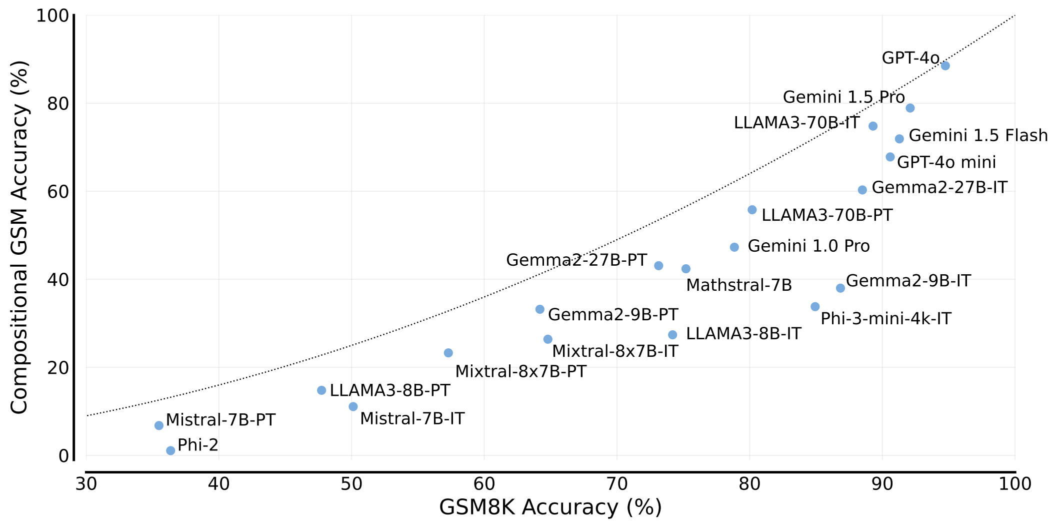

Figure 1: **Reasoning Gap:** Pairs of GSM8K test questions are chained together so that the answer of the first question (Q_1) is a variable in the second one (Q_2). The model is required to correctly answer both questions to solve the problem. If a model has an accuracy of S_1 on the Q_1 set, and S_2 on Q_2 set, then the expected Compositional GSM accuracy is $S_1 \times S_2$. The x-axis corresponds to the geometric mean $\sqrt{S_1 \times S_2}$, labeled GSM8K accuracy for simplicity. The trend-line $y = x^2$ is the expected Compositional GSM accuracy.

Compositional GSM Problem

Let X be the answer to the $\mathbf{Q_1}$:

$\mathbf{Q_1}$: There are 27 unicorns left in the world. One third of them are in the Scottish Highlands. Two thirds of the Scottish unicorns are female. How many female Scottish unicorns are there?

Solve it and use the value of X to solve $\mathbf{Q_2}$. Explain your answer step by step.

$\mathbf{Q_2}$: Zack's locker is half as big as Timothy's locker. Peter's locker is 1/4 as big as Zack's locker. If Peter's locker is X cubic inches, how big is Timothy's locker in cubic inches?

Figure 2: **Example Problem from the Compositional GSM benchmark**. The answer of Question-1 ($\mathbf{Q_1}$) is a variable X in Question-2 ($\mathbf{Q_2}$). Therefore, the model has to be able to solve the first question correctly in order to solve the second question. The new final answer of Question-2 is calculated by modifying its code-form solution and executing it. Question-1 and the number to modify in Question-2 are chosen to have a new final answer which is a positive integer not too far from the old answer of Question-2.

1 Introduction

The strong performance of large language models (LLMs) on high-school and college-level math reasoning benchmarks (Dubey et al., 2024; Google, 2024; OpenAI, 2023b), has led to the common belief that LLMs have "mastered" grade-school math, particularly as measured by the GSM8K benchmark (Cobbe et al., 2021). This apparent mastery of grade-school math problems raises a deeper question: do LLMs truly grasp the underlying concepts or do they mostly rely on dataset contamination or memorization (Srivastava et al., 2024)? For example, a recent examination on private "held-out" grade-school problems (Zhang et al., 2024) reveals that while frontier closed-source LLMs show minimal signs of overfitting, some open-weights models show systematic overfitting, possibly due to test data contamination.

In this work, we perform a case study to evaluate how well LLMs can combine learned concepts to solve unseen problems, to probe the brittleness of their reasoning abilities. To do so, we introduce *Compositional GSM*, a two-hop version of GSM8K with higher difficulty, where each problem chains two test questions together such that the answer to the first question is used as a variable in the second question (Figure 2). As LLMs can easily solve grade-school math problems, they should also be capable of solving combinations of those problems. As such, we measure the gap between their performance on solving the questions individually and on Compositional GSM. Specifically, we benchmark frontier open-weights and closed LLMs, including Gemini (Google, 2023, 2024), Gemma2 (Gemma Team et al., 2024), Llama-3 (AI@Meta, 2024), GPT (OpenAI, 2023a), Phi (Abdin et al., 2024), Qwen (Yang et al., 2024) and Mistral families (Jiang et al., 2024).

Here are our key findings:

- Most models exhibit a gap between their performance on GSM8K test set and Compositional GSM (Figure 1).
- This reasoning gap is larger in small and more cost-efficient models (Figure 5 and Figure 3).
- Instruction-following tuning of LLMs heavily favours the original GSM8K split (Figure 4).
- Finetuning with human data and synthetic data results in a similar reasoning gap trend (Figure 7).
- Smaller models benefit more from generating code rather than natural language Chain-of-Thought (CoT) to solve Compositional GSM problems (Figure 6).

