Pre-trained Language Models Learn Remarkably Accurate Representations of Numbers

Marek Kadlčík[♣†] **Michal Štefánik**[♡♣*†] **Timothee Mickus**[♡†]
Michal Spiegel[♣◇] **Josef Kuchař**[♣]

[♣]TransformersClub @ Faculty of Informatics, Masaryk University
[♡]Language Technology, University of Helsinki
[◇]Kempelen Institute of Intelligent Technologies
[†]Core contributors

Abstract

Pretrained language models (LMs) are prone to arithmetic errors. Existing work showed limited success in probing numeric values from models' representations, indicating that these errors can be attributed to the inherent unreliability of distributionally learned embeddings in representing exact quantities. However, we observe that previous probing methods are inadequate for the emergent structure of learned number embeddings with sinusoidal patterns.

In response, we propose a novel probing technique that decodes numeric values from input embeddings with *near-perfect accuracy* across a range of open-source LMs. This proves that after the sole pre-training, LMs represent numbers with remarkable precision. Finally, we find that the embeddings' precision, judged by our probe's accuracy, explains a large portion of LM's errors in elementary arithmetic, and show that aligning the embeddings with the pattern our probes discover can mitigate these errors.

1 Introduction

The landmark paper of Brown et al. (2020) showed that generic neural networks trained on text prediction alone could develop surprising arithmetic capabilities. In the years since, this observation has flourished into a large and vibrant field interested in the arithmetic reasoning capabilities of Transformers (Ahn et al., 2024), rife with research opportunities ranging from interpretability work (Akter et al., 2024) to solving Olympiad-level problems in mathematics (Li et al., 2025). Yet this work has also underscored the limitations of LMs on arithmetic tasks: Previous studies have explored how models can benefit from incorporating precise numeric representations (Feng et al., 2024), or offloading the arithmetic computation to a tool (Schick et al., 2023; Kadlčík et al., 2023), suggesting that their native learned representations are not reliable. Other works (Kantamneni and Tegmark, 2025; Zhou et al., 2024) have inspected such learned representations directly and tried to understand how models use them. Although model probing methods showed some success in interpreting numeric values from model representations (Zhu et al., 2025), the accuracy of those methods is low, suggesting that learned representations are highly imprecise.

In this paper, we push back on this interpretation: we show that a probe with the *right kind of inductive bias* can retrieve numeric information from number embeddings with *near-perfect accuracy* across an extensive range of LMs, spanning the Llama 3 (Grattafiori et al., 2024), Phi 4 (Abdin et al., 2024) and OLMo 2 (OLMo et al., 2025) series and ranging from 1B to 72B parameters. Given that number embeddings usually follow a sinusoidal wave-like pattern (Nanda et al., 2023; Kantamneni and Tegmark, 2025), this characteristic must be accounted for when designing probes.

We further show how these insights can be leveraged to improve performances on arithmetic reasoning: errors on addition and subtraction tasks can often be matched with an inability of the probe to retrieve the expected numerical information for a given embedding, and demonstrate that intervening on number embeddings such that they more cohesively follow the pattern of other number embeddings can directly improve arithmetic performances. Lastly, we document edge cases that do not fall within this previously understood pattern: in particular, OLMo2 32B (OLMo et al., 2025) learns embeddings that are not sinusoidal-like, despite a high success rate on arithmetic tasks.

[*]From October 2025 at National Institute of Informatics Research and Development Center for Large Language Models, Tokyo, Japan

2 Related Work

One line of work focuses on incorporating numerical values directly into token representations, providing LMs with a prior. Charton (2022) explores different number encodings based on scientific notation for training LM solvers of linear algebra problems. Golkar et al. (2023) propose representing numbers as a learned <NUM> token scaled by the number scalar value, demonstrating how models can adopt this scheme for regression tasks.

Another line of work investigates how models learn to represent and process numerical information. Nanda et al. (2023) show that a transformer with one-hot encoding trained from scratch on modular addition discovers Fourier basis and its computation is interpretable in trigonometric functions. Kantamneni and Tegmark (2025) discover an analogous circuitry for (non-modular) addition in a general pretrained language model, and find that its intermediate representations combine both linear and periodic components, reminiscent of a helix structure. Zhou et al. (2024) further identifies subcomponents of the addition circuitry implemented by the attention mechanism and feedforward layers. Zhu et al. (2025) demonstrate that hidden states of pretrained language models can be approximately decoded with a linear (or multi-layer) probe to estimate the logarithm of the number value. Although the probe outputs correlate with the target value, decoding achieves low accuracy. Recently, Levy and Geva (2025) show success in recovering the values of *digits* from internal representations of intermediate layers, hinting on a more generalized, circular pattern in representations of numbers.

