Estimating Numbers without Regression

Abstract

Despite recent successes in language models, their ability to represent numbers is insufficient. Humans conceptualize numbers based on their magnitudes, effectively projecting them on a number line; whereas subword tokenization fails to explicitly capture magnitude by splitting numbers into arbitrary chunks. To alleviate this shortcoming, alternative approaches have been proposed that modify numbers at various stages of the language modeling pipeline. These methods change either the (1) notation in which numbers are written (e.g. scientific vs decimal), the (2) vocabulary used to represent numbers or the entire (3) architecture of the underlying language model, to directly regress to a desired number.

In this work, we show that a potential trade-off to the more complex architectural changes is to simply change the model's vocabulary instead, e.g. introduce a new token for numbers in range 10-100. In the context of masked number prediction, we find that a carefully designed tokenization scheme is both the simplest to implement and sufficient i.e. with similar performance to the state-of-the-art approach that requires making significant architectural changes. Finally, we evaluate the various number representation schemes on the downstream task of numerical fact estimation (for Fermi Problems) in a zero-shot setting and find similar trends i.e. changes at the tokenization level achieve near state-of-the-art results while requiring minimal resources compared to other number representation schemes.

1 Introduction

The standard practice in the natural language processing (NLP) community is to process numbers in exactly the same manner as words. This counter-intuitive treatment of numbers leads to their inaccurate representation and therefore, limited numerical understanding of large-scale language models (LMs) (Razeghi et al., 2022). To illustrate, a number like $\$799$ is *subword* tokenized (Sennrich et al., 2016) as 79 and $\#\#9$. Such a tokenization method, by construction, prevents accurately modeling the relationship of this number with others close on the number line say, $\$800$, as the surface forms share no common tokens.

Many alternatives have been proposed to capture the scalar magnitude of numbers (Thawani et al., 2021b). All number decoders proposed to capture the magnitude of numbers fall into one of the following categories, corresponding to changes in 1) **notation** (e.g. scientific vs decimal) or 2) **vocabulary** (e.g. introducing new tokens that denote all numbers within a specified range) or 3) **architectural** changes (e.g. directly regressing to a number). Figure 1 shows various alternative number representation methods ordered by increasing levels of intervention on a typical NLP pipeline.

We find that applying the vocabulary-level changes leads to near state-of-the-art performance requiring no additional pre-training or architectural changes. This is a surprising yet useful finding, which can substantially speed up adoption of numeracy into any given language model. Any arbitrary LM can be made *numerate* by simply tokenizing numbers on the number line.

We further evaluate the number representation schemes on their ability to generalize to downstream tasks – in this case, numerical fact estimation in the context of solving Fermi problems (Kalyan et al.,

36th Conference on Neural Information Processing Systems (NeurIPS 2022) Workshop on MATH-AI.

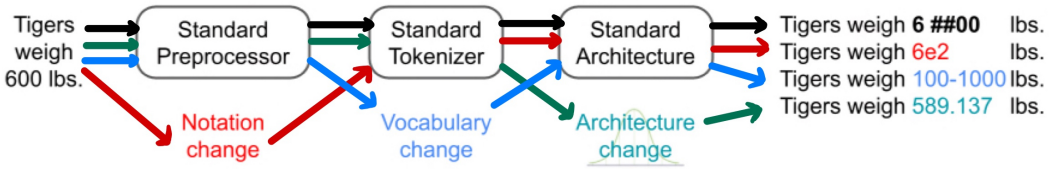

Figure 1: Alternative number representations change one of the three stages in the NLP pipeline.

2021). We find trends similar to the task of masked number prediction demonstrating the utility of the simple yet effective tokenization scheme in the decoding setting.

2 Background

Mathematics includes not only performing *exact* arithmetic over operands, but also a comprehensive understanding of *approximate* numeracy. This paper focuses on the latter, with the setting of masked number prediction (MNP) in natural language. In this section, we introduce existing classes of number decoders and discuss the trade-offs involved in using them.

Subword. The default way that language models decode numbers is the same way as words, one subword at a time, e.g. , the number 600 could be decoded as two individual tokens 6 and ##00.