2 Compositional Grade-School Math (GSM)

Each question in compositional GSM consists of two questions, Question-1 and Question-2, from a subset of 1200 examples of the original GSM8K test set. The final answer of Question-1 is refereed to as X which is a variable in Question-2 (Figure 2). The final answer of Question-2 is obtained by substituting X and solving it. The choice of Question-1 and the number to modify and replace with X in Question-2 was made in a a way such that the new final answer of Question-2 is different from its old final answer, and is a positive integer not too far from the old final answer.

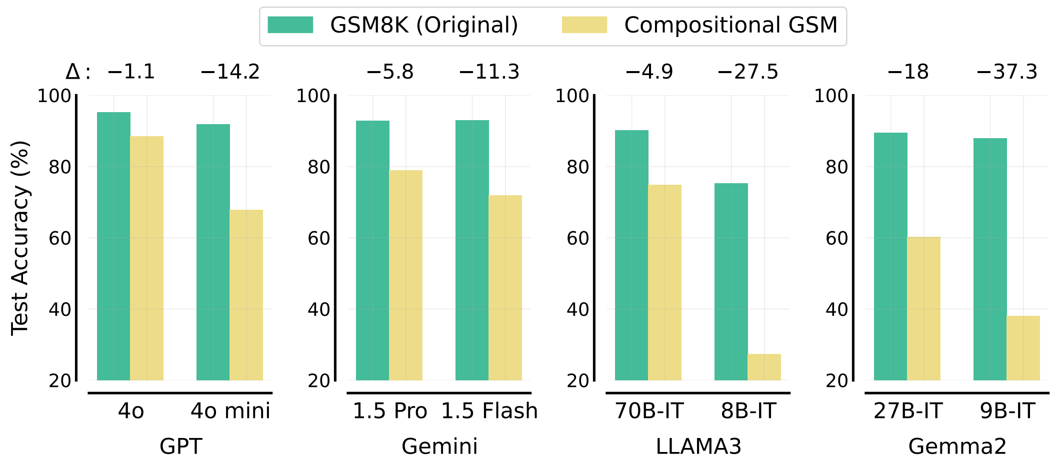

Figure 3: **Cost efficient LLMs reason differently:** showing four family of models, each having a high-cost and low-cost option. Although the cheaper models perform similarly on the original GSM8K test, they show a significant decline in performance on the Compositional GSM test.

Reasoning Gap Question-1 and Question-2 in our compositional queries are from the original test split $\mathcal{D}_{\text{original}}$, and the modified test split $\mathcal{D}_{\text{modified}}$ respectively. Assuming that a model has an accuracy of S_1 on $\mathcal{D}_{\text{original}}$ and S_2 on $\mathcal{D}_{\text{modified}}$, it is expected for it to have an accuracy of $S_1 \times S_2$ on the compositional split $\mathcal{D}_{\text{comp}}$. We report the following as the compositional reasoning gap score,

$$\text{Compositional reasoning gap}: \Delta = S_c - S_1 \times S_2 \tag{1}$$

where S_c is the performance of the model on $\mathcal{D}_{\text{comp}}$.

3 Experiments & Results

The distance to the trend-line in Figure 1 shows the reasoning gap of models. The x-axis corresponds to $\sqrt{S_1 \times S_2}$, which is the geometric mean of the accuracies on the set of Q_1 and Q_2 independently. We find that most models fall below expectation on Compositional GSM. Specifically, it is evident that cost-efficient models have a larger gap than more expensive models. More analysis is provided in the Appendix.

3.1 Cost-Efficient LLMs Reason Differently

The reasoning abilities of cost-efficient LMs has been rapidly improving over time, as evaluated using standard benchmarks (Bansal et al., 2024). For example, GPT-4o mini and Gemini 1.5 Flash both achieve above 90% accuracy on GSM, while costing $25 - 35\times$ cheaper than GPT-4o and Gemini 1.5 Pro respectively. This progress could be attributed to several factors, such as better pretraining data (AI@Meta, 2024), and knowledge distillation (Agarwal et al., 2024; Team et al., 2024). To this end, we investigate whether these reasoning gains on GSM8K still persist on Compositional GSM.

We study four family of models, each comprising both a high-cost and low-cost option, where cost is measured via parameter count or API pricing. Figure 3 shows the original GSM8K test split performance and Compositional GSM performance for all models. The numbers above the bars represents the reasoning gap defined in Eq 1. While cheaper models perform comparably on the original GSM8K test, they exhibit a notable drop in performance on the Compositional GSM test set. These results suggest that critical flaws of cost-efficient LLMs in their reasoning may be obscured by high scores on standard benchmarks. This underscores the need to rethink current strategies for developing cost-efficient language models.

3.2 Instruction-Tuning Impacts LLM Reasoning Differently

We compare pretrained and instruction-following tuned versions of models in three families of Mistral, LLAMA3 and Gemma2. Figure 4 illustrates this comparison, along with the performance gains from