In summary, prior works suggest that language models *attempt* to encode numerical information into token representations during pretraining, but their precision is rather limited. However, we hypothesize that this perception stems from inadequate probing methods, and learned representations are much more precise than previously estimated.

3 Recovering numerical information from number embeddings

We study LMs from the Llama 3 (Grattafiori et al., 2024), Phi 4 (Abdin et al., 2024), and OLMo 2 (OLMo et al., 2025) series, ranging from 1B to 72B parameters. Wide selection allows us to verify the validity of our observations across a panel of models sharing the characteristic of representing all integers between 0 and 999 with unique tokens.

Motivations. The central and foremost point to address is whether the embeddings representing specific numbers in LMs contain the numeric information of the value they represent. In practice, this is best addressed with a *probing* setup: If embeddings do contain numerical information, we should be able to learn a decoding function from number embedding to the corresponding integer value. Probing as a methodology comes with its own set of caveats: probes should be kept as simple as possible, and their expressivity should be compared against baseline benchmarks (Hewitt and Liang, 2019). Our specific use case adds further constraints: in particular, we have only one instance per LLM of each integer representation, viz., there is only one vector for the token 42. This rules out naive classifier implementations, as we aim for the probe to generalize to entirely unseen classes.

Probe architectures. We consider four probes:

$$f_{\text{lin}}(\mathbf{x}) = \mathbf{a}^T\mathbf{x} + b \tag{1}$$

$$f_{\text{log lin}}(\mathbf{x}) = \exp\left(\mathbf{a}^T\mathbf{x} + b\right) - 1 \tag{2}$$

$$f_{\text{sin}}(\mathbf{x}) = (\mathbf{W}_{\text{out}}\mathbf{S})^T(\mathbf{W}_{\text{in}}\mathbf{x}) \tag{3}$$

$$f_{\text{bin}}(\mathbf{x}) = (\mathbf{W}_{\text{out}}\mathbf{B})^T(\mathbf{W}_{\text{in}}\mathbf{x}) \tag{4}$$

where $\mathbf{a}$, b, $\mathbf{W}_{\text{in}}$, and $\mathbf{W}_{\text{out}}$ are learned parameters, whereas $\mathbf{S}$ and $\mathbf{B}$ are means of injecting inductive biases in the linear classifiers f_{sin} and f_{bin}:

$$\mathbf{S}_{ij} = \begin{cases} \sin(ie^j 1000/d) & \text{if } j \equiv 0 \mod 2 \\ \cos(ie^{j+1} 1000/d) & \text{if } j \equiv 1 \mod 2 \end{cases}$$

$$\mathbf{B} = \begin{bmatrix} 0 & \dots & 0 & 0 & 1 \\ 0 & \dots & 0 & 1 & 0 \\ 0 & \dots & 0 & 1 & 1 \\ \vdots & & & & \end{bmatrix}$$

I.e., the i^{th} row of $\mathbf{B}$ corresponds to the integer i expressed in binary, whereas $\mathbf{S}$ is defined as a Fourier basis, suggested by Zhou et al. as the hidden structure learned by pretrained models. The matrices $\mathbf{S}$ and $\mathbf{B}$ thus allows us to partition the label projection of the classifier into three components: a learned projection $\mathbf{W}_{\text{in}} : \mathbb{R}^d \to \mathbb{R}^h$ to project the number embeddings into a reduced low-dimensional space, a fixed matrix ($\mathbf{S}$ or $\mathbf{B}$) allowing us to encode integers using an *a priori* scheme, and a learned projection $\mathbf{W}_{\text{out}} : \mathbb{R}^d \to \mathbb{R}^h$ mapping these *a priori* representations onto the same space as the reduced embeddings. Intuitively, W_{in} uncovers the underlying hidden structure of the learned embeddings, while W_{out} expresses it in terms of interpretable a priori basis, which allows us to generalize to unseen tokens.

Implementation. We evaluate the probes in Equations (1) to (4) using a cross-validation setup with 20 folds. We report their accuracy measured by rounding the output of the regression probes Equations (1) and (2) to the nearest integer, or by retrieving the index of the row in $\mathbf{S}$ or $\mathbf{B}$ that maximizes the output distribution of the classifier probes Equations (3) and (4). We control the validity of our probes by ensuring that they reach an accuracy of 0 for standard Gaussian vectors as well as for a random permutation of the embeddings. Parameters for regressions are estimated using a least-squares algorithm; whereas our classifiers' parameters are optimized with Adam with a learning rate of 0.0001, weight decay of 0.001, and $\beta = (0.9, 0.999)$. We choose a hidden dimension of 100. The classifiers are optimized to distinguish output *only between training tokens*, and during testing, must choose between all tokens. The probes are optimized until loss converges on a validation split separate from the testing split.

