

---

# Basic Arithmetic Properties in the Space of Language Model Prompts

---

**Mateusz Krubiński**

Charles University, Faculty of Mathematics and Physics  
Institute of Formal and Applied Linguistics  
krubinski@ufal.mff.cuni.cz

## Abstract

Large pre-trained neural Language Models (LLMs) that can effectively utilize enormous amounts of unlabeled textual data have recently changed the whole field of Natural Language Processing. By utilizing prompting techniques enabled by the in-context learning capabilities, LLMs have been shown to perform on par with dedicated models trained for downstream tasks. One such a task is numerical reasoning and, in particular, the ability to conduct basic arithmetic operations. The question we wish to answer is whether the basic properties of arithmetic operations, such as the commutative property, hold in the space of LLM prompts – does asking the LLM to compute  $13 + 37$  vs  $37 + 13$  result, on average, in the same outcome? In contrast to most previous works, which reported Accuracy only, we take a closer look (MAE, Pearson’s R) at the error distribution to better understand the performance with regard to prompt perturbations and scaling laws.

## 1 Introduction and Related Work

Large Language Models (LLMs) [Brown et al., 2020, Zhang et al., 2022, Black et al., 2022, Touvron et al., 2023a, OpenAI, 2023] are able to solve not only classical NLP tasks, such as summarization (when conditioned on a long document and a “TL;DR:” token) but also to perform advanced reasoning [Qiao et al., 2023, Huang and Chang, 2023], including mathematical reasoning [Lu et al., 2023, Imani et al., 2023]. Previous works tackled several known issues of the arithmetic reasoning: performance drop when the number of symbols increases [Qian et al., 2023], handling of the out-of-range numbers [Kim et al., 2021] or the unclear impact of tokenization [Lee et al., 2023]. While recent developments, such as the Toolformer [Schick et al., 2023], enable precise arithmetic computations via the direct usage of a calculator API, there is still a lot of open questions regarding the *native* mathematical capabilities of LLMs. Lu et al. [2022] have shown that the order in which the samples are provided to the model in the prompt can make a difference between state-of-the-art and random guess performance for NLP tasks such as sentiment classification or textual entailment. Inspired by those observations, we wish to establish whether a similar phenomenon affects the numerical reasoning of LLMs – or, phrasing it differently, whether the commutative property holds in the space of LLM prompts. We also examine the associative and distributive properties not only reporting the frequency of correct predictions but also trying to measure the errors quantitatively.

## 2 Experimental Setup

In our experiments, we explore the OPT model family [Zhang et al., 2022] of Transformer-based language models trained with the next-token prediction task (causal language modeling) on large corpora, including the Pile [Gao et al., 2020] and the Pushshift.io [Baumgartner et al., 2020] datasets. We test the following variants (differing in the number of trainable parameters): 2.7B, 6.7B, 13B,

30B, and 66B. Furthermore, we report results of the 175B parameter GPT<sup>1</sup> davinci-002 model, accessed via the OpenAI API. We utilize greedy decoding (without beam search) and generate in a deterministic manner, i.e., without output sampling. Razeghi et al. [2022] have previously shown that the number of occurrences of a particular number in the data used to train the LLM affects the performance. For example, numbers that correspond to important historical events (e.g., 1939) or ones used in common phrases (e.g., 24 hours) are over-represented. Therefore, in our experiments, whenever we compare different prompt variants, we start by sampling a random set of numbers that we utilize in creating an artificial test-set (2000 examples). We explore four arithmetic tasks: 2D++ (addition of three 2-digit integers), 3D+ (addition of two 3-digit integers), 2D× (multiplication of two 2-digit integers), and 1DC (composite operation on three digits, i.e.,  $a \times b + c$ ). For the  $n$ -digit operation, we sample uniformly from the  $[0, 10^n]$  range. The size of the prompt (number of examples) equals three and we use the purely numerical prompt (white spaces inserted for readability), i.e.:

```
2+9+3=14 \n 7+1+12=20 \n 6+1+67=74 \n 59+12+76=
```