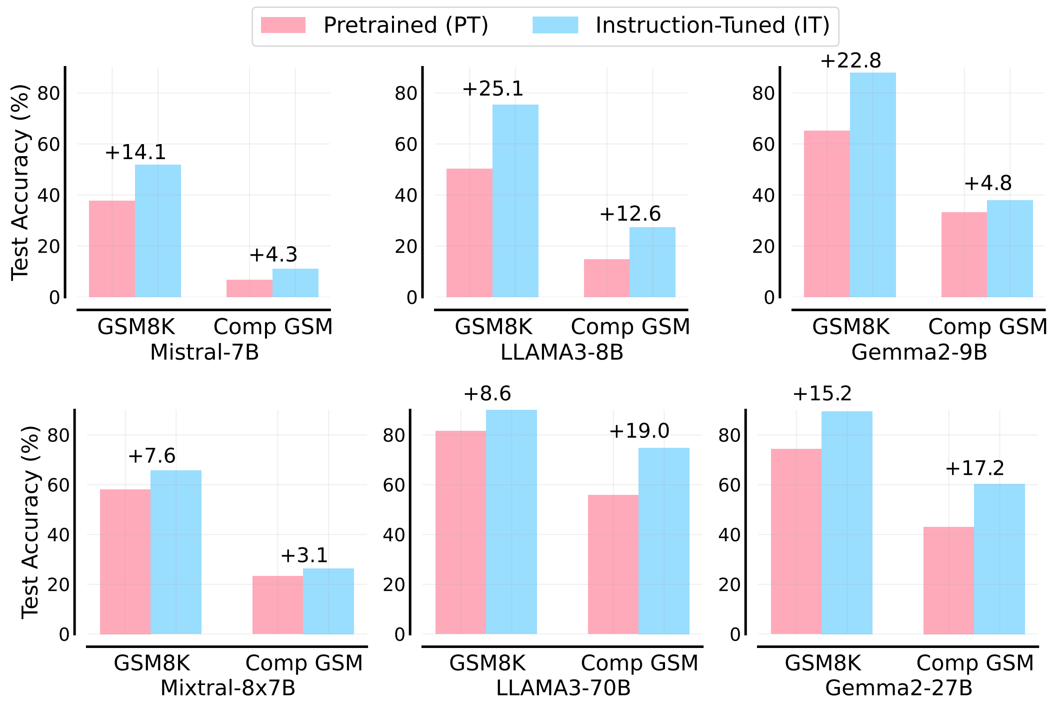

Figure 4: **Impact of Instruction-Tuning on Reasoning Gap:** comparing pretrained and instruction-following tuned variant of models from Mistral, LLAMA3 and Gemma2 families. Numbers above bars represent improvements from instruction-tuning on each set. For smaller models (top), we observe that instruction-tuning is highly optimized for GSM8K questions, which results in a greater improvement on the original GSM8K test set compared to the Compositional GSM test. However, this pattern does not hold for larger models (bottom).

instruction-tuning, displayed above bars for each test set. On small models (top row), this comparison shows that current instruction-tuning is heavily optimized for GSM8K questions. Instruction-tuning leads to a significantly larger improvement on the original GSM8K test set than the Compositional GSM test across model families. However, this trend does not apply to larger models (bottom row), where the improvements are inconsistent.

4 Discussion and Conclusion

We designed the Compositional GSM benchmark, which requires solving dependent pairs of math word problems. These problems are from the original GSM8K test split. We investigate the "System 2" mathematical reasoning capabilities of LLMs by comparing their performance on the original GSM8K test split and our Compositional GSM test set. Our analysis reveals a notable reasoning gap in most models. Many leading LLMs exhibit a substantial difference in performance when solving questions independently versus as part of the compositional pair. Our study indicates that smaller and more cost efficient models exhibit a larger reasoning gap. Models frequently struggle with pairs of questions and get distracted likely because they are tuned to handle one question at a time. They often answer the first question correctly, but lose attention to details and make subtle errors in answering the second question. We also noticed that learning from human data and self-generated data results in similar behaviour. In both settings, as training progresses, the model's performance on the original test split improves. However, beyond a certain point, performance on the Compositional GSM test begins to decline.

We emphasize that this benchmark should not be viewed as an endpoint or merely as a tool for generating additional training data, but as a catalyst to gain insights about current models and to re-evaluate and improve existing benchmarks. Our findings are intended to stimulate further exploration and provide new perspectives. Future research could build on this setup by incorporating more challenging questions, such as those from the MATH dataset, or by extending the framework to multi-modal problems to gain deeper insights into the reasoning capabilities of LLMs.

References

M. Abdin, S. A. Jacobs, A. A. Awan, J. Aneja, A. Awadallah, H. Awadalla, N. Bach, A. Bahree, A. Bakhtiari, J. Bao, H. Behl, A. Benhaim, M. Bilenko, J. Bjorck, S. Bubeck, Q. Cai, M. Cai, C. C. T. Mendes, W. Chen, V. Chaudhary, D. Chen, D. Chen, Y.-C. Chen, Y.-L. Chen, P. Chopra, X. Dai, A. D. Giorno, G. de Rosa, M. Dixon, R. Eldan, V. Fragoso, D. Iter, M. Gao, M. Gao, J. Gao, A. Garg, A. Goswami, S. Gunasekar, E. Haider, J. Hao, R. J. Hewett, J. Huynh, M. Javaheripi, X. Jin, P. Kauffmann, N. Karampatziakis, D. Kim, M. Khademi, L. Kurilenko, J. R. Lee, Y. T. Lee, Y. Li, Y. Li, C. Liang, L. Liden, C. Liu, M. Liu, W. Liu, E. Lin, Z. Lin, C. Luo, P. Madan, M. Mazzola, A. Mitra, H. Modi, A. Nguyen, B. Norick, B. Patra, D. Perez-Becker, T. Portet, R. Pryzant, H. Qin, M. Radmilac, C. Rosset, S. Roy, O. Ruwase, O. Saarikivi, A. Saied, A. Salim, M. Santacroce, S. Shah, N. Shang, H. Sharma, S. Shukla, X. Song, M. Tanaka, A. Tupini, X. Wang, L. Wang, C. Wang, Y. Wang, R. Ward, G. Wang, P. Witte, H. Wu, M. Wyatt, B. Xiao, C. Xu, J. Xu, W. Xu, S. Yadav, F. Yang, J. Yang, Z. Yang, Y. Yang, D. Yu, L. Yuan, C. Zhang, C. Zhang, J. Zhang, L. L. Zhang, Y. Zhang, Y. Zhang, Y. Zhang, and X. Zhou. Phi-3 technical report: A highly capable language model locally on your phone, 2024. URL `https://arxiv.org/abs/2404.14219`.