We release an implementation and training recipes for the new probes, including all configurations we use, in the project's GitHub repository.[1]

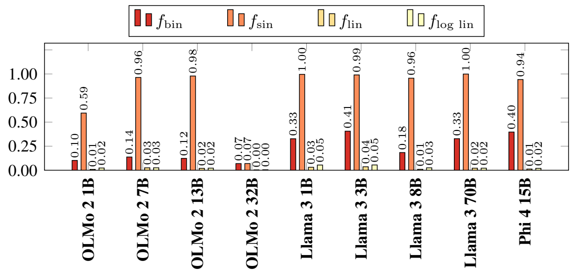


Figure 1: Overview of probes' accuracy (↑).

Results. We summarize performances, measured in terms of accuracy, in Figure 1. Crucially, we are almost systematically able to retrieve the integer value corresponding to the embedding's number with very high accuracy. Another salient observation is that f_{sin} consistently outperforms all other probe architectures including the regression probe used in previous work of Zhu et al. (2025), contradicting their finding that LMs learn to encode numbers linearly. Explaining the success of the Fourier basis, we note that other prior literature has suggested that sinusoidal features are used for arithmetic computation in LMs (Zhou et al., 2024). Adding onto this, we can also stress that, qualitatively, most of the models' whose number embeddings we survey here exhibit wave-like patterns in a PCA projection and have sparse Fourier transform, confirming regularity in the hidden structure. See Figures 2 and 3 in Appendix A.1 for visualizations of PCA and its Fourier transform. Notably, OLMo 2 32B is the only model with low resemblance of the pattern, which is consistent with the low performance of its sinusoidal probe.

Analysis. To verify that our sin-base probes indeed reach their superior accuracy by learning to extract a generalized, sin-like representation from models' representations, we analyse the encoded representations that trained sin probes produce as the output of $\mathbf{W}_{\text{in}}$. We experiment with two training settings: (i) using L1 regularization — encouraging sparsity, and (ii) using L2 regularization — encouraging the employment of a broader scale of input features. We note that in both of these settings, the probes achieve almost identical generalization capacity as assessed by their accuracy on unseen inputs (embeddings of numbers).

Figure 6 in Appendix A.3 displays the resulting representations for model embeddings of Llama3 1B associated with different numeric values. We can observe that the L1- and L2-regularized probes learn a substantially distinct representational pattern. We hypothesize that a main difference between probes trained with different regularizations is that the L1 probe learns to follow a broader scale of distinct frequencies, while the L2 probe follows similar frequencies shifted by a different constant. Nevertheless, in both of the cases, the probe learns a projection into a wave-like pattern across input numbers, thus successfully following their injected inductive bias.

4 Leveraging numerical information from number embeddings

Motivation. Having established that number embeddings do encode retrieval numerical information about the integers they represent, we now turn to **how this numerical information is leveraged** by LMs to perform arithmetic tasks. We study the zero-shot performances of a subset of our models on addition and subtraction tasks. We define our addition task as taking any pair of integers x_1, x_2 such that $0 < x_i < 500$ as input, and computing

[1]`https://github.com/prompteus/numllama`

the expected output $x_1 + x_2$. The subtraction task is defined by taking as inputs any pair x_1, x_2 such that $0 < x_2 < x_1 < 1000$, and computing the expected output $x_1 - x_2$.

Performance. To perform the arithmetic tasks, we conduct minimal prompt engineering: we systematically evaluate a handful of natural language prompts for their accuracy in a zero-shot setting, and then select the highest-performing for subsequent analyses. Due to computational costs, we ignore the two largest models (OLMo2 32B and Llama 3 70B). All prompts are listed in Appendix B, see Table 3a for addition and Table 3b for subtraction.

	OLMo2 1B	**OLMo2 7B**	**OLMo2 13B**	**Llama 3 1B**	**Llama 3 3B**	**Llama 3 8B**	**Phi 4 15B**
Add.	21.39	1.12	0.17	2.58	0.45	0.25	0.00
Sub.	28.12	0.36	0.16	1.43	0.03	0.01	0.00

Table 1: Overview of error rates (%, ↓) on arithmetic tasks in zero-shot setting.

An overview of the error rates from the LMs we study is listed in Table 1. As is apparent, most models achieve high degrees of performance (except for OLMO 2 1B); we also observe a trend towards fewer errors for models with more parameters.

Error analysis. To assess how numerical information and arithmetic performance are linked, we evaluate whether the errors we see in these arithmetic tasks are associated with defects of the number embeddings used as inputs.

We measure the error rate on the downstream addition and subtraction task in two separate cases – in the first case, both input tokens are decodable by the probe, in the second case, at least one value is not. The results can be seen in Table 2.

The results show that models tend to make more errors when the input embeddings are misaligned with the pattern used by the probe, as undecodable inputs lead to higher error rates in 8 out of 12 configurations. The effect is more prominent for models with substantial error rates, such as OLMo2 1B.

