BACKTRACKING MATHEMATICAL REASONING OF LANGUAGE MODELS TO THE PRETRAINING DATA

Anonymous authors
Paper under double-blind review

ABSTRACT

In this study, we identify subsets of model pretraining data that contribute to the math reasoning ability of language models and evaluate it on several mathematical tasks (e.g., addition, multiplication). We find that training on math-only data improves simple arithmetic but doesn't fully account for complex reasoning abilities, such as chain-of-thought reasoning. We also find that code data contributes to chain-of-thought reasoning while reducing arithmetic performance.

1 INTRODUCTION

A standout capability of foundational models is their proficiency in tackling mathematical problems (Kojima et al., 2022; Wei et al., 2022) only with in-context learning or chain-of-thought prompting. In this work, we aim to identify the source of such math reasoning capabilities *within the pretraining data* of language models (LM), which allows them to obtain such capabilities. We investigate which specific parts of the pretraining data contribute to LM's mathematical reasoning abilities. Identifying the pivotal parts of pretraining data that contribute to mathematical reasoning can help to design new LM's pretraining data mixtures more effectively. This is of particular importance as we enter an era where data availability and quality become critical bottlenecks in advancing model's capabilities.

Our underlying hypothesis for identifying pivotal pretraining data subsets is straightforward: if a subset elevates benchmark performance beyond what a random subset achieves, we infer that this subset contributes to the capabilities measured by that benchmark. Since all data subsets we test are within the pretraining data, continued pretraining serves as a rough approximation of up-weighting the data points within our chosen subsets. We propose some methods for identifying relevant subsets based on rules (e.g., math subsets) or similarity (e.g., cosine similarity-based) methods. We employ multiple mathematical reasoning evaluation tasks to distinguish between levels of mathematical reasoning capabilities. Our analysis shows that specific subsets, such as school math segments, enhance simple arithmetic skills in language models (e.g., resulting in more than 20% improvement on addition questions). However, they are less effective in enhancing more advanced mathematical skills, such as developing mathematical reasoning via sequential, chain-of-thought explanations. Our findings highlights that training on code-specific subsets improves chain-of-thought mathematical reasoning. Our proposed method sets the stage for future exploration into the impact of tailored pretraining data subsets on model's capabilities.

2 METHODOLOGY

Our objective is to identify and trace subsets of pretraining data responsible for enhancing language models with math reasoning capabilities. We do this by first identifying the data subsets, and then continuing model pretraining on these subsets. We then evaluate these models and compare their performance to the original model. Since this data was already included in the initial pretraining phase, extending the training of the language model on these specific subsets emphasizes their contribution without introducing additional information to the model. Consequently, if our evaluations show improved math abilities, we attribute the math proficiency of the language model to these specific subsets. We also use a baseline model, further trained on a random data subset, to maintain consistent training steps. Our assumption is that if training on a pretraining data subset results in improved performance in comparison to our random baseline, this subset contributes to the capability of the model tested by the benchmark. We illustrate our methodology in Figure 1 in the appendix.

Table 1: Performance on evaluation tasks after training on different subsets for 2,000 steps. * we trained on the similarity and GSM8K sets for 50 steps only since the GSM8K train set is small.

Training Subset	**add-2d**	**add-3d**	**mul-2d**	**asdiv-COT**
Baseline (no training)	62.0%	4.0%	1.0%	13.9%
Random Subset	62.5%	4.5%	2.0%	12.8%
DM-Math	**89.0%**	**44.5%**	**4.0%**	0.0 %
Code (w comment)	41.5%	5.0%	1.5%	**15.8%**
Code (wo comment)	43.0%	9.5%	3.0%	14.7%
Similarity Subset*	74.5%	21.0%	5.5%	13.8 %
GSM8k Train Set* (upper bound)	81.0%	24.0%	9.5%	28.0%

We choose a number of different subsets that could be intuitively considered as 'important'. This includes: (1) *DM-Math*, which is the sole segment dedicated entirely to mathematical content among the subsets of classified data in the Pile; (2) *Code* (with and without comments), which is a subsection from GitHub motivated by prior work (Wei et al., 2022) demonstrating that models trained on code significantly outperform non-code models in evaluations of mathematical performance that involve chain-of-thought prompting; (3) *Similarity Subset*, which we curate by computing the cosine similarity metric between GSM8K-training data (Cobbe et al., 2021) (as it is similar to our test datasets) and a subset of the Pile, and choosing the highest-ranked sections of the Pile.