R. Agarwal, N. Vieillard, Y. Zhou, P. Stanczyk, S. R. Garea, M. Geist, and O. Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*, 2024.

AI@Meta. Llama 3 model card. 2024. URL `https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md`.

H. Bansal, A. Hosseini, R. Agarwal, V. Q. Tran, and M. Kazemi. Smaller, weaker, yet better: Training llm reasoners via compute-optimal sampling. *arXiv preprint arXiv:2408.16737*, 2024.

M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

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

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

L. Gao, A. Madaan, S. Zhou, U. Alon, P. Liu, Y. Yang, J. Callan, and G. Neubig. PAL: program-aided language models. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 10764–10799. PMLR, 2023. URL `https://proceedings.mlr.press/v202/gao23f.html`.

T. M. Gemma Team, C. Hardin, R. Dadashi, S. Bhupatiraju, L. Sifre, M. Rivière, M. S. Kale, J. Love, P. Tafti, L. Hussenot, and et al. Gemma. 2024. doi: 10.34740/KAGGLE/M/3301. URL `https://www.kaggle.com/m/3301`.

G. T. Google. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

G. T. Google. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv e-prints*, pages arXiv–2403, 2024.

Z. Gou, Z. Shao, Y. Gong, Y. Yang, M. Huang, N. Duan, W. Chen, et al. Tora: A tool-integrated reasoning agent for mathematical problem solving. *arXiv preprint arXiv:2309.17452*, 2023.

A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. S. Chaplot, D. de las Casas, E. B. Hanna, F. Bressand, G. Lengyel, G. Bour, G. Lample, L. R. Lavaud, L. Saulnier, M.-A. Lachaux, P. Stock, S. Subramanian, S. Yang, S. Antoniak, T. L. Scao, T. Gervet, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed. Mixtral of experts, 2024. URL `https://arxiv.org/abs/2401.04088`.

OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023a. doi: 10.48550/ARXIV.2303.08774. URL `https://doi.org/10.48550/arXiv.2303.08774`.

OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023b.

S. Srivastava, A. PV, S. Menon, A. Sukumar, A. Philipose, S. Prince, S. Thomas, et al. Functional benchmarks for robust evaluation of reasoning performance, and the reasoning gap. *arXiv preprint arXiv:2402.19450*, 2024.

G. Team M. Riviere, S. Pathak, P. G. Sessa, C. Hardin, S. Bhupatiraju, L. Hussenot, T. Mesnard, B. Shahriari, A. Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

A. Yang, B. Zhang, B. Hui, B. Gao, B. Yu, C. Li, D. Liu, J. Tu, J. Zhou, J. Lin, K. Lu, M. Xue, R. Lin, T. Liu, X. Ren, and Z. Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

H. Zhang, J. Da, D. Lee, V. Robinson, C. Wu, W. Song, T. Zhao, P. Raja, D. Slack, Q. Lyu, S. Hendryx, R. Kaplan, M. Lunati, and S. Yue. A careful examination of large language model performance on grade school arithmetic. *CoRR*, abs/2405.00332, 2024. doi: 10.48550/ARXIV.2405.00332. URL `https://doi.org/10.48550/arXiv.2405.00332`.

Appendices

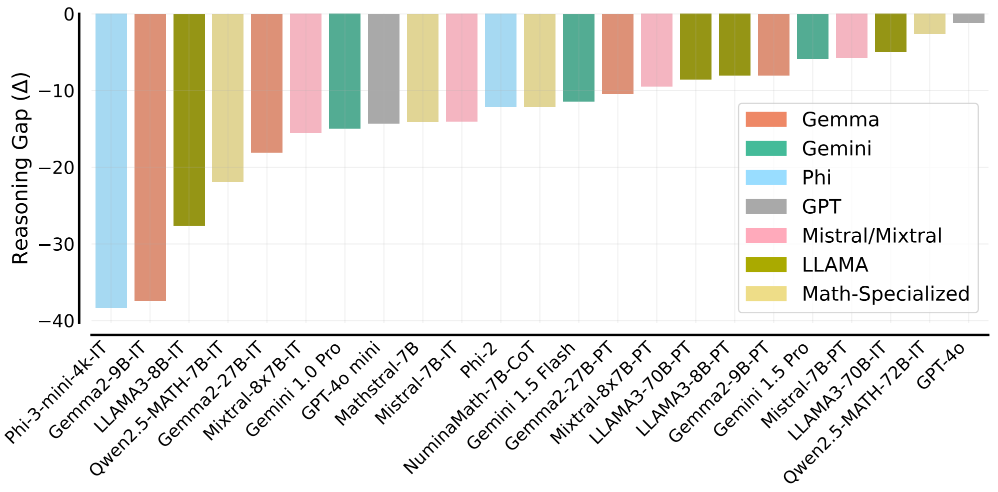

Figure 5: **Reasoning Gap** of notable open-weights and closed-source LLMs. Smaller, more cost-efficient and math specific models have a bigger gap.

A Compositional GSM Details

To obtain the new final answer of Question-2 automatically, we replace a number in the code-form solution of Question-2. We used a slightly modified version of the code-form solutions from Gao et al. (2023). The new final answer is the result of executing the code with the new number. We put significant efforts into ensuring that the modification to Question-2 is sensible. Figure 10 shows the distribution of final answers (magnitude) of the original test set of GSM8K and compositional GSM. Both test sets have a similar distribution of final answers.