Direct intervention. We hypothesize that embeddings of tokens that our probes can not correctly decode diverge from the model's robust representa-

	OLMo2 1B	**OLMo2 7B**	**OLMo2 13B**	**Llama 3 1B**	**Llama 3 3B**	**Llama 3 8B**
Addition						
decodable	20.30	0.98	0.16	2.48	0.46	0.24
undecodable	23.33	1.84	0.20	14.86	0.28	0.28
Subtraction						
decodable	24.61	0.32	0.19	1.43	0.03	0.01
undecodable	31.45	0.53	0.13	0.90	0.04	0.00

Table 2: Downstream arithmetic error rate (%, ↓) given that (1) all tokens are decodable, and (2) at least one token is non-decodable. Results are measured on all possible input combinations. Phi is omitted because it does not make errors.

tion scheme and thus contribute to errors in arithmetic tasks. With this motivation, we test whether a direct intervention on the embeddings of these tokens can improve models' performance on arithmetic. In practice, we start from the f_{sin} probes described in Equation (3) and trained for Llama 3 1B and freeze all probe parameters. We then perform gradient descent to optimize the *embeddings* of all incorrectly decoded tokens (namely `0`, `4`, `977` and `999`) with respect to the probe decoding loss, aiming to align those tokens with the overall pattern discovered by the probe.

We finally measure how this embedding intervention impacts model error rate on addition and multiplication tasks involving these four tokens as one of the inputs or expected outputs, using the model's best-performing template (a set of our experimental templates is listed in Table 3a).

We find that in additions involving these assumably divergent tokens, our intervention reduces 26% of errors (from 17.6% to 13.0%). In multiplications, our intervention brings error reduction by 9.4% (from 8.5% to 7.7%). This experiment, while of an anecdotal scale determined by a low error rate of our probes, shows that more accurate probes of models' representations can also guide direct refinements of models' possibly imprecise embeddings, aligning them with the model's general hidden structure and bringing improvements in accuracy of the model's predictions.

5 Conclusion

In this paper, we have inspected the embedding representations for number tokens across a range of widely used open-source LMs. Our observations consolidate a growing body of studies showcasing how LMs learn sinusoidal hidden structure in number representations. Building upon this observation, we design a probing method leveraging this structure that decodes LMs' embeddings with *near-perfect* accuracy across multiple models, demonstrating that the quality of numeric representations in pretrained LMs was strongly underestimated in previous work. Still, we find a model (OLMo 2 32B) that deviates from this pattern, calling into question the generalizability of the conclusions of works such as Zhou et al.'s (2024). Finally, we show that the preciseness of embeddings relative to the sinusoidal pattern can explain a proportion of practical errors on arithmetic tasks, especially when models fail to align closely with this sinusoidal pattern.

Furthermore, we demonstrate improved accuracy on those tasks by aligning imprecise embeddings to the model's learned embedding pattern. To some extent, our findings curtail the validity of offloading approaches for numerical reasoning (Schick et al., 2023; Kadlčík et al., 2023): showing that their initial premise — of models not learn accurately representations of numbers — is incorrect. We hope that our findings will motivate future work to rigorously compare relative advantages of tool-using models in terms of computational efficiency, and challenge future work towards the *data* (Štefánik et al., 2024) and *architecture* refinements (Spiegel et al., 2025) accelerating more efficient learning of accurate representations of exact elements of language.

Acknowledgements

This work is supported by the Research Council of Finland through project No. 353164 "Green NLP – controlling the carbon footprint in sustainable language technology". This project has received funding from the European Union's (EU) Horizon Europe research and innovation programme under Grant agreement No. 101070350 and from UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee (grant number 10052546). The contents of this publication are the sole responsibility of its authors and do not necessarily reflect the opinion of the EU.

Limitations

Our work, while demonstrating the remarkable accuracy of number embeddings in pre-trained language models, comes with several limitations that warrant consideration for future research.

First, our probing method, though highly effective for many models, relies on an assumed hidden structure of models' learned representations, and therefore expects a broad a priori understanding of models' representation space. This necessarily limits the applicability of our approach to models where a known structure exists; Our results aim to show that some language models indeed *do* exhibit alternativel encoding schemes, exemplified by OLMo 2 32B that, ableit being highly accurate in arithmetics, can not be accurately probed by our sinusoidal probes.

Second, our intervention method was performed on a small-scale experiment, and its generalization across a large suite of models remains an object for future work.

Third, even when we do not perform any pre-training of models, reproducing our experiments requires access to computational resources. We estimate that replicating all our results requires around several hundred GPU hours.

Fourth, our analysis targets model embeddings. It is thus limited to single-token representations, and does not address the inner mechanisms of numeric information processing in large language model. This area also calls for further research.