Zhou et al. [2022] have previously shown that within the range in which we operate, such a simple prompt formulation achieves results comparable to advanced techniques such as *scratchpad* [Nye et al., 2021] or Chain-of-thought [Wei et al., 2023] approaches while making it straightforward to formulate concepts such as “associative” or “commutative”. We will refer to the first part of the prompt (e.g.,  $2 + 9 + 3 = 14$ ;  $7 + 1 + 12 = 20$ ) as *template* and the second one (e.g.,  $59 + 12 + 76 =$ ) as *query*. We measure the model performance by reporting not only Accuracy (percentage of correct predictions) but also the Mean Absolute Error (MAE) and Pearson’s correlation coefficient (R) between correct results and model predictions.

In our experiments, we always trim the output at the first generated newline symbol. On average, the output is parsable to an integer value for over 99% of the inputs after a single post-processing step that removes commas and points (models sometimes output a thousands separator, e.g., 2, 537). The remaining ones (e.g., “12 24”) when the correct answer is “36” or “ $9 \times 8 = 72$ ” with “17” being the correct answer) are excluded from the evaluation, as they would bias both MAE and Pearson’s R. We acknowledge that this may overestimate the models’ Accuracy.

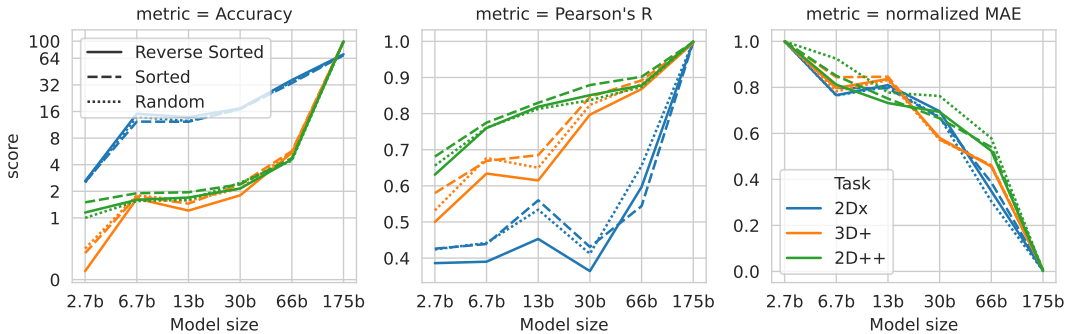


Figure 1: Commutative property – the cumulative results for the three prompt variants (see Section 2.1) reported on the 2D++, 3D+, and 2D× tasks. For the description of the metrics, see Section 2. For readability, we normalize the MAE values (*normalized MAE*) by the error of the smallest (2.7B) model (see Table 5 in the Appendix.)

## 2.1 Commutative property

In order to explore the commutative property, we compare three variants of prompts for the 2D++, 3D+, and 2D× tasks. The aspect that varies is the ordering of the *left-side* numbers in each example (we average over ascending and descending ordering):

- Rand – random ordering of numbers both in the template and in the query ( $1 + 17 + 5 = 23$ ;  $3 + 7 + 2 = ?$ )
- Sort – all numbers sorted ( $1 + 5 + 17 = 23$ ;  $2 + 3 + 7 = ?$ )

<sup>1</sup><https://learn.microsoft.com/en-us/semantic-kernel/prompt-engineering/llm-models>

- RevSort – all numbers sorted, but reversed in the query ( $1 + 5 + 17 = 23$ ;  $7 + 3 + 2 = ?$ )

Considering that LLMs do not differentiate between numbers and ordinary strings, we consider lexicographic ( $1 < 11 < 3$ ) vs numerical ( $1 < 3 < 11$ ) ordering.

Table 1: Cumulative (averaged over model sizes and tasks) results for all three properties (see Section 2), highlighting the differences between prompt variants. The best-scoring prompt variant within a particular property is highlighted independently for each metric.