Quality Checks To make sure that the modified questions are sensible and logical, we generated 16 candidate solutions per modified question from GPT-4o and Gemini 1.5 Pro. We filtered those questions for which less than 4 (out of 16) agree with the expected final answer from code execution. We checked these questions manually and modified them if needed so that they are logical (about 25% of questions).

B Experiment Details

Setup We evaluate each model on three test sets: **1) the original GSM8K test split**, **2) the modified GSM8K test split** which are the questions with X being substituted, and **3) the compositional GSM** test set. Each test set has 1200 examples. Following Zhang et al. (2024), we evaluate all models with an 8-shot prompt (Appendix I) for the original and modified GSM8K test splits. We also created a similar 8-shot prompt (Appendix J) for the compositional GSM questions. We evaluate GPT-4o, GPT-4o mini, LLAMA3-70B and 8B (PT and IT), Phi 2, Phi-3-mini-instruct, Gemini 1.0, 1.5 Flash and 1.5 Pro, Gemma2 9B and 27B (PT and IT), Mistral-7B (PT and IT), Mixtral-8x7B (PT and IT) and Mathstral-7B. All models are sampled with temperature 0, and pass@1 (Chen et al., 2021) is used to measure the performance on each test split. Some of the models require a preamble prefixed to the 8-shot prompt in order to output in a consistent format (Appendix F). We test both cases and report the best performance for each model.

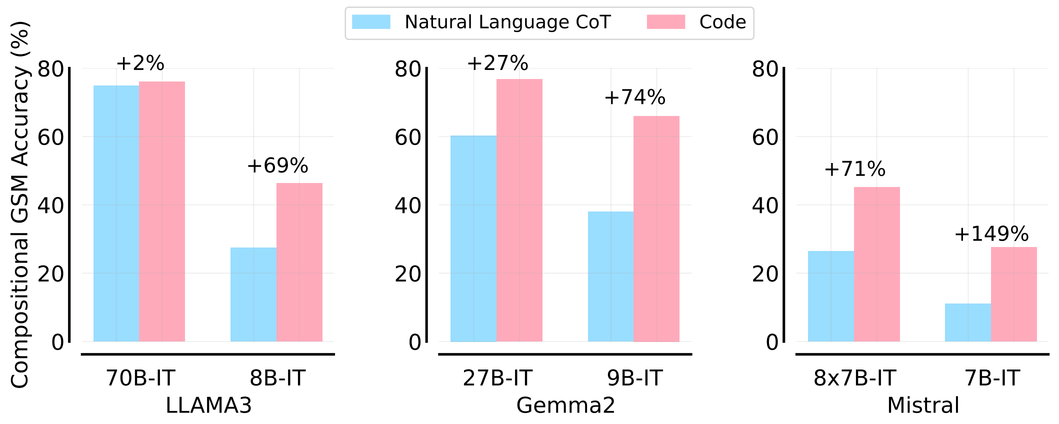

Figure 6: **Natural Language CoT v.s. Code:** Generating code to solve the problems helps in both settings of original test split and Compositional GSM split. Numbers above bars represent relative improvements over natural language Chain-of-Thought (CoT) generation. Smaller models benefit more from generating code rather than natural language CoT to solve Compositional GSM questions, further highlighting that smaller models demonstrate systematic differences in reasoning capabilities.

C Thinking in Natural Language v.s. Code

Breaking down the natural language problem into executable code steps has been shown to improve models' reasoning and generalization abilities (Gao et al., 2023; Gou et al., 2023). To this end, we evaluate whether the compositional problem-solving ability of LLMs improves when generating natural langauge CoT rationales compared to generating executable Python code. For code generation, we utilize a compositional 8-shot prompt(Appendix K), where the answers are written as two functions, one which solves the first question *solve_q1()*, and *solution()* which solves the second question with a *X* = *solve_q1()* line at the beginning.

Our results are shown in Figure 6 for three families of open-weight instruction-tuned models: LLAMA3-8B and 70B, Gemma2-9B and 27B, and Mistral 7B and (Mixtral) 8×7B. Notably, generating code generally improves performance on Compositional GSM problems, albeit not uniformly. Specifically, comparing relative improvements, smaller models benefit significantly more from generating code solutions compared to generating natural language CoTs. This ruther underscores the systematic differences in reasoning behaviors of smaller models.

D Finetuning Can Lead to Task Overfitting

Finetuning models on task specific problems is a common strategy to improve reasoning performance. In this section, we explore how it impacts the performance on Compositional GSM. We investigate the performance of Gemma2 27B PT as we finetune it on the original GSM8K training data, and self-generated rationales (aka synthetic data) to identify any difference in the characteristics of these two sources. We collect self-generated rationales which results in correct final answers for all GSM8K training queries.

See Appendix H for details of data generation and training for this set of experiments. We evaluated intermediate checkpoints (at 50, 100 and 400 training steps) from both settings on GSM8K original test split and Compositional GSM split (Figure 7). We observer a similar pattern for both settings. The Compositional GSM performance increases with some training (up to 100 steps), but drops with more training steps while GSM8K test performance keeps increasing, which suggests overfitting. Our results show that training on synthetic data generally leads to a higher Compositional GSM performance. We did not observe further improvements on either test splits after 400 training steps.