While we recognize the ethical risks associated with AI research, given that our paper focuses on fundamentals of internal representations of numbers within pre-trained language models and their immediate impact on basic arithmetic tasks, broader societal ethical concerns like bias, discrimination, privacy, or job displacement are not directly relevant. Our research operates at a fundamental level of understanding how models encode numerical information, rather than exploring their application or misuse in real-world systems with downstream societal consequences.

References

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, James R. Lee, Yin Tat Lee, Yuanzhi Li, Weishung Liu, Caio C. T. Mendes, Anh Nguyen, Eric Price, Gustavo de Rosa, Olli Saarikivi, and

8 others. 2024. Phi-4 technical report. *Preprint*, arXiv:2412.08905.

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. *Preprint*, arXiv:2402.00157.

Mst. Shapna Akter, Hossain Shahriar, and Alfredo Cuzzocrea. 2024. Towards analysis and interpretation of large language models for arithmetic reasoning. In *2024 11th IEEE Swiss Conference on Data Science (SDS)*, pages 267–270.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared 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 M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Francois Charton. 2022. Linear algebra with transformers. *Transactions on Machine Learning Research*.

Guhao Feng, Kai Yang, Yuntian Gu, Xinyue Ai, Shengjie Luo, Jiacheng Sun, Di He, Zhenguo Li, and Liwei Wang. 2024. How numerical precision affects mathematical reasoning capabilities of llms. *Preprint*, arXiv:2410.13857.

Siavash Golkar, Mariel Pettee, Michael Eickenberg, Alberto Bietti, Miles Cranmer, Geraud Krawezik, Francois Lanusse, Michael McCabe, Ruben Ohana, Liam Parker, Bruno Régaldo-Saint Blancard, Tiberiu Tesileanu, Kyunghyun Cho, and Shirley Ho. 2023. xval: A continuous number encoding for large language models. In *NeurIPS 2023 AI for Science Workshop*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743, Hong Kong, China. Association for Computational Linguistics.

Marek Kadlčík, Michal Štefánik, Ondřej Sotolář, and Vlastimil Martinek. 2023. Calc-x and calcformers: Empowering arithmetical chain-of-thought through interaction with symbolic systems. In *Proceedings of the The 2023 Conference on Empirical Methods in Natural Language Processing: Main track*, Singapore, Singapore. Association for Computational Linguistics.

Subhash Kantamneni and Max Tegmark. 2025. Language models use trigonometry to do addition. *Preprint*, arXiv:2502.00873.

Amit Arnold Levy and Mor Geva. 2025. Language models encode numbers using digit representations in base 10. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 385–395, Albuquerque, New Mexico. Association for Computational Linguistics.

Zenan Li, Zhaoyu Li, Wen Tang, Xian Zhang, Yuan Yao, Xujie Si, Fan Yang, Kaiyu Yang, and Xiaoxing Ma. 2025. Proving olympiad inequalities by synergizing llms and symbolic reasoning. *Preprint*, arXiv:2502.13834.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. 2023. Progress measures for grokking via mechanistic interpretability. In *The Eleventh International Conference on Learning Representations*.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, and 21 others. 2025. 2 olmo 2 furious. *Preprint*, arXiv:2501.00656.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessí, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: language models can teach themselves to use tools. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Michal Spiegel, Michal Štefánik, Marek Kadlčík, and Josef Kuchař. 2025. Attend or perish: Benchmarking attention in algorithmic reasoning. *Preprint*, arXiv:2503.01909.

Michal Štefánik, Marek Kadlčík, and Petr Sojka. 2024. Concept-aware data construction improves in-context learning of language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 12335–12352, Bangkok, Thailand. Association for Computational Linguistics.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3319–3328. PMLR.

Tianyi Zhou, Deqing Fu, Vatsal Sharan, and Robin Jia. 2024. Pre-trained large language models use fourier features to compute addition. In *Advances in Neural Information Processing Systems*, volume 37, pages 25120–25151. Curran Associates, Inc.

Fangwei Zhu, Damai Dai, and Zhifang Sui. 2025. Language models encode the value of numbers linearly. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 693–709, Abu Dhabi, UAE. Association for Computational Linguistics.

A Supplementary visualization

A.1 Wave-like patterns in embeddings

Figure 2 displays the sinusoidal patterns in Llama 3 70B and OLMo2 13B after PCA dimensionality reduction. For clarity, we only include the first 16 principal components.

A.2 Explainability plots for arithmetic tasks.

Model behavior. To better explain the behavior of the LMs, we conduct a simple circuit analysis and a feature attribution experiment using integrated gradients (Sundararajan et al., 2017). For convenience, we focus on the two smaller models in our panel. OLMo 2 1B and Llama 3 1B.