Property	Variant	Metric		
		Accuracy $\uparrow$	Pearson’s R $\uparrow$	MAE $\downarrow$
Commutative	Rand	20.62	0.72	398.59
	Sort	20.52	<b>0.73</b>	<b>395.45</b>
	RevSort	<b>20.88</b>	0.70	426.71
Associative	NOP	<b>18.27</b>	<b>0.82</b>	<b>22.50</b>
	PS1	14.05	0.78	25.74
	PS2	16.87	0.79	29.69
Distributive	PAR	21.40	<b>0.69</b>	22.21
	DIS	<b>23.45</b>	<b>0.69</b>	<b>19.40</b>
	SWP	22.83	0.63	23.35

## 2.2 Associative and Distributive properties

To check whether the associative property holds, we explore the 2D++ task. For consistency, we do not insert parentheses into the template and consider three query variants:

- NOP – no parentheses ( $3 + 4 + 31 = ?$ )
- PS1 – parentheses rearranged to enclose the first and the second numbers ( $(3 + 4) + 31 = ?$ )
- PS2 – parentheses rearranged to enclose the second and the third numbers ( $3 + (4 + 31) = ?$ )

We validate the distributive property by looking at the modified 1DC task. The prompt variants we consider are:

- PAR – both template and query with the parentheses formulation ( $2 \times (8 + 4) = 24$ ;  $4 \times (6 + 4) = ?$ )
- DIS – both template and query with the distributed formulation ( $(2 \times 8) + (2 \times 4) = 24$ ;  $(4 \times 6) + (4 \times 1) = ?$ )
- SWP – different template/query format, averaged over PAR;DIS and DIS;PAR ( $2 \times (8 + 4) = 24$ ;  $(4 \times 6) + (4 \times 1) = ?$  and  $(2 \times 8) + (2 \times 4) = 24$ ;  $4 \times (6 + 4) = ?$ )

## 3 Results

**Commutative property** We start by analyzing the type of ordering used to sort the numbers in the prompt. Averaging over model size and type of prompt, the difference in Accuracy between lexicographic and numerical ordering is below 0.5%. Therefore, we consider it insignificant, and from this point onward, we will report results for the numerical ordering. Figure 1 presents the results with respect to the model size, and the aggregated statistics are reported in Table 1. Surprisingly, the RevSort variant of the prompt achieves the highest average Accuracy, but the differences are subtle. However, it results in the highest average MAE and the lowest Pearson’s R.

The largest (175B) parameter model that we consider solves the 3D+ and 2D++ addition tasks for every prompt variant with an almost perfect Accuracy of 99. For the 2D $\times$  task, despite the Accuracy of roughly 70, the MAE evaluates to a single digit, and Pearson’s R evaluates to 0.99, indicating a very strong linear correlation. Prompted with the challenging, purely numerical prompt, the second-largest 66B parameter returns the correct result only for less than 10% of inputs. Smaller models achieve even lower Accuracy. Those results are surprising if we compare them with the MAE and Pearson’s R. For example, the 66B model with the Rand prompt variant scores 4.45/18.6/0.87 Accuracy/MAE/Pearson’s R on the 2D++ task. This suggests that while the model does not predict the correct result *exactly*, the output is, on average, relatively close (see Table 6 in the Appendix).

Inspired by those observations, we take a closer look at the *number of tokens* that a particular model outputs for a given input. In Table 2, we report the average Accuracy for each model with respect to the number of tokens and their corresponding frequency in the output. Outputs with 3 or more tokens constituted less than 0.1% of all outputs. Therefore, for clarity, we decided to exclude them from the analysis. One can observe a clear tendency – the number of correct predictions is much higher if the model outputs only a single token. The smaller the model, the more significant the difference. Furthermore, for models from the OPT family (2.7B–66B ones) that use the same vocabulary, we can notice that the frequency of single-token outputs does not correlate with the model’s size. It is, however, significantly smaller than the frequency of single-token outputs for the largest 175B GPT model that uses a different vocabulary – this may be a further indication that not only the number of trainable parameters but also the tokenization method affects the performance.