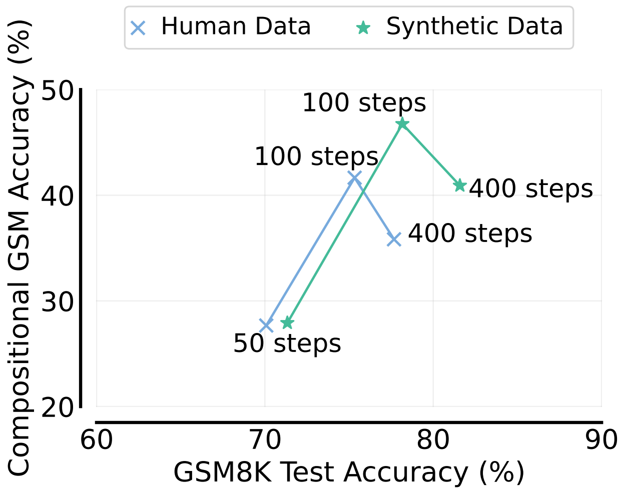

Figure 7: **Human Data v.s. Synthetic Data:** We finetune Gemma2 27B on the original GSM8K training data, and self-generated rationales. In both settings, after 100 training steps, Compositional GSM test performance drops while original GSM8K test performance keeps increasing. No further improvements were observed on either test split after 400 training steps.

E Failure Modes of LLMs on Compositional Questions

Does Solving Question-1 Guarantee Solving Question-2? Correctly solving Question-1 is a prerequisite to solve Question-2 in the compositional format. In Figure 8, we look at how often models are able to solve a question independently versus how often can they solve it given they have correctly solved the previous question in the compositional format. What remains for the model to do here is to substitute X and solve Q_2. The deviation from the diagonal line indicates that certain models may have become too specialized in handling GSM8K-style questions, and are unable to answer a second question having generated the solution to the first question. Our qualitative analysis shows that when given two questions, the model might answer the first one correctly, but often makes subtle errors and overlooks details, leading to inaccurate reasoning and solution for the second question.

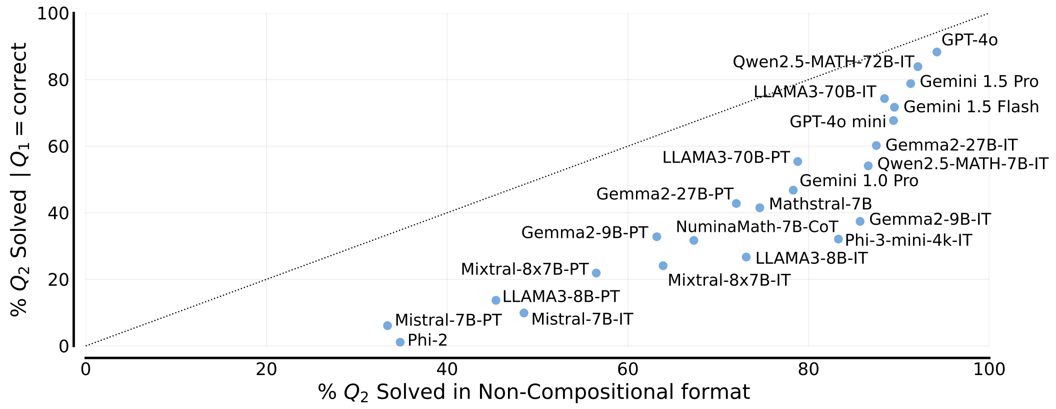

Figure 8: **Can models answer the second question if they have correctly answered the first one?** Here, we compare how often models are able to solve a question independently to how often they are able to solve them in the compositional format given that the first question is solved correctly. This is an alternate measurement of the compositional reasoning gap. If a model can solve a question independently, it should be able to solve it in a compositional setting given that the prerequisites are met. The gap from the diagonal line suggests that some models have overfit to the format of GSM8K type questions. While models may correctly answer the first question, they frequently makes subtle errors and miss key details when solving the second question.

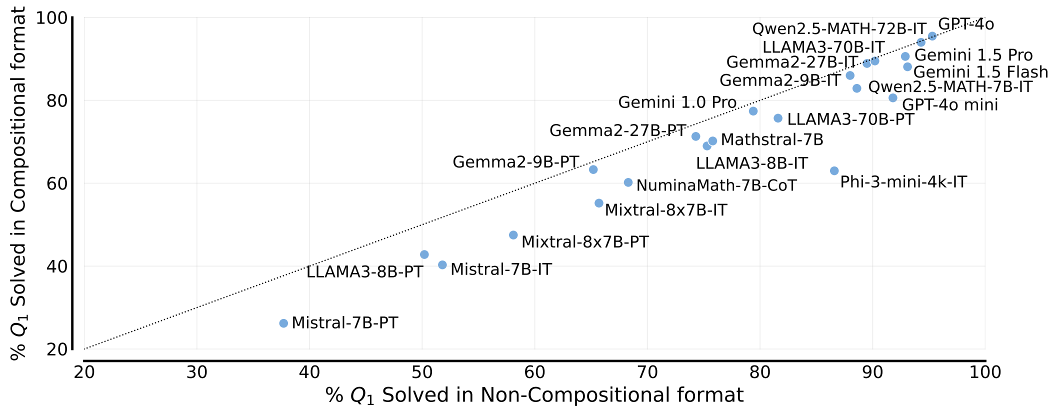

Figure 9: **Some LLMs get distracted easily:** Measuring models' ability to solve a question in the standard format (non-compositional) versus solving the same question as Q_1 in the compositional format. Models below the trend-line get distracted and cannot answer Q_1 in the compositional format even though solving it does not depend on solving any other question.