3 Evaluation and Conclusion

We evaluate on multiple tasks, covering both simple arithmetic (a common test for measuring the numerical reasoning of language models Gao et al. (2021)) and elementary-level math word problems: (1) **add-2d**: Addition of two-digit numbers, (2) **add-3d**: Addition of three-digit numbers, (3) **mul-2d**: Multiplication of two-digit numbers, (4) **asdiv-COT**, elementary-level math word problems, which are known to showcase improved accuracy when models employ chain-of-thought reasoning (Wei et al., 2022). We use the pythia 2.8B model (Biderman et al., 2023) as the pretraining data, checkpoints, and code for this model are publicly available (Gao et al., 2020).

Our results are presented in Table 1. We first evaluate the original performance of the model on tasks without any further training (**no training**). This serves as our initial reference point. We then continue the training of our model on a **random subset** of the original training data. The performance after this phase will act as our second baseline. We find that: *Continued pretraining on a random subset does not dramatically change model performance.* A comparison between the model trained on a random subset to the original model, as seen in Table 1, reveals a negligible performance difference of less than 1% across all evaluation datasets. This observation validates our hypothesis that the continued pretraining itself does not significantly alter model behavior.

Results from Table 1 show that training on DM-Math, focused on math question-answer pairs, boosts simple arithmetic but drops asdiv-COT performance to 0.0, likely due to memorization or overfitting. Upon manually inspecting model generations, we observe that the model's natural language generation abilities, crucial for chain-of-thought reasoning, deteriorate. Moreover, *code data improves chain-of-thought reasoning, but not simple arithmetic reasoning*. Models trained on code, both with and without comments, perform largely similarly. Both subsets (especially with comments) appear to moderately improve chain-of-thought performance by increasing the asdiv-COT performance by about 3% compared to the random trained model (with 12.8% accuracy) while maintaining or degrading simple arithmetic reasoning performance.

In conclusion, our study reveals the utility of math and code subsets for mathematical and chain-of-thought reasoning, respectively, showing promise for future work applying this to larger models and data subsets. Our findings, especially the trends we observe with code-based model training, highlight the need for thorough research on how different data mixtures, including code, affect model reasoning.

URM Statement

The authors acknowledge that at least one key author of this work meets the URM criteria of ICLR 2024 Tiny Papers Track.

References

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. Pythia: A suite for analyzing large language models across training and scaling, 2023.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf`.

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. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Albert Danial. cloc: v1.92, December 2021. URL `https://doi.org/10.5281/zenodo.5760077`.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Amir Feder, Abhilasha Ravichander, Marius Mosbach, Yonatan Belinkov, Hinrich Schütze, and Yoav Goldberg. Measuring causal effects of data statistics on language model's 'factual' predictions, 2023.

Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 2881–2891. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/1e14bfe2714193e7af5abc64ecbd6b46-Paper.pdf`.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, September 2021. URL `https://doi.org/10.5281/zenodo.5371628`.

Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, Evan Hubinger, Kamilė Lukošiūtė, Karina Nguyen, Nicholas Joseph, Sam McCandlish, Jared Kaplan, and Samuel R. Bowman. Studying large language model generalization with influence functions, 2023.

Han Guo, Nazneen Rajani, Peter Hase, Mohit Bansal, and Caiming Xiong. FastIF: Scalable influence functions for efficient model interpretation and debugging. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 10333–10350, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.808. URL `https://aclanthology.org/2021.emnlp-main.808`.

