Using Extractable Symbolic Programs to Evaluate Mathematical Reasoning

Anonymous ACL submission

Abstract

Existing math datasets evaluate the reasoning abilities of large language models (LLMs) by either using the final answer or the intermediate reasoning steps derived from static examples. However, the former approach fails to surface model's uses of shortcuts and wrong reasoning while the later poses challenges in accommodating alternative solutions. In this work, we seek to use symbolic programs as a means for automated evaluation if a model can consistently produce correct final answers across various inputs to the program. We begin by extracting programs for popular math datasets (GSM8K and Math) using GPT4-o. For those executable programs verified using the original input-output pairs, they are found to encapsulate the proper reasoning required to solve the original text questions. We then prompt GPT4-o to generate new questions using alternative input-output pairs based the extracted program. We apply the resulting datasets to evaluate a collection of LLMs. In our experiments, we observe significant accuracy drops using our proposed evaluation compared with original static examples, suggesting the fragility of math reasoning in state-of-the-art LLMs.

1 Introduction

Mathematical reasoning is a fundamental skill essential for numerous complex applications, leading to a recent growing research effort on advancing large language models (LLMs) in this area. Thus, proper evaluation of LLMs' mathematical reasoning is crucial. Most previous studies have primarily evaluated LLMs using static datasets, such as GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021). Typically, evaluations focus solely on the final answers, overlooking reasoning flaws (Lewkowycz et al., 2022) and potential data contamination issues. Despite impressive results, LLMs can reply on shortcuts rather than true reasoning, displaying high sensitivity to input tokens (Li et al., 2024b,a). Alternatively, some works (Sawada et al., 2023; Golovneva et al., 2023) use model-based techniques to assess the reasoning quality, but these can suffer from model biases, limiting accommodation for alternative solutions.

In this paper, we present a focused study on evaluating mathematical reasoning which can be concisely encapsulated by symbolic programs, *i.e.,* Python programs. For such cases, we can automatically generate a diverse set of new test cases (input-output pairs) by varying the valid inputs fed into the program. Thus, if LLMs truly employ the appropriate reasoning process (as embodied by the programs) to solve the original question, they should also be able to consistently solve all new test cases. This approach allows us to evaluate the reasoning quality directly by examining the final answers, without ruling out alternatives.

To avoid costly manual annotations, we use the state-of-the-art LLM (GPT4-o) to generate Python programs for GSM8K and MATH. We retain only those questions with *extractable* programs, which can be automatically validated for our evaluation. This means the programs can be executed to produce the original gold answers. Upon manual inspection, 92% and 83% of the programs from GSM8K and MATH genuinely demonstrate the correct reasoning process required to solve the original questions. We then prompt GPT4-o to propose alternative valid inputs based on the extracted program and the original question. These inputs are then used to generate new input-output pairs derived from the program. Finally, GPT4-o is tasked to update the original question using these proposed inputs to create new test cases for evaluation.

Our experiments show that model performances drop drastically when evaluated on perturbed questions. For example, on original questions that GPT-4-Turbo answers correctly, only 48% of the time, the same model can answer all perturbed questions correctly, too. This suggests that GPT-4-Turbo, one