Both experiments suggest one major difference between operand pairs leading to failure and to success: the probability assigned by the LLM to the predicted output token tends to be statistically lower when the model produces an incorrect output, see Figure 4. We also observe the same subset of heads being activated for failure and success on the arithmetic task. Besides the usefulness of this difference in probability mass for diagnostic purposes, these experiments also suggest a difference in *degree* rather than *kind* between failures and successes.

In Figure 5, we present an overview of head-level attribution of the logits in Llama 2 1B. The same heads in Layers 13 through 15 appear activated in all cases, playing the same inhibitor and booster roles. Incorrectly performed addition leads to a noisier overall pattern. Remarkably, we observe that activity occurs in the latter stages of the model, whereas input embeddings (layer 0) already contain precise numeric information, as per our probing experiments. This delayed processing may explain some of the errors we observe, despite the high accuracy of our probes in Section 3.

A.3 Analysis of sin-base probes' learned representations

In Figure 6, we can see that our newly proposed sin-like probes indeed learn to project input embeddings of models into an expected, generalized wave-like representation.

B Experimental details

1	"x_1+x_2 equals to "
2	"The result of x_1+x_2 is "
3	"The result of x_1 plus x_2 is "
4	"The result of x_1 plus x_2 = "
5	"The result of x_1 plus x_2 ="
6	"x_1 plus x_2 equals to "
7	"x_1+x_2="
8	"x_1 plus x_2 equals "
9	"x_1 plus x_2 is equal to "
10	"x_1+x_2 equals "
11	"x_1+x_2 is equal to "
12	"x_1 plus x_2 equals "
13	"x_1 plus x_2 is equal to "

(a) Prompts considered for addition task. x_1 and x_2 are placeholders for the augend and the addend. Prompts are delimited by double quotes; trailing white-space is significant.

1	"The result of x_1 minus x_2 is "
2	"The result of x_1 minus x_2 = "
3	"The result of x_1 minus x_2 ="
4	"x_1 minus x_2 equals to "
5	"x_1-x_2="
6	"x_1 minus x_2 equals "
7	"x_1 minus x_2 is equal to "
8	"x_1-x_2 equals "
9	"x_1-x_2 is equal to "
10	"x_1 minus x_2 equals "
11	"x_1 minus x_2 is equal to "

(b) Prompts considered for subtraction task. x_1 and x_2 are placeholders for the minuend and the subtrahend. Prompts are delimited by double quotes; trailing white-space is significant.

Table 3: Prompts considered for engineering of arithmetic zero-shot setting.

We conduct a minimal prompt optimization in Section 4 to maximize the performances on arithmetic task. For all models below 20B parameters, we explore the prompts listed in Table 3a and Table 3b, and report results with the highest performance in a zero-shot setting in Section 4.

The most successful prompts for addition are prompt #4 for Llama 3 1B, 2B and 8B as well as OLMo2 1B, and prompt #3 for OLMO2 7B and 13B. As for subtraction, the most effective prompt was prompt #1 for Llama 3 1B, OLMo2 1B and 7B, and prompt #2 for Llama 3 3B and 8B as well as OLMo2 13B.