Zayd Hammoudeh and Daniel Lowd. Training data influence analysis and estimation: A survey, 2023.

Xiaochuang Han, Byron C. Wallace, and Yulia Tsvetkov. Explaining black box predictions and unveiling data artifacts through influence functions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 5553–5563, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.492. URL https://aclanthology.org/2020.acl-main.492.

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

Ruoxi Jia, Fan Wu, Xuehui Sun, Jiacen Xu, David Dao, Bhavya Kailkhura, Ce Zhang, Bo Li, and Dawn Song. Scalability vs. utility: Do we have to sacrifice one for the other in data importance quantification? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8239–8247, June 2021.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.

Maxwell Nye, Anders Andreassen, Guy Gur-Ari, Henryk Witold Michalewski, Jacob Austin, David Bieber, David Martin Dohan, Aitor Lewkowycz, Maarten Paul Bosma, David Luan, Charles Sutton, and Augustus Odena. Show your work: Scratchpads for intermediate computation with language models, 2021. https://arxiv.org/abs/2112.00114.

Yasaman Razeghi, Robert L Logan IV, Matt Gardner, and Sameer Singh. Impact of pretraining term frequencies on few-shot numerical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 840–854, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.59. URL https://aclanthology.org/2022.findings-emnlp.59.

Subhro Roy and Dan Roth. Mapping to declarative knowledge for word problem solving. *Transactions of the Association for Computational Linguistics*, 6:159–172, 2018. doi: 10.1162/tacl_a_00012. URL https://aclanthology.org/Q18-1012.

Y. SOMDA. Guesslang. https://github.com/yoeo/guesslang, 2021.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

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

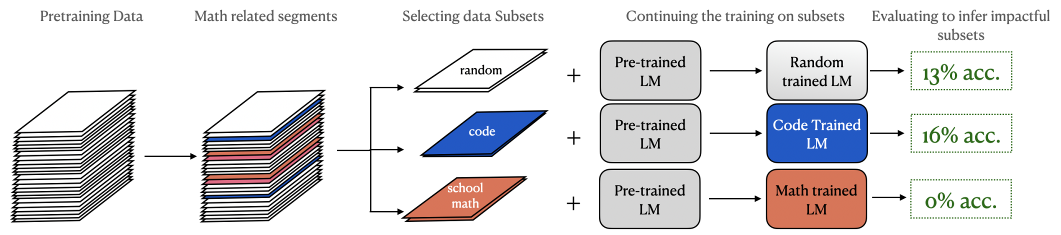
A Appendix

A.1 Limitations and Future Work

We note that there are several limitations to our approach. First, our similarity search was confined to small sections of the pretraining data because of lack of resources. We anticipate discovering more substantial and influential parts if we extend our search across a broader dataset. We hope that future work will utilize our similarity search method on a larger scale, applying it not only to math reasoning but also to other evaluations of large language models.

Additionally, our method involves comparing evaluations of models further trained on subsets of relevant data to those trained on a random subset. Consequently, we cannot identify contributing subsets of the pretraining data that do not yield improved performance. This situation may arise,

Figure 1: Overview of our approach to identify pretraining subsets contributing to model performance. Initially, we select subsets of the pretraining data for continued model training. Then, the evaluation results – in this case, evaluating on asdiv dataset with chain-of-thought prompting – are compared between the model further trained on these subsets and a model trained on a random subset.



for example, if the subset contributes only when combined with other data, or only has a minor impact on our evaluation settings. Furthermore, we did not examine multiple models in this work, and our results may not be applicable to larger models or models trained on significantly different datasets. For future work, it would be beneficial to investigate the effects of combining different data subsets, particularly in understanding how synergistic combinations influence model performance. Additionally, exploring a wider range of models, including larger and more diverse ones, could offer deeper insights into the generalizability of our findings.