Table 2: Accuracy for the commutative property experiments (averaged over tasks and prompt variants) with respect to the number of tokens and their corresponding frequency in the generated output.

#Tokens	Model Accuracy (Frequency %)					
	2.7B	6.7B	13B	30B	66B	175B
1	3.3 (48.3%)	17.4 (46.8%)	13.2 (49.4%)	21.8 (46.9%)	36.2 (48.9%)	99.0 (61.3%)
2	0.3 (51.7%)	1.5 (53.2%)	1.7 (50.6%)	2.5 (53.0%)	8.6 (51.1%)	77.2 (38.7%)

**Associative property** Based on all three metrics in Table 1, we notice that the NOP query variant (without parentheses) performs best. The differences, especially in terms of Accuracy, are more significant than for the commutative property. This is the case even for the largest 175B parameter model that achieved 99.2/75.8/95.8 Accuracy for the NOP/PS1/PS2 query variants. In order to understand the differences, we measured the average number of tokens<sup>2</sup> required to encode the query. The results are as follows: 6/7/6 for the NOP/PS1/PS2 query variants. Those results suggest that the model is mostly able to incorporate the parentheses (Accuracy of 99.2 vs 95.8), but only if they do not influence the tokenization. We leave the question of whether the inclusion of parentheses in the template would make a difference for future work.

**Distributive property** Based on our findings (Table 1), one can observe that swapping the template/query format (SWP) hurts the performance – it leads to the largest MAE and lowest Pearson’s R. The Accuracy is higher than for the PAR variant. However, this can be explained by the significantly lower (93.8%) than the average (99.6%) frequency of parsable outcomes from the 175B model – for both PAR and DIS prompt variants, the output was parsable in 100% of the cases. Surprisingly, the Pearson’s R for the PAR prompt variant (Figure 3 in the Appendix) was actually higher for the 66B model (0.750) than for the 175B model (0.745), despite both generating 100% of parsable outcomes.

## 4 Additional Experiments

**Next generation LLMs** Since the release of the OPT model family, the next generation of LLMs such as Pythia [Biderman et al., 2023], LLaMA [Touvron et al., 2023a], LLAMA 2 [Touvron et al., 2023b] or Mistral [Jiang et al., 2023] have been publicly released. They incorporate various improvements developed by the community – concerning the architecture (e.g., Rotary Embeddings [Su et al., 2021] or Grouped-Query Attention [Ainslie et al., 2023]), the training process (e.g., 1T training tokens for the 7B LLaMA vs 180B for OPT models) and the tokenization (e.g., LLaMA tokenizer splitting all numbers into individual digits), outperforming larger models on standard benchmarks. To establish whether the improvements also carry to numerical reasoning, we repeat our experiments (see Table 3) with the 7B and 13B variants of LLaMA and LLAMA 2. Overall, the results are inferior to the 175B GPT model but much better than even the largest 66B OPT model. For the commutative experiments, the differences are small, but the same tendency holds, i.e., sorting improves the Accuracy, and reverse-sorting hurts it. The ability to handle parentheses (PS1 and PS2) is an emergent one, with clear differences between the 7B and 13B variants. The relatively low numbers on the distributive experiments show that LLMs still struggle with composite operations, even when handling single digits.

<sup>2</sup><https://github.com/openai/tiktoken>

Table 3: Accuracy for the 7B and 13B variants of LLaMA and LLAMA 2, reported for each prompt variant and all three arithmetic properties considered. For the commutative property, we average over the 3D+, 2D++ and 2D× tasks.