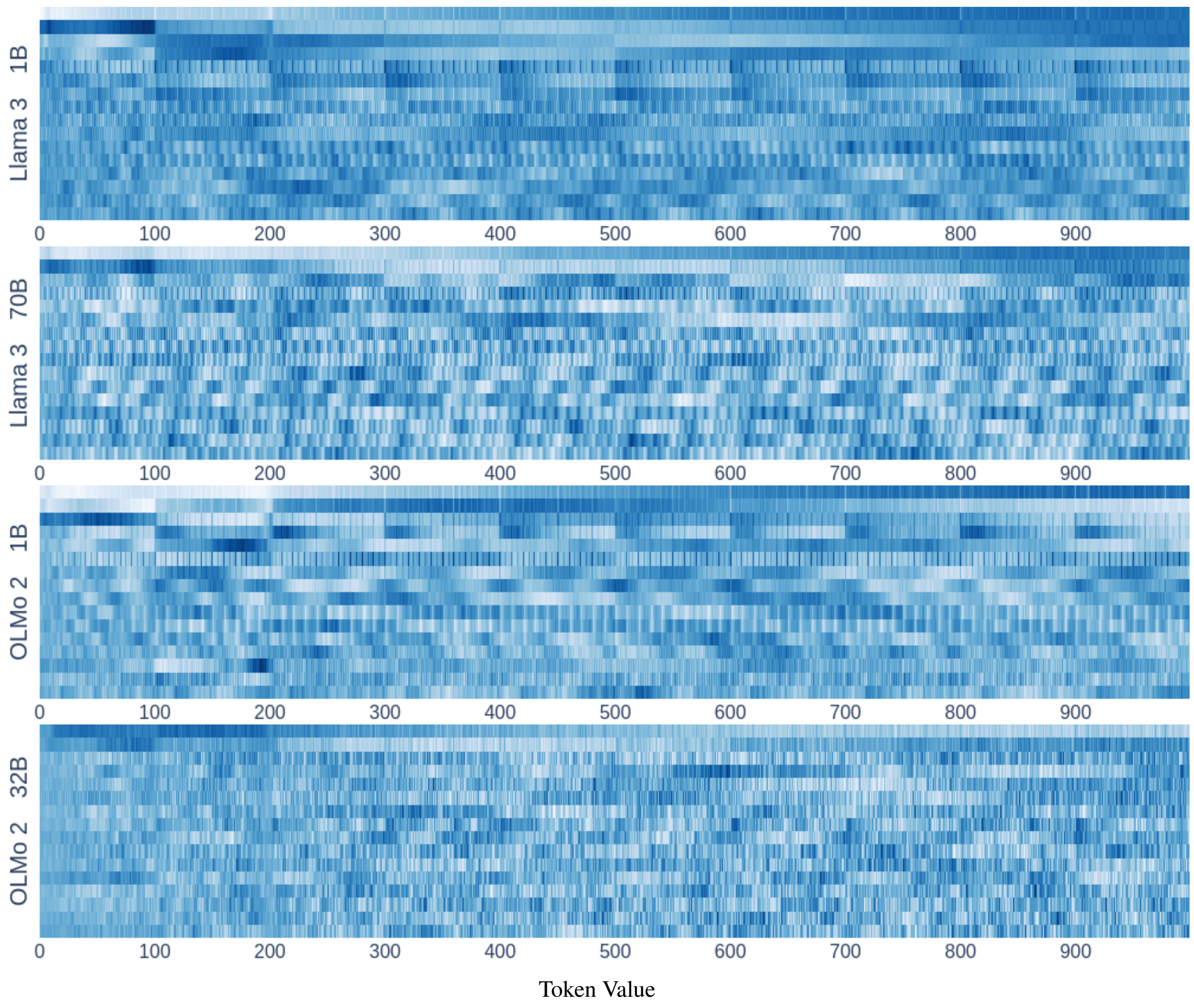
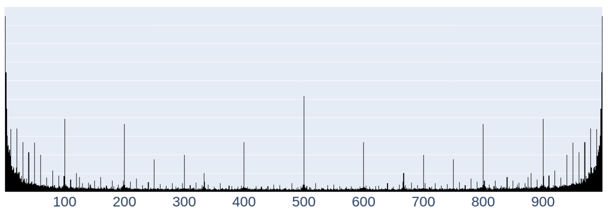


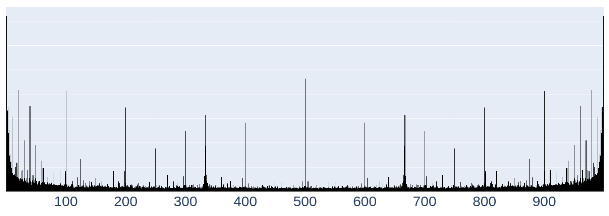
Figure 2: Visualization of PCA (DIM=16) reduced number embeddings, selected models. Although most model exhibit relatively regular wave-like patterns, OLMO 2 32B exhibit little regularity.

C Disclosure of usage of AI assistance

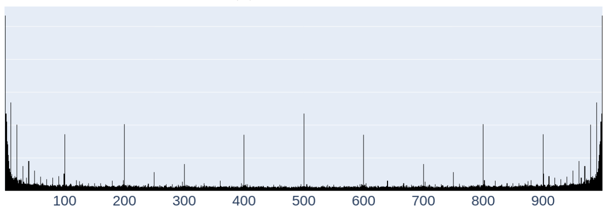
We disclose that we used AI assistance during implementation of this work and its writing. Specifically, we used AI-based code auto-completion (Github Copilot) for increasing productivity of programming, and conversational chatbots (OpenAI ChatGPT, Google Gemini) for improving grammar and fluency of the text. We guarantee that all content is original and factually accurate.