Models Get Distracted Easily Assuming an LLM answers a question correctly, it is somewhat expected that it would answer the same question correctly with additional context. Figure 9 shows how often a model answers a question (from Q_1 set) correctly on the x-axis, and how often it answers it correctly in our compositional format, as Q_1. Ideally, models should be on the $x = y$ line, but we observe that most of the models fall short of this expectation. Examining the responses from models with greater deviations from the trendline in Figure 9 reveals that they frequently make subtle errors. They often overlook important details, such as missing a reasoning step related to *each* in the question or omitting a multiplication step when the question specifies *in a month*. The models generally adhere well to the output format provided in the 8-shot context, resulting in negligible instances of non-extractable answers. This distraction is caused by the existence of a second question Q_2 in the prompt. Such failures lead to not being able to correctly answer Q_1, which subsequently impairs the models' ability to answer Q_2 correctly.

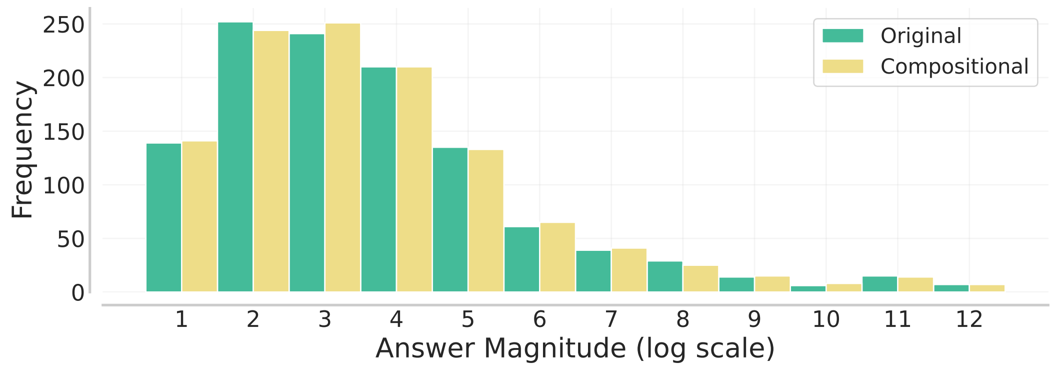

Figure 10: **Distribution of final answers** from the test set of original GSM8K and compositional GSM benchmark. The number modification in the compositional benchmark was done in a way to ensure that the new final answer is a positive integer not too far from the old answer. Our compositional GSM benchmark has a similar distribution of final answers.

F Prmopt Preambles

GSM8K Preamble
`I am going to give you a series of demonstrations of math Problems and Solutions. When you respond, respond only with the Solution of the final Problem, thinking step by step. At the end of the Solution, when you give your final answer, write it in the form ‘‘The final answer is ANSWER.’’`
Compositional GSM Preamble
`I am going to give you a series of demonstrations of compositional math questions and solutions. Respond by thinking step by step. Solve the first question and write the intermediate answer as ‘‘The Q1 answer is ANSWER1.‘‘ Then solve Q2. At the end of the solution, when you give your final answer, write it in the form ‘‘The final answer is ANSWER2.’’`

G GSM8K Original vs Modified

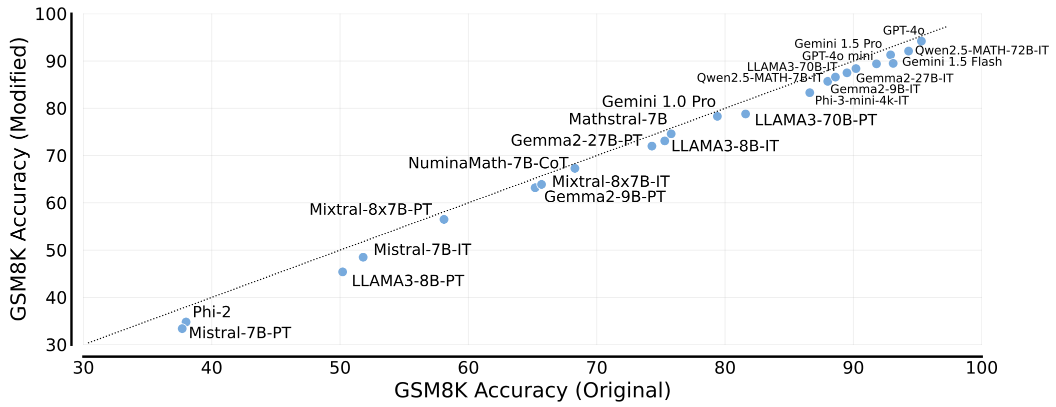

Figure 11: **Original v.s. Modified GSM8K test accuracy**. Most models are very close to the $x = y$ line, indicating that contamination is not a significant concern.

H Rejection Finetuning Details

Synthetic data was generated by prompting Gemma2 27B PT model with the 8-shot prompt in Appendix I to solve GSM8K training questions. We generated 10 solutions for each question in the original GSM8K training data, and only kept those solutions with a correct final answer. These model generation solutions were used to train the model.