Notation-change. Here, the numbers are represented in an alternative notation by preprocessing text before feeding into any off-the-shelf tokenizer and model. We consider the following variations: **1. Scientific:** Using scientific notation, e.g., $6e2$ (where 6 is the mantissa and 2 is the exponent) in lieu of the usual decimal notation was first proposed by Zhang et al. (2020). In this work, we closely follow their version with minor implementation level changes. Note that following the notation change, the tokenizer nevertheless splits it into subwords. **2. Digits:** Here, the number is split into its constituent digits or characters, e.g., 600 becomes 6 0 0. This approach offers a consistent decomposition of numbers into digits as opposed to arbitrary subword segmentation, and has been proven effective on simple numeric probes as well as arithmetic word problems Geva et al. (2020).

Vocabulary change. Unlike words, the notion of distance or similarity is more obviously defined for numbers in terms of their separation on the number line, a cognitive tool that human beings are known to intuitively use to process numeracy (Dehaene, 2011). This forms the basis of a change of vocabulary: numbers within a specified range are collapsed into a single token (e.g. 100-1000) – at the cost of precise representation of numbers. While this approach has already been used in the context of *encoding* numbers (Berg-Kirkpatrick and Spokoyny, 2020; Thawani et al., 2021a), our work is the first to use and study this approach when outputting or *decoding* numbers. This approach does not modify the LM architecture, instead merely adds new tokens to the vocabulary.

Architecture change. Finally, several recent methods have modified the underlying language model to emit continuous values when predicting numbers. At their core, they operate by regressing to the desired number conditioned on the language context. See Berg-Kirkpatrick and Spokoyny (2020) for a thorough comparison within this class of methods. We directly compare against their best variant: Discrete Latent Exponents (DExp), which first models the exponent part of a number as a multinomial, then uses it to parameterize a truncated log normal distribution to sample the mantissa, a continuous value. Note that this is the highest level of intervention possible, thereby making the method ineffective whenever the underlying LM architecture is not accessible, say over an API.

3 Experimental setup

We evaluate different number decoders on the task of masked number prediction (MNP): Given a sentence with a masked number (e.g. *"Tigers weigh [MASK] lbs."*), the model must predict a number as close as possible to the ground truth (e.g. 600). Before analyzing the performance of different models, we first describe the datasets, metrics, baselines, and models used.

Datasets: We follow Berg-Kirkpatrick and Spokoyny (2020) to finetune and evaluate our models on three datasets – Financial News Articles (FinNews), its subset containing mostly price-based numbers

Metrics	FinNews E-Acc ↑	FinNews LogMAE↓	FinNews-$ E-Acc↑	FinNews-$ LogMAE↓	SciDocs E-Acc↑	SciDocs LogMAE↓
Baselines						
Train-Mean	$1.0 \pm 0.1\%$	7.69	$6.0 \pm 0.4\%$	4.68	$0.0 \pm 0.0\%$	8.81
Train-Median	$5.5 \pm 0.2\%$	1.88	$10.6 \pm 0.5\%$	2.66	$49.5 \pm 0.7\%$	0.83
Train-Mode	$24.2 \pm 0.4\%$	2.02	$8.1 \pm 0.5\%$	6.30	$49.5 \pm 0.7\%$	1.00
Subword-Pad8	$63.6 \pm 0.5\%$	0.68	$29.1 \pm 0.8\%$	1.36	$68.0 \pm 0.6\%$	0.68
Notation-change						
Digit-Pad17	$52.2 \pm 0.5\%$	0.93	$33.0 \pm 0.8\%$	1.37	$55.1 \pm 0.5\%$	0.91
Scientific-Pad8	$52.5 \pm 0.5\%$	0.84	NA	NA	$71.1 \pm 0.6\%$	0.66
Vocabulary-change						
Vocab-AM	$74.4 \pm 0.4\%$	0.65	$57.1 \pm 0.8\%$	0.93	$81.2 \pm 0.5\%$	0.51
Vocab-GM	$73.7 \pm 0.4\%$	<u>0.60</u>	$57.0 \pm 0.8\%$	<u>0.92</u>	$81.3 \pm 0.5\%$	<u>0.44</u>
Architecture-change	Berg-Kirkpatrick and Spokoyny (2020)					
DExp	$74.6 \pm 0.4\%$	**0.50**	$57.5 \pm 0.8\%$	**0.89**	$81.2 \pm 0.5\%$	**0.39**

Table 1: Order of magnitude accuracy (E-Acc) and Log Mean Absolute Error (LogMAE) on test sets.