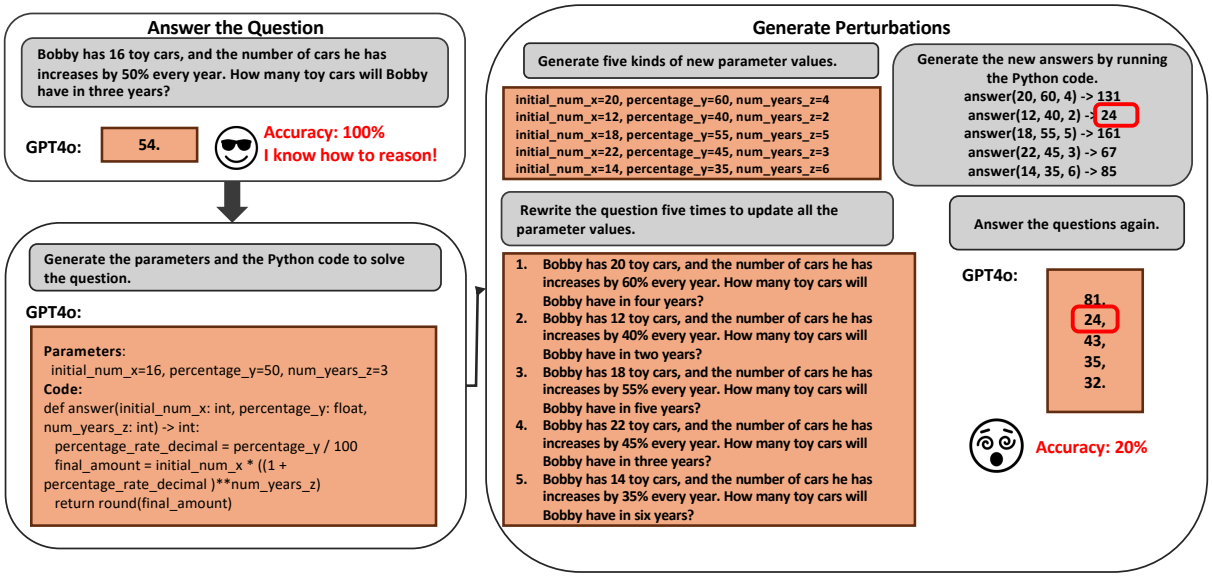

Figure 1: Pipeline of ReasonAgain, and a running example perturbed by ReasonAgain.

of the most capable large language models available, only truly understands the reasoning paths of less than half of the math questions. In other words, we show that existing evaluation methods on math questions are over-optimistic, and our proposed ReasonAgain is a better alternative.

2 Methods

Assessing the reasoning capabilities of large language models (LLMs) presents significant challenges, primarily because the reasoning process is not consistently articulated, and standardizing its representation is difficult. Moreover, multiple reasoning paths may exist to arrive at the same solution. Consequently, it is impractical to simply output the reasoning process and evaluate its correctness directly. Typically, the accuracy of reasoning is assessed through question-answering formats, such as verifying the accuracy of an answer to a mathematical problem. However, this paper contends that relying solely on a single question-answer pair is inadequate for genuinely assessing reasoning capabilities because: 1) an incorrect reasoning path may coincidentally yield the correct answer, and 2) potential data contamination could enable models to memorize answers without engaging in a legitimate reasoning process. To effectively evaluate the reasoning abilities of LLMs, we introduce ReasonAgain, which conceptualizes the reasoning process within Python code and automatically generates five additional perturbations of the same question. These perturbations retain the original reasoning process but feature different input values, thereby testing whether the model genuinely employs a correct reasoning process. The pipeline of ReasonAgain is illustrated in Figure 1.

Encapsulating the reasoning process. To explicitly represent the reasoning process of a math question, we first ask the pivot LLM to generate the parameters of questions.

Generate Parameters of the Question

Identify numerical values in the given question, then replace some of them with Python parameters that are either int or float, so that the resulting abstract question is still answerable with the same general solution as the original question. Follow the the provided examples.

{Few-shots examples}

{Question}

Then we use the generated parameter names to replace all the values in the question, and ask the LLM to generate a Python function that uses the generated parameters as the input to solve the question.

Generate Python Function of the Question

Write a Python program to solve the given abstract math question. Your program must contain a function called 'answer' that accepts the input parameters as specified in the question.

{Few-shots examples}

{Question with parameters.}

After generating Python code for all the ques-

tions, to ensure the quality of the code, we first filter out all the code that cannot be compiled. Then we run the code by inputting the original parameter values, and we only keep the code that can output the correct answer.

Generate the perturbations of the question. To generate the perturbations of the question, we first ask the model to generate 5 kinds of new parameter values given the original parameters using the following prompt.

Prompt for Generating Alternative Parameter Values

Here is a math question with the parameter and parameter values. Please perturb the value of parameters into different values. Output five kinds of new values in the same format as the given parameters in five lines without index.

Question: {Question}

Parameters: {Parameters}

After getting the new parameter values, we ask the model to update all the values in the question to generate the corresponding new questions.

Prompt for Generating New Questions

Here is a math question with old parameter values, and five kinds of new parameter values. Please rewrite the question five times to update all the parameters from old value to each corresponding new value in five lines without index.

Question: {Question}

Old Parameters: {Parameters}

New Parameters: {New Parameters}

New Questions:

To get the answers for each new questions, we run the Python code for each set of new parameter value, and use the output of the code as the gold answer. To examine the robustness of models' reasoning capability, we ask the models to answer the new questions, and compare the output with the gold answer.

3 Experiments

3.1 Experiment Settings

Datasets. We sample 2k questions from GSM8k (Cobbe et al., 2021) and 1k questions from MATH (Hendrycks et al., 2021). As discussed in Section 2, we first ask the model to generate the Python code for each question, and then we filter out all the problematic code that cannot be compiled or fail to return the correct gold answer. After filtering, in total, we have 1121 cases from GSM8k, and 268 cases from MATH. For each case, we use `ReasonAgain` to generate 5 perturbations as the new test cases, which gives us 5605 cases for GSM8k, and 1072 cases for MATH. We use GPT-4o (OpenAI et al., 2024) as the pivot LLM to generate all the parameters, code, and perturbations.