		Commutative			Associative			Distributive		
		Rand	Sort	RevSort	NOP	PS1	PS2	PAR	DIS	SWP
LLaMA	7B	57.5	59.9	56.4	53.6	3.8	6.2	25.0	24.6	25.7
	13B	71.0	72.9	68.8	73.9	35.9	42.4	32.4	31.8	31.7
LLAMA 2	7B	69.2	70.9	69.1	70.6	34.6	52.6	30.7	26.8	31.3
	13B	78.3	77.4	77.2	82.6	57.8	71.4	34.8	39.1	32.0

**Fine-tuning for arithmetic reasoning** So far, we have been exploring the general-purpose language models. To establish how the task-specific models handle the prompt perturbations, we have fine-tuned the 2.7B and 6.7B OPT models to perform arithmetic reasoning. We focused on the commutative experiments, fine-tuning the models on the artificially created training data that consisted of uniformly mixed equations (corresponding to the 3D+, 2D++, and 2D× tasks) and Rand prompt formulation. We used the LoRA [Hu et al., 2022] procedure with the default parameters from the PEFT [Mangrulkar et al., 2022] library and trained for 5,000 steps with a linearly decreasing learning rate of 1e-4, an effective batch of 256 and using the AdamW [Loshchilov and Hutter, 2019] optimizer.

Table 4: Accuracy of the 2.7B and 6.7B OPT models fine-tuned for arithmetic reasoning.

	2D++			2D×			3D+		
	Rand	Sort	RevSort	Rand	Sort	RevSort	Rand	Sort	RevSort
OPT 2.7B FT	98.3	97.6	98.2	77.2	77.7	77.2	53.6	55.0	53.8
OPT 6.7B FT	100.0	100.0	100.0	93.0	92.4	92.6	95.8	95.6	95.4

In Table 4, we report Accuracy on the same test-sets used in previous experiments. Both models are able to solve the 2D++ task (almost) perfectly but the 6.7B variant performs significantly better on the 3D+ and 2D× tasks. The differences between the Rand/Sort/RevSort prompt variants are negligible, with the Sort variant (to a certain degree) positively affecting the performance of the smaller (2.7B) model.

## 5 Conclusion

In this work, we explored the problem of establishing whether the basic arithmetic properties hold in the space of Language Model prompts by querying a number of LLMs with several prompt variants and measuring the statistical differences in the output. Our initial results are indefinite – while the number ordering does not influence the largest GPT model, the inclusion of the parentheses may influence the tokenization and, thus, performance. The OPT models perform significantly worse, which makes it difficult to pinpoint the influence of every factor. The performance of the recent LLaMA models is closer to the GPT model. It shows that the field is quickly developing, but LLMs still struggle with composite tasks.

We acknowledge the limitations of our work: we limit the analysis to particular families of LLMs (OPT and LLaMA), we arbitrarily fix the size of the prompt and its formulation, we operate in a limited range (numbers not greater than  $10^3$ ), and we do not explore all of the possible template/query combinations.

## Acknowledgements

This work was supported by Charles University (projects GAUK 291923 and SVV 260698) and by CELSA (project no. 19/018). It used resources provided by the LINDAT/CLARIAH-CZ Research Infrastructure (<https://lindat.cz>), supported by the Ministry of Education, Youth and Sports of the Czech Republic (project no. LM2023062). The author would like to thank Pavel Pecina and Katarzyna Woźnica for their time spent discussing this work.