(FinNews-$), and Scientific Articles (SciDocs) (Lo et al., 2020); all numbers in these datasets lie between 1-10^{16}. All datasets are accessible (MIT License) at `https://github.com/dspoka/mnm`.

Metrics: We evaluate using two metrics – a) Exponent Accuracy (E-Acc) that checks whether the predicted answer is of the same order of magnitude as the ground truth and b) Log Mean Absolute Error (LogMAE). Confidence Intervals for Exponent Accuracy, a classification metric, are reported as the Wilson Score Interval (Wilson, 1927): $a \pm z\sqrt{a(1-a)/n}$, where a is the accuracy, z is the constant (equal 2.58 for 99% CI), and n is the number of observations in the respective test set.

Baselines: Our primary baseline is the standard approach of subword tokenization. We require each number prediction to be 8 tokens long, with appropriate padding, to be able to fairly represent all numbers in our range. Additionally, we evaluate on three trivial baselines that make a constant prediction corresponding to the mean, median, and mode of all numbers in the training set.

Models: We compare against both notation-level changes i.e. scientific and digit, with a padding of 8 and 17 respectively. Among the approaches that introduce architectural changes, we compare against the SotA method of DExp (see previous section). Finally, we compare against two variations that introduce vocabulary level changes – both discretize the number line with logarithmically sized bins (with base 10). The two variants differ in how the mantissa is chosen – the arithmetic mean (5) or the geometric mean ($\sqrt{10}$), named Vocab-AM and Vocab-GM, respectively.

Implementation: Following the setup in Berg-Kirkpatrick and Spokoyny (2020), our base language model is 12-layer BERT-base and we fine-tune all models with a batch-size of 32 for 10 epochs. We use early stopping with a patience of three on the validation loss. We use two learning rates 3e-5 and 1e-2 for all pretrained parameters and newly added parameters respectively. For legibility, we skip variance estimates (bootstrapped over 10 samples, each of size 75% of the test set) in Table 1 – they range from 1e-7 to 1e-5. Please see Appendix A.3 for more details.

4 Results

We **bold-face the best** and <u>underline the next best</u> LogMAE scores in each column (dataset), and we highlighted exponent accuracies that are within 99% confidence of the SotA E-Acc. NA denotes subword models which were unable to emit valid numbers for at least 50% of the examples.

Intrinsic results (Table 1) We find that the straightforward, change of notation approaches are inferior to the subword baseline. This is in contrast to prior work on extrapolating the arithmetic abilities of language models by notation changes (Nogueira et al., 2021; Geva et al., 2020). It suggests that simple pre-processing changes of notation are not sufficient for contextual understanding of numbers for language modeling. Next, we find that the vocabulary change methods (Vocab-AM/GM) are at par or better than the architectural change model (DExp). The improvement from subword to the DExp model, is achievable (within statistical bounds) without modelling the mantissa at all!