Baselines. We evaluate 4 LLMs in this paper: GPT-4-Turbo (OpenAI et al., 2024), GPT-4o (OpenAI et al., 2024), LLama-3.1-8B (Dubey et al., 2024), and Qwen-2.5-7B (Team, 2024) using the following different settings: direct, few-shot Chain-of-thought (CoT) (Wei et al., 2022), and few-shot Chain-of-thought + self-consistency (CoT+SC) (Wang et al., 2022).

Direct: We ask the model to directly answer the question without providing any examples using the following prompts.

Prompt for Generating Alternative Parameter Values

Answer the math question below. Only output the answer without units and any context words.

Question: {Question}

Answer:

Few-shot CoT: We follow the same CoT template and the same 8-shot math examples from Wei et al. (2022). Temperature is set to 0.

Few-shot CoT+SC: Following Wang et al. (2022), temperature is set to 0.7, and we run each query 5 times. The majority of the outputs will be used as the final answer.

Evaluation Metrics. We report Exact Match accuracy (EM) for all the experiments. Predicted answers are parsed by CoT format, and we round both gold answers and predicted answers before checking if the values are same.

3.2 Main Results

We show our main experiment results using our proposed `ReasonAgain` evaluation pipeline in Fig. 1. We observe a substantial performance drop across all models on both GSM8K and MATH. For direct inference, models drop 10%-15%, regardless of model sizes and capabilities. Such a drop is not mitigated by chain-of-thought and self-consistency inference methods, where we observe a similar 10% to 20% drop after our perturbation. In the normalized accuracy results, we show that models often demonstrate false hopes in their performances: they only answer 50% to 80% of the perturbed questions correctly on the questions they initially answered correctly. A more concerning finding is that models only truly understand at most half of

Model	Prompt	GSM8K					MATH				
		Accu.		Normalized Accu.			Accu.		Normalized Accu.		
		Before	After	Before	After	% of Correct	Before	After	Before	After	% of Correct
Llama3.1-8B	Direct	21.59	7.05	100	34.27	5.39	20.88	14.30	100	32.31	9.62
	CoT	88.26	69.62	100	75.31	48.63	71.49	44.02	100	54.04	33.71
	CoT+SC	85.75	68.01	100	71.95	39.92	69.48	43.13	100	53.06	25.43
Qwen2.5-7B	Direct	38.44	22.20	100	42.78	13.29	35.34	19.52	100	38.41	13.64
	CoT	60.04	49.48	100	63.79	30.44	38.96	25.30	100	46.19	17.53
	CoT+SC	61.23	50.12	100	45.97	29.66	40.56	26.43	100	44.95	16.83
GPT4o	Direct	66.57	52.93	100	72.89	48.86	57.83	37.27	100	56.25	36.11
	CoT	93.73	75.68	100	**79.52**	**58.80**	84.34	**50.76**	100	55.62	34.29
	CoT+SC	94.44	74.87	100	78.05	55.12	82.33	50.36	100	55.41	27.80
GPT4-Turbo	Direct	45.43	35.04	100	56.13	26.63	47.39	31.16	100	45.76	22.88
	CoT	54.75	43.49	100	70.02	48.12	36.14	28.19	100	**61.78**	**42.22**
	CoT+SC	51.52	40.95	100	71.23	50.09	55.02	37.59	100	55.62	32.85

Table 1: Performance of LLMs on GSM8K, MATH and corresponding perturbations generated by `ReasonAgain`. "Normalized Accu." refers to the performance on the subset of the test cases that the model answers correctly before perturbation. "Before" refers to the performance on the original dataset. "After" refers to the performance on the perturbations. "% of Correct" refers to the percentage of the cases that the model solves all the perturbations correctly.

the questions, and sometimes even less than 30%, as suggested by the "% of Correct" results. Combining these findings, we argue that `ReasonAgain` is an effective method for evaluating the true capabilities of models on mathematical reasoning, and current models' performances are overestimated by previous evaluation methods.

3.3 Human Evaluation

To evaluate if the generated code is reliable, we sample 200 cases from GSM8K and MATH (100 each), and ask three human experts to annotate the correctness of the generated perturbations of these 200 cases. The annotators are asked to understand the generated code, and check the correctness of the gold answers of perturbations. We observe 8 of the 100 cases from GSM8K and 17 of the 100 cases from MATH are problematic, mainly because some positive parameters being negative or the model cannot generate the correct code.