A.2 Training Details

We train most models (except the GSM8k and similarity-based ones) for 2,000 steps, continuing the pretraining process using the same hyperparameters as Biderman et al. (2023), but keeping the learning rate constant (as opposed to continuing the learning rate decay) at 1.6 $\times 10^{-5}$.

We load the model checkpoint with optimizer states[1] included, and train all models using the same GPT-NeoX codebase[2] originally used to pretrain. We use a batch size of 131,072 tokens (i.e., 64 sequences of length 2,048), smaller than the original pretraining batch size of 2,097,152, due to computational and dataset size constraints. All other hyperparameters are matched to those in Table 6 of Biderman et al. (2023). This includes training in fp16 with mixed precision, weight decay of 0.1, no dropout, and clipping gradient norms over 1.0.

We train the GSM8k and similarity-based models for 50 steps due to the small size of GSM8k - the train split comprises of 1,132,169 tokens, and so training for 50 steps corresponds to training for roughly 5.8 epochs. In initial experiments, we also found that performance tended to plateau at this point. We also limit the amount of training steps performed when training over the similarity search subset to match the GSM8k setting.

All models were trained on 4 80GB A100s, with a per-GPU batch size of 16.

A.3 Comment Stripping

We remove comments using cloc (Danial, 2021) and guesslang (SOMDA, 2021). We pass each document through guesslang, which uses an MLP trained on existing source code to determine which programming language the document contains. We then use cloc's `--strip-comments` command to strip comments. Notably, this removes block comments but keeps inline comments.

A.4 Evaluation details

In the case of simple arithmetic tasks (add-2, add-3d, mul-2d), we evaluate 200 randomly generated instances. The arithmetic task evaluations are for assessing the model's simple math capabilities. We also employ the asdiv dataset, utilizing chain-of-thought prompting, to evaluate the models' step-by-

[1]Available at `https://huggingface.co/EleutherAI/neox-ckpts-pythia-2.8b`
[2]`https://github.com/EleutherAI/gpt-neox`

"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 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 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 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 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 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 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 answer is 33.
Q: Olivia has $23. She bought five bagels for $3 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 answer is 8."

Figure 2: 8-shot chain of thought prompt used for evaluation on asdiv.

step mathematical reasoning through natural language explanations. For asdiv, we use the existing test split of 2,305 examples. In both cases, we evaluate using 8-shot in-context learning, following the evaluation setting of prior work (Wei et al., 2022), in order to be able to extract answers from non-finetuned language models reliably. We use the prompt seen in Figure 2 when evaluating on asdiv, following Wei et al. (2022). Importantly, we check for exact matches between our evaluation sets and training data segments to mitigate the potential impact of model memorization on our results. We remove any evaluation examples we found from our training data. For example, we find 247 evaluation examples in the DM-Math train set.

A.5 Data Subset Statistics

We show the number of documents in each data subset we examine in Table 2. Note that we do not train for a single epoch for the majority of these subsets.

A.6 Examples of Subsets

The example of the dm math subset of the Pile and the similarity search subsets are presented in Figures 3 and 4.

Table 2: Number of documents in each of the data subsets we examine.

Subset Name	**# Documents**
Random	1999980
DM-Math	9223022
Code (w comments)	28512522
Code (wo. comments)	13938984
Similarity Search	3000