Fermi-Real 510 egs.	trained on FinNews E-Acc ↑	LogMAE↓	trained on FinNews-$ E-Acc↑	LogMAE↓	trained on SciDocs E-Acc↑	LogMAE↓
Sub-Pad8	$26 \pm 5\%$	2.38	$16 \pm 4\%$	3.17	$26 \pm 5\%$	2.84
Dig-Pad17	$19 \pm 5\%$	2.58	NA	NA	$23 \pm 5\%$	2.87
Sci-Pad8	$25 \pm 5\%$	2.93	NA	NA	$20 \pm 5\%$	2.75
Vocab-AM	$32 \pm 5\%$	2.19	$24 \pm 5\%$	**2.42**	$27 \pm 5\%$	2.42
DExp	$32 \pm 5\%$	**2.13**	$25 \pm 5\%$	2.51	$28 \pm 5\%$	**2.40**
Fermi-Syn 3437 egs.	trained on FinNews E-Acc ↑	LogMAE↓	trained on FinNews-$ E-Acc↑	LogMAE↓	trained on SciDocs E-Acc↑	LogMAE↓
Sub-Pad8	$29 \pm 2\%$	2.89	$19 \pm 2\%$	3.25	$39 \pm 2\%$	2.83
Dig-Pad17	$23 \pm 2\%$	2.93	NA	NA	$41 \pm 2\%$	2.87
Sci-Pad8	$26 \pm 2\%$	3.06	NA	NA	$27 \pm 2\%$	2.76
Vocab-AM	$39 \pm 2\%$	2.61	$41 \pm 2\%$	**2.42**	$48 \pm 2\%$	2.52
DExp	$39 \pm 2\%$	**2.44**	$41 \pm 2\%$	2.44	$48 \pm 2\%$	**2.48**

Table 2: Downstream performance of main methods over fact estimation for solving Fermi Problems.

Downstream transfer (Table 2) Given such trends in masked number prediction, we are interested in the utility of these models on a downstream number prediction task. For this purpose, we evaluate on numerical fact estimation using the Fermi Problems dataset (Kalyan et al., 2021)[1], which consists of challenging estimation problems such as *"How many tennis balls fit in a school bus?"* Solving such questions require estimating numeric facts e.g. *volume of tennis ball* & *length of bus*.

We evaluate our models (trained with different number decoders on one of the three datasets) in a zero-shot setting on such annotated facts provided as part of both the real and synthetic datasets part of the Fermi problem dataset. The task setup is of masked number prediction as before, e.g., "*the size of a tennis ball is* `[MASK]` *cubic centimeters.*" We find similar trends as before i.e. change of notation is insufficient while vocabulary-change approaches are equal or better than architectural changes – highlighting that most of the gains could be retained by simply tokenizing in number space.

5 Related work

The NLP community has recently proposed several ways of improving the numeracy of language models, including architectural and notation interventions. Several such methods are aimed at helping LMs extrapolate easily to larger numbers (Kim et al., 2021) or for improving their arithmetic skills (Nogueira et al., 2021). We restrict our analysis to the task of *approximately* decoding numbers in MNP setting, which requires different methods and metrics from the tasks that instead evaluate their *exact* arithmetic skills (Thawani et al., 2021b).

The method we highlight in this paper i.e. change of vocabulary to tokenize numbers on a log-scaled number line, has been previously used in different settings. Others have shown the benefits of using such exponent embeddings as *number encoders* for language models, whether it be for the task of masked number prediction (Berg-Kirkpatrick and Spokoyny, 2020) or masked word prediction (Thawani et al., 2021a). Our work extends these results with further evidence of the representational power gained by simply tokenizing numbers on the number line.

6 Conclusion

Subword tokenization, the standard approach to representing numbers leads to inaccurate numerical understanding. In this work, we analyze number representation approaches that make notational (e.g. scientific vs. decimal), vocabulary (i.e. tokenizing on the number line), and architectural changes (i.e. regressing to the number). We find that tokenization on the number line achieves near or better than state-of-the-art results while requiring minimal resources as opposed to making architectural changes. This finding allows language models to conveniently improve their numeracy, including cases where users may not have access to the model's architecture and are only provided a typical finetuning regime with small changes to the tokenizer's vocabulary. Finally, we find similar trends in the challenging setting of numerical fact estimation for solving Fermi Problems – indicating that vocabulary-change is sufficient to represent approximate numbers effectively and with minimal effort.

[1]Accessible at `https://allenai.org/data/fermi` with CC BY License.

References

Taylor Berg-Kirkpatrick and Daniel Spokoyny. 2020. An empirical investigation of contextualized number prediction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4754–4764, Online. Association for Computational Linguistics.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502, Dublin, Ireland. Association for Computational Linguistics.