4 Related Work

Many works have discussed language model bias and inconsistency during reasoning (Li et al., 2024b,a; Zhou et al., 2024) and adversarial and contrastive evaluation (Gardner et al., 2020; Patel et al., 2021; Yu et al., 2024). To this end, our work provides a novel way for automatic mathematical consistency evaluation. At the same time, while some works have decomposed methods to solve or evaluate math questions (Hao et al., 2023; Madaan et al., 2023; Gao et al., 2023; Xia et al., 2024), our work points out that better language modeling is also needed to achieve consistent reasoning.

5 Conclusion

In this work, we propose `ReasonAgain`, a novel evaluation method to benchmark large language models' true capabilities on mathematical reasoning. `ReasonAgain` employs a programmatic perturbation method that changes the numbers in the original math questions and derives the corresponding gold answers. We then evaluate models on such perturbed questions. Experiments show that all models perform drastically worse on these perturbed questions, suggesting that 1) existing models do not truly understand the reasoning process of math questions, even though they sometimes predict the correct answer; 2) existing evaluation methods are inadequate, and current perceptions of model performances on math reasoning are over-optimistic. `ReasonAgain` can be the better alternative for evaluating model reasoning capabilities.

Limitations

Our work has several limitations.

Imperfect Programs. As pointed out in §3.3, some mistakes exist in the current generated programs, which leads to partially incorrect gold labels in some perturbed questions. We will explore better filtering mechanisms in later versions. However, such mistakes do not impact our overall conclusion, as model performances are much lower than the upper bounds.

Limited Program Coverage. Our program generation is limited by a conceptualization process proposed in Zhou et al. (2024), which does

not work well on certain types of math questions, such as geometry-related ones. As a result, `ReasonAgain` only works on a subset of all existing math questions.

Limited Reasoning Types. Our general formulation can be applied to other reasoning types, such as multiple-choice questions. However, we only focus on math questions in this work.

References

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. *arXiv preprint arXiv:2110.14168*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaoqing Ellen Tan, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aaron Grattafiori, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alex Vaughan, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Franco, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, Danny Wyatt, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Firat Ozgenel, Francesco Caggioni, Francisco Guzmán, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Govind Thattai, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Japhet Asher, Jean-Baptiste Gaya,

Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Karthik Prasad, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kun Huang, Kunal Chawla, Kushal Lakhotia, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Maria Tsimpoukelli, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikolay Pavlovich Laptev, Ning Dong, Ning Zhang, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Rohan Maheswari, Russ Howes, Ruty Rinott, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Kohler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vítor Albiero, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaofang Wang, Xiaojian Wu, Xiaolan Wang, Xide Xia, Xilun Wu, Xinbo Gao, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yuchen Hao, Yundi Qian, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, and Zhiwei Zhao. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. In *International Conference on Machine Learning*, pages 10764–10799. PMLR.

Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, et al. 2020. Evaluating models' local decision boundaries via contrast sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1307–1323.

Olga Golovneva, Moya Chen, Spencer Poff, Martin Corredor, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. 2023. Roscoe: A suite of metrics for scoring step-by-step reasoning. *Preprint*, arXiv:2212.07919.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8154–8173.

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. *arXiv preprint arXiv:2103.03874*.

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

Bangzheng Li, Ben Zhou, Xingyu Fu, Fei Wang, Dan Roth, and Muhao Chen. 2024a. Famicom: Further demystifying prompts for language models with task-agnostic performance estimation. *arXiv preprint arXiv:2406.11243*.

Bangzheng Li, Ben Zhou, Fei Wang, Xingyu Fu, Dan Roth, and Muhao Chen. 2024b. Deceptive semantic shortcuts on reasoning chains: How far can models go without hallucination? 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 7668–7681.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. In *Thirty-seventh Conference on Neural Information Processing Systems*.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin,

Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

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.

Tomohiro Sawada, Daniel Paleka, Alexander Havrilla, Pranav Tadepalli, Paula Vidas, Alexander Kranias, John J. Nay, Kshitij Gupta, and Aran Komatsuzaki. 2023. Arb: Advanced reasoning benchmark for large language models. *Preprint*, arXiv:2307.13692.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Shijie Xia, Xuefeng Li, Yixin Liu, Tongshuang Wu, and Pengfei Liu. 2024. Evaluating mathematical reasoning beyond accuracy. *arXiv preprint arXiv:2404.05692*.

Xiaodong Yu, Hao Cheng, Xiaodong Liu, Dan Roth, and Jianfeng Gao. 2024. ReEval: Automatic hallucination evaluation for retrieval-augmented large language models via transferable adversarial attacks. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1333–1351, Mexico City, Mexico. Association for Computational Linguistics.

Ben Zhou, Hongming Zhang, Sihao Chen, Dian Yu, Hongwei Wang, Baolin Peng, Dan Roth, and Dong

Yu. 2024. Conceptual and unbiased reasoning in language models. *arXiv preprint arXiv:2404.00205*.