“...Let g be m(16). Suppose 5*x - 6299997 = 3*v, g*x + 0*v - 3779998 = 2*v. Round x to the nearest 100000. 1300000 Let w = -1.37 - -1.21584. Round w to three decimal places. -0.154 Let p(w) = -268*w + 3. Suppose 2*g + 12 = -g. Let m be p(g). Suppose -1895 - m = -3*q. What is q rounded to the nearest one hundred? 1000 Let p = -19981 - -13485. Let l = 6426.47 + p. Let q = l - -69. Round q to 1 decimal place. -0.5 Let m = 53 + -26. Let n = 27.0512 - m. What is n rounded to three decimal places? 0.051 Let l = -24 + 23.1. Let r = l + -18.1. Let g = r + 18.48. Round g to one decimal place. -0.5 Let w = -0.084 - -0.22. Let c = -3.38 + 3.1. Let z = w + c. What is z rounded to 2 decimal places? -0.14 Suppose 3*o = 7*o - 2*p, 2*p = -5*o + 18. Suppose -o*s + 44205 = 12205. Suppose -s = -m - 5*j, -m + 5*j + 64000 = 3*m. What is m rounded to the nearest 1000? 16000 Let l = -19505445 + 19505446.5667497. Let d = -1.96675038 + l. Let z = d + 0.4. What is z rounded to 7 decimal places? -0.0000007 Let u = 455.015 - 455. Round u to 3 decimal places. 0.015 Let q = -9.93149 + -43.00517. Let o = q + 9.93882. Let r = 43 + o. What is r rounded to 4 dps? 0.0022 Let i = 11.05 + -11.0502842. Round i to 5 decimal places. -0.00028 Suppose 3*m = 8*m. Suppose -p +q(19)? 5 Let x(z) = -z - 1. Let w be x(5)....”

Figure 3: Example of text in The Pile for the DM-Math subsection

“...the first group bought 8 slices of pizza and 6 soft drinks for $41.14. The second group bought 5 slices of pizza and 6 soft drinks for $30.10. How much does one slice of pizza cost? Let x = the price of a slice of pizza Let y = the price of a soft drink 8 slices of pizza and 6 soft drinks cost $41.14 $8x + 6y = 41.14$. 5 slices of pizza and 6 soft drinks for $30.10. $5x + 6y = 30.10$. A slice of pizza costs $3.68 and a soft drink costs $1.95. * * * 1. A collection of dimes and nickels is made up of 16 coins and is worth $1.25. How many nickels are in the collection?...”

Figure 4: Example of text in The Pile found by similarity search method

A.7 Related Work

Tracing model abilities to data Numerous previous studies have examined tracing model abilities to data (Hammoudeh & Lowd, 2023), with work on language models typically focused on the influence of finetuning data (Han et al., 2020; Guo et al., 2021), rather than data seen during pretraining. These techniques can broadly be classified as either retraining-based or gradient-based (Hammoudeh & Lowd, 2023), the former examining model capabilities after (re)training on data subsets, and the latter examining gradients to calculate the influence of training data points. Due to the extreme computational costs of the latter (Grosse et al., 2023), we take a retraining-based approach, continuing the pretraining of a model on carefully chosen data subsets. However, unlike prior retraining-based work (Jia et al., 2021; Feldman & Zhang, 2020), we do not entirely retrain multiple models from scratch, but rather just continue model pretraining, which is considerably computationally cheaper. However, it comes at the cost of being unable to estimate pointwise influence (since we require multiple datapoints to continue pretraining without the model achieving zero loss trivially). As such, our approach allows us to identify and validate the effect of pretraining data subsets (e.g., data from the same domain), similar to prior studies studying the impact of various higher-level pretraining data statistics (e.g. presence of numbers) on model behavior (Elazar et al., 2023; Razeghi et al., 2022).

Mathematical & reasoning capabilities of LMs Solving mathematical problems, especially when expressed in natural language, has been studied as a way to evaluate language models abilities in problem-solving and reasoning abilities (Hendrycks et al., 2021; Roy & Roth, 2018). Recent work has found that large pretrained language models struggle with these problems without the addition of various inference-time techniques, such as providing in-context examples (Brown et al., 2020), prompting a model to explain its reasoning (Wei et al., 2022; Nye et al., 2021), or exploring multiple potential generations (Wang et al., 2023). Curiously, such abilities often require well-trained or large 'enough' models to make use of them (Wei et al., 2022). Given this, we study mathematical problems to gain insights on the more general reasoning capabilities of language models, and due to the need for models equipped with certain capabilities (such as the ability to perform chain-of-thought) to achieve strong performance.