Stanislas Dehaene. 2011. *The number sense: How the mind creates mathematics*. OUP USA.

Mor Geva, Ankit Gupta, and Jonathan Berant. 2020. Injecting numerical reasoning skills into language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 946–958, Online. Association for Computational Linguistics.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Devin Johnson, Denise Mak, Andrew Barker, and Lexi Loessberg-Zahl. 2020. Probing for multilingual numerical understanding in transformer-based language models. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 184–192, Online. Association for Computational Linguistics.

Ashwin Kalyan, Abhinav Kumar, Arjun Chandrasekaran, Ashish Sabharwal, and Peter Clark. 2021. How much coffee was consumed during EMNLP 2019? fermi problems: A new reasoning challenge for AI. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7318–7328, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jeonghwan Kim, Giwon Hong, Kyung-min Kim, Junmo Kang, and Sung-Hyon Myaeng. 2021. 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. Association for Computational Linguistics.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Michael Kinney, and Daniel S. Weld. 2020. S2orc: The semantic scholar open research corpus. In *ACL*.

Mikhail Nefedov. 2020. Dataset for evaluation of mathematical reasoning abilities in russian. In *Conference on Artificial Intelligence and Natural Language*, pages 135–144. Springer.

Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. 2021. Investigating the limitations of transformers with simple arithmetic tasks.

Yasaman Razeghi, Robert L Logan IV, Matt Gardner, and Sameer Singh. 2022. Impact of pretraining term frequencies on few-shot reasoning. *arXiv preprint arXiv:2202.07206*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Avijit Thawani, Jay Pujara, and Filip Ilievski. 2021a. Numeracy enhances the literacy of language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6960–6967, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Avijit Thawani, Jay Pujara, Pedro A. Szekely, and Filip Ilievski. 2021b. Representing numbers in NLP: a survey and a vision. *CoRR*, abs/2103.13136.

Edwin B. Wilson. 1927. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212.

Xikun Zhang, Deepak Ramachandran, Ian Tenney, Yanai Elazar, and Dan Roth. 2020. Do language embeddings capture scales? In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4889–4896, Online. Association for Computational Linguistics.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] We describe limitations in Appendix B due to space constraints.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] We describe limitations in Appendix B due to space constraints.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We have included zipped code in supplementary materials, and will link to our repository in the camera ready.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 2.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We report statistical significance tests using Wilson Score Confidence Interval in Tables 1 and 2. We do not retrain methods with multiple random seeds to preserve computation budget but we found our run of the DExp method (Berg-Kirkpatrick and Spokoyny, 2020) to match the original reported results (on a different random seed) within 0.1% accuracy scores.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Please see Appendix A.3.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 3
   (b) Did you mention the license of the assets? [Yes] See Section 3. We link to MIT Licensed online repository which contains the data.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] The data used originate from hand-annotated or public Kaggle datasets or public scientific papers.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] It does not contain personally identifiable information.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Input	FY2018 Earnings per share view $ **[MASK]** , revenue view . . .	Daniels maintains Cohen paid her $130000 via essential consultants to hush up a **[MASK]** s. encounter with Trump.
True	1.63	2006
Sub	1000000	1
DExp	2.695	2792.66
Ours	1-10	1k-10k

Table 3: Example predictions from FinNews dev set. Ours (Vocab-GM) and DExp estimate numbers in the same order of magnitude as ground truth; but the subword baseline (Sub) is far off.

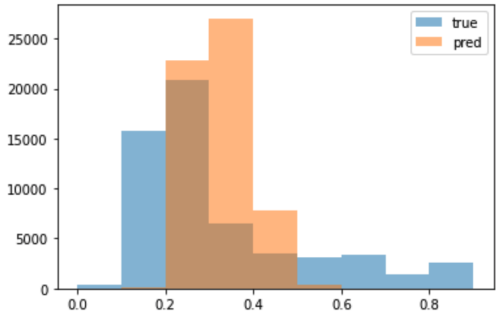

Figure 2: Histogram of mantissas for the 58K sentences in FinNews dev set (true) and corresponding predictions by DExp (pred). See Section 4 for details.