## References

- J. Ainslie, J. Lee-Thorp, M. de Jong, Y. Zemlyanskiy, F. Lebrón, and S. Sanghai. GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints. *arXiv preprint arXiv:2305.13245*, 2023.
- J. Baumgartner, S. Zannettou, B. Keegan, M. Squire, and J. Blackburn. The Pushshift Reddit Dataset. *Proceedings of the International AAAI Conference on Web and Social Media*, 14(1):830–839, May 2020. doi: 10.1609/icwsm.v14i1.7347.
- S. Biderman, H. Schoelkopf, Q. G. Anthony, H. Bradley, K. O’Brien, E. Hallahan, M. A. Khan, S. Purohit, U. S. Prashanth, E. Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR, 2023.
- S. Black, S. Biderman, E. Hallahan, Q. Anthony, L. Gao, L. Golding, H. He, C. Leahy, K. McDonnell, J. Phang, M. Pieler, U. S. Prashanth, S. Purohit, L. Reynolds, J. Tow, B. Wang, and S. Weinbach. GPT-NeoX-20B: An Open-Source Autoregressive Language Model. In *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 95–136, virtual+Dublin, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.bigscience-1.9.
- T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language Models are Few-Shot Learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.
- L. Gao, S. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite, N. Nabeshima, S. Presser, and C. Leahy. The Pile: An 800GB Dataset of Diverse Text for Language Modeling. *arXiv preprint arXiv:2101.00027*, 2020.
- E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*, 2022.
- J. Huang and K. C.-C. Chang. Towards Reasoning in Large Language Models: A Survey. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1049–1065, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.67.
- S. Imani, L. Du, and H. Shrivastava. MathPrompter: Mathematical Reasoning using Large Language Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 37–42, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-industry.4.
- A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed. Mistral 7B. *arXiv preprint arXiv:2310.06825*, 2023.
- J. Kim, G. Hong, K.-m. Kim, J. Kang, and S.-H. Myaeng. Have You Seen That Number? Investigating Extrapolation in Question Answering Models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7031–7037, Online and Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.563.
- N. Lee, K. Sreenivasan, J. D. Lee, K. Lee, and D. Papailiopoulos. Teaching Arithmetic to Small Transformers. *arXiv preprint arXiv:2307.03381*, 2023.
- I. Loshchilov and F. Hutter. Decoupled Weight Decay Regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

- P. Lu, L. Qiu, W. Yu, S. Welleck, and K.-W. Chang. A Survey of Deep Learning for Mathematical Reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14605–14631, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.817.
- Y. Lu, M. Bartolo, A. Moore, S. Riedel, and P. Stenetorp. Fantastically Ordered Prompts and Where to Find Them: Overcoming Few-Shot Prompt Order Sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.556.
- S. Mangrulkar, S. Gugger, L. Debut, Y. Belkada, S. Paul, and B. Bossan. PEFT: State-of-the-art Parameter-Efficient Fine-Tuning methods. <https://github.com/huggingface/peft>, 2022.
- M. Nye, A. Andreassen, G. Gur-Ari, H. W. Michalewski, J. Austin, D. Bieber, D. M. Dohan, A. Lewkowycz, M. P. Bosma, D. Luan, C. Sutton, and A. Odena. Show Your Work: Scratchpads for Intermediate Computation with Language Models. *arXiv preprint arXiv:2112.00114*, 2021.
- OpenAI. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*, 2023.
- J. Qian, H. Wang, Z. Li, S. Li, and X. Yan. Limitations of Language Models in Arithmetic and Symbolic Induction. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9285–9298, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.516.
- S. Qiao, Y. Ou, N. Zhang, X. Chen, Y. Yao, S. Deng, C. Tan, F. Huang, and H. Chen. Reasoning with Language Model Prompting: A Survey. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5368–5393, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.294.
- Y. Razeghi, R. L. Logan IV, M. Gardner, and S. Singh. Impact of Pretraining Term Frequencies on Few-Shot Numerical Reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 840–854, Abu Dhabi, United Arab Emirates, Dec. 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.59.
- T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, L. Zettlemoyer, N. Cancedda, and T. Scialom. Toolformer: Language Models Can Teach Themselves to Use Tools. *arXiv preprint arXiv:2302.04761*, 2023.
- J. Su, Y. Lu, S. Pan, B. Wen, and Y. Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding. *arXiv preprint arXiv:2104.09864*, 2021.
- H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample. LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971*, 2023a.
- H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv preprint arXiv:2307.09288*, 2023b.
- J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv preprint arXiv:2201.11903*, 2023.