I GSM8K 8-shot Prompt

```
Q: There are 15 trees in the grove.  Grove workers will plant trees in the grove
today.  After they are done, there will be 21 trees.  How many trees did the
grove workers plant today?
A: There are 15 trees originally.  Then there were 21 trees after some more were
planted.  So there must have been 21 - 15 = 6.  The final answer is 6.

Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars
are in the parking lot?
A: There are originally 3 cars.  2 more cars arrive.  3 + 2 = 5.  The final
answer is 5.

Q: Leah had 32 chocolates and her sister had 42.  If they ate 35, how many pieces
do they have left in total?
A: Originally, Leah had 32 chocolates.  Her sister had 42.  So in total they had
32 + 42 = 74.  After eating 35, they had 74 - 35 = 39.  The final answer is 39.

Q: Jason had 20 lollipops.  He gave Denny some lollipops.  Now Jason has 12
lollipops.  How many lollipops did Jason give to Denny?
A: Jason started with 20 lollipops.  Then he had 12 after giving some to Denny.
So he gave Denny 20 - 12 = 8.  The final answer is 8.

Q: Shawn has five toys.  For Christmas, he got two toys each from his mom and
dad.  How many toys does he have now?
A: Shawn started with 5 toys.  If he got 2 toys each from his mom and dad, then
that is 4 more toys.  5 + 4 = 9.  The final answer is 9.

Q: There were nine computers in the server room.  Five more computers were
installed each day, from monday to thursday.  How many computers are now in the
server room?
A: There were originally 9 computers.  For each of 4 days, 5 more computers were
added.  So 5 * 4 = 20 computers were added.  9 + 20 is 29.  The final answer is
29.

Q: Michael had 58 golf balls.  On tuesday, he lost 23 golf balls.  On wednesday,
he lost 2 more.  How many golf balls did he have at the end of wednesday?
A: Michael started with 58 golf balls.  After losing 23 on tuesday, he had 58 -
23 = 35.  After losing 2 more, he had 35 - 2 = 33 golf balls.  The final answer
is 33.

Q: Olivia has 23.Sheboughtfivebagelsfor3 each.  How much money does she have left?
A: Olivia had 23 dollars.  5 bagels for 3 dollars each will be 5 x 3 = 15 dollars.
So she has 23 - 15 dollars left.  23 - 15 is 8.  The final answer is 8.

Q: {question}
A:
```

J Compositional 8-shot Prompt

```
Let X be the answer to Q1:

Q1:  There are 15 trees in the grove.  Grove workers will plant trees in the
grove today.  After they are done, there will be 21 trees.  How many trees did
the grove workers plant today?

solve it and use the value of X to solve Q2.  Explain your answer step by step.

Q2:  There are X students in Marissa's class.  Each student started the year
with 10 pencils.  After two months, 1/5 of the total pencils in class were used.
At the end of the year, only 1/3 of the remaining pencils were left.  How many
pencils were left?

A: There are 15 trees originally.  Then there were 21 trees after some more were
planted.  So there must have been 21 - 15 = 6.  The Q1 answer is 6.  Therefore
X=6.  So there were 6 x 10 = 60 pencils in the class at the start of the year.
After two months, 60 x 1/5 = 12 pencils were used.  Thus, 60 - 12 = 48 pencils
were left unused after two months.  Therefore, 48 x 1/3 = 16 pencils were left at
the end of the year.  The final answer is 16.

Let X be the answer to Q1:

Q1:  If there are 3 cars in the parking lot and 2 more cars arrive, how many cars
are in the parking lot?

solve it and use the value of X to solve Q2.  Explain your answer step by step.

Q2:  Ingrid drinks X cups of water every day.  If there are 16 cups in a gallon,
how many gallons of water does she drink in 32 days?

A: There are originally 3 cars.  2 more cars arrive.  3 + 2 = 5.  The Q1 answer
is 5.  Therefore X=5.  So Ingrid drinks 5 cups of water a day so after 32 days
she drinks 5 * 32 = 160 cups of water.  There are 16 cups in 1 gallon so she
drinks 160 / 16 = 10 gallons of water in 30 days.  The final answer is 10.
             ⋮
Let X be the answer to Q1:

Q1:  {QUESTION_1}

solve it and use the value of X to solve Q2.  Explain your answer step by step.

Q2:  {QUESTION_2}

A:
```

Some examples in the prompt are omitted due to space constraints. The remaining question-and-answer pairs follow the same format.

K Compositional Code Prompt

Some examples in the prompt are omitted due to space constraints. The remaining question and code solutions are written with the same format. The answers are provided through two functions: *solve_q1()*, which addresses the first question, and *solution()*, which answers the second question. The *solution()* function begins with a line *X* = *solve_q1()* to get the result from the first function.

~~~
Write two functions ‘solve_q1‘ and ‘solution‘ to solve Q1 and Q2 problems.

Let X be the answer to Q1:

Q1:  There are 15 trees in the grove.  Grove workers will plant trees in the
grove today.  After they are done, there will be 21 trees.  How many trees did
the grove workers plant today?

Q2:  There are X students in Marissa’s class.  Each student started the year
with 10 pencils.  After two months, 1/5 of the total pencils in class were used.
At the end of the year, only 1/3 of the remaining pencils were left.  How many
pencils were left?

A: The answer is
```
def solve_q1():
            """There are 15 trees in the grove.  Grove workers will plant trees
in the grove today.  After they are done, there will be 21 trees.  How many trees
did the grove workers plant today?"""
            trees_initial = 15
            trees_after = 21
            trees_added = trees_after - trees_initial
            result = trees_added
            return result

def solution():
            """There are X students in Marissa’s class.  Each student started the
year with 10 pencils.  After two months, 1/5 of the total pencils in class were
used.  At the end of the year, only 1/3 of the remaining pencils were left.  How
many pencils were left?"""
            X = solve_q1()
            num_students = X
            pencils_per_student = 10
            total_pencils = num_students * pencils_per_student
            pencils_left_after_two_months = total_pencils * (4/5)
            remaining_pencils = pencils_left_after_two_months * (1/3)
            result = remaining_pencils
            return result
```
            ⋮

Let X be the answer to the following question:

Q1:  {QUESTION_1}

Q: {QUESTION_2}

A: The answer is
~~~