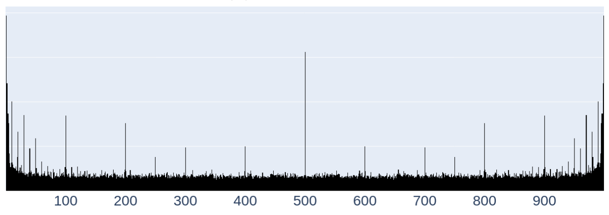
(a) Llama 3 1B



(b) Llama 3 70B

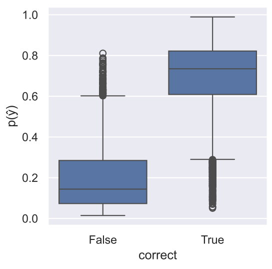


(c) OLMo 2 1B

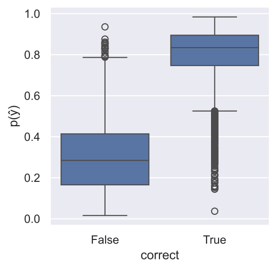


(d) OLMo 2 32B

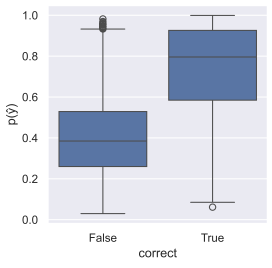
Figure 3: Maximal contribution (magnitude) of each Fourier base frequency's to embedding features in PCA (d=128) reduced space. Sparsity in this plot indicates strong regularity in the hidden structure of model embeddings. OLMo 2 32B has noticeably stronger contribution of all low-contribution frequencies, indicating high irregularity.



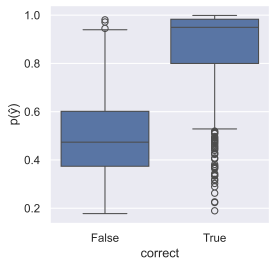
(a) Llama 3 1B, addition.



(b) Llama 3 1B, subtraction.

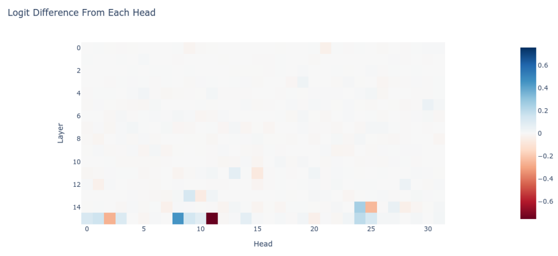


(c) OLMo 2 1B, addition.

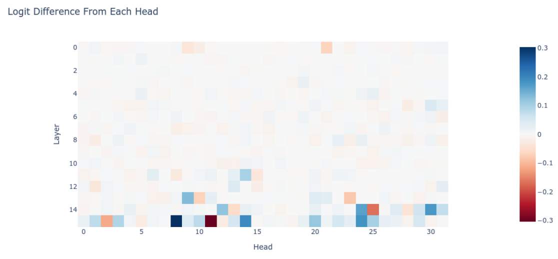


(d) OLMo 2 1B, subtraction.

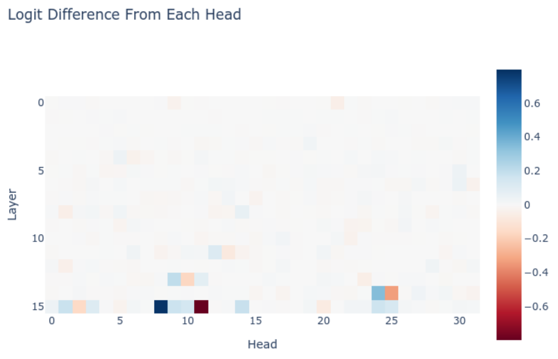
Figure 4: Probability mass on the predicted output token when the LLM yields a correct vs. incorrect answer.



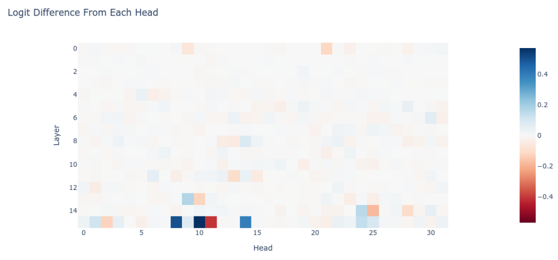
(a) Llama 3 1B, addition performed correctly.



(b) Llama 3 1B, addition performed incorrectly.

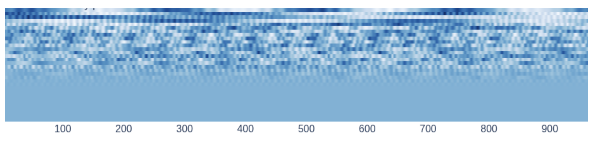


(c) Llama 3 1B, subtraction performed correctly.

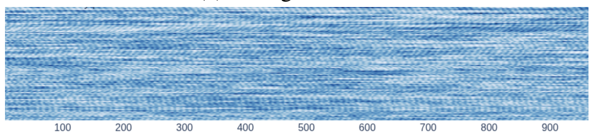


(d) Llama 3 1B, subtraction performed incorrectly.

Figure 5: Head activations across arithmetic tasks for Llama 3 1B, broken down by task (addition and subtraction) and success (correct or incorrect computation.



(a) L1 regularization



(b) L2 regularization

Figure 6: Hidden representations of sin-base probes for numeric input embeddings of Llama 3 1B model, after training with different regularization strategies show that our sin-base probes learn to project numeric embeddings into a generalized, wave-like representation used as target inductive bias.