- S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin, T. Mihaylov, M. Ott, S. Shleifer, K. Shuster, D. Simig, P. S. Koura, A. Sridhar, T. Wang, and L. Zettlemoyer. OPT: Open Pre-trained Transformer Language Models. *arXiv preprint arXiv:2205.01068*, 2022.
- H. Zhou, A. Nova, H. Larochelle, A. Courville, B. Neyshabur, and H. Sedghi. Teaching Algorithmic Reasoning via In-context Learning. *arXiv preprint arXiv:2211.09066*, 2022.



## A Appendix

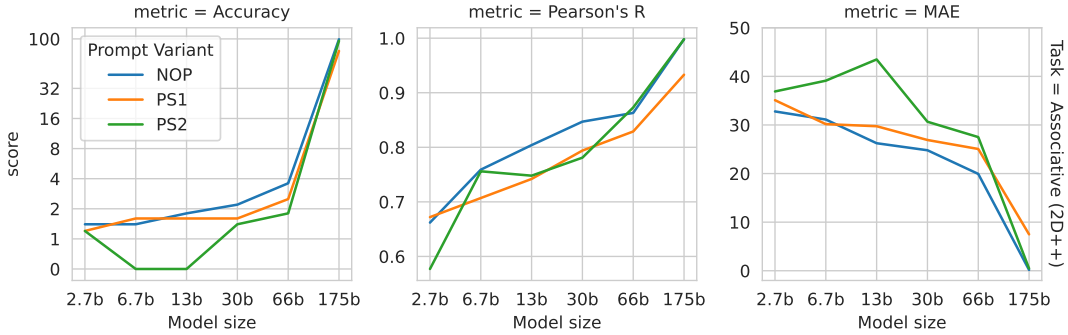


Figure 2: Associative property – the cumulative results on the 2D++ task for the three query variants, see Section 2.2.

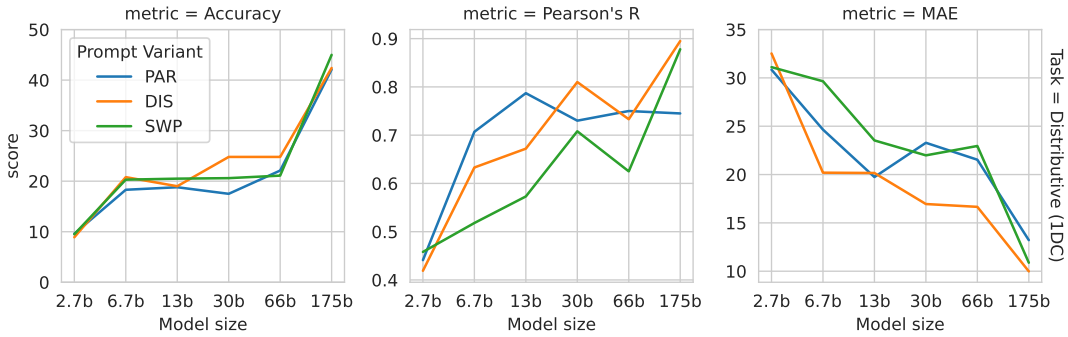


Figure 3: Distributive property – the cumulative results on the IDC task for the three prompt variants, see Section 2.2.

In Figure 2, we plot the results (Section 3) of the associative experiments and in Figure 3 of the distributive experiments. In Table 5, we include the MAE values of the 2.7B OPT model that were used to compute the *normalized MAE* values in Figure 1. In Table 6, we present a sample of the model outputs for the Rand prompt variant on the 3D+ task; see Section 3.

Table 5: MAE values of the smallest 2.7B OPT model computed for the tasks and prompt variants considered in Section 2.1.

Task	Variant		
	Rand	Sort	RevSort
2D++	32.1	31.1	33.9
2D×	1690.4	1641.9	1775.3
3D+	287.6	268.2	304.1

Table 6: Sample of model predictions.

Query	Correct Result	66B OPT Output	30B OPT Output	6.7B OPT Output
318 + 491 =	809	769	741	769
154 + 932 =	1086	1158	1662	1234
951 + 504 =	1455	1047	1154	1053
265 + 572 =	837	872	852	854