A Appendix

A.1 Implementation Details

Each of our experiments took a few hours on NVIDIA Quadro RTX 8000 GPU (one per experiment). We report results on the same random seed across models. We were able to reproduce DExp result scores exactly up to 1 decimal place. Note that we only compare number decoders and not the encoders – therefore, when numbers are present in the input, standard encoding schemes are used. For approaches with changes to vocabulary and architecture, we follow (Berg-Kirkpatrick and Spokoyny, 2020) and use exponent embeddings to encode numbers (with no shared parameters with the decoder's tokens) and for approaches with notation changes, we use subword tokenization.

The key contribution of this work is to highlight the possibility of achieving near state-of-the-art results from Berg-Kirkpatrick and Spokoyny (2020) with a much simpler method. Thus, we used the same hyperparameters and extend their code[2] for most of our experiments. Please refer to Section 3 in their paper for dataset details.

With scientific notation, a previous approach NumBERT (Zhang et al., 2020) denotes 329 as 329 `[EXP]` 2. However, we find that representing the same instead as $3x29$ where 'x' is the common English alphabet, works better in practice.

A.2 Example predictions

Finally, Table 3 shows some representative examples from FinNews dataset where the Subword baseline's estimate is far off from the ground truth, whereas predictions of both DExp and Vocab-GM are within the correct order-of-magnitude.

A.3 Comparing Mantissas

To study why Vocabulary change is nearly as good as Regression, we dig deeper into the only component that differentiates our proposed Vocab-AM/GM models from the state-of-the-art DExp:

[2] `https://github.com/dspoka/mnm`

Metrics	FinNews		FinNews-$	
	E-Acc ↑	LogMAE↓	E-Acc↑	LogMAE↓
Vocab-AM	<u>74.40</u>	0.65	<u>57.14</u>	0.93
Vocab-GM	73.70	<u>0.60</u>	56.99	<u>0.92</u>
DExp-21	72.2	0.51	47.6	1.04
DExp	**74.56**	**0.50**	**57.50**	**0.89**

Table 4: Comparing variable sized numeric vocabulary (Vocab-21) with static variants and architecture change (DExp) shows no gains, except in LogMAE over Financial News dataset. See §A.4 for details.

mantissas. We plot the mantissas from DExp's predictions against the ground truth (FinNews dev set) in Figure 2. We find that in the naturally occurring datasets, the leading digit of numbers is likely to be small (Benford's Law) and the mantissa peaks around 2, owing to the frequent mentions of years $(2000 - 2022)$ from our current millennium (recency bias). This rather simple distribution of numbers in the real world helps our static Vocab-AM/GM models perform at par with the state-of-the-art DExp without making any architectural changes to the underlying language model.

A.4 Variable Length Binning

Motivated by the success of frequency-based surface-level vocabulary, we further experiment with an extension of the vocabulary change. Instead of collapsing numbers into order-of-magnitude or exponent bins which are equally spaced on the log scale, we find bins such that their overall frequencies in a corpus are more uniform. By arranging all numbers from the FinNews corpus in ascending order and dividing them into equal sized (by frequency) bins, we get the following variable length vocabulary: $1, 2, 3, 4, 6, 10, 14, 21, 30, 31, 70, 415, 2011, 2017, 2018, 5131, 30207, 252178, 1700000, 30000000, 1152337024$. With these 21 bins[3], we retrain the Vocab-AM method and compare with our earlier static bins which corresponded to powers of 10: $1, 10, 100, \ldots$.

Table A.4 shows the results on both FinNews and FinNews-$ datasets. We observe that this vocabulary, despite having a more uniform distribution of numbers, does not do any better than the original naive method (except on LogMAE over the FinNews dataset). We note this as further evidence of the robustness of merely tokenizing on the number line. If variable sized bins were crucial for strong performance, practitioners may have had to relearn the model's numeric vocabularies based on different datasets and corpus frequencies. On the other hand, the order-of-magnitude-10 vocabulary is a simple, intuitive and robust method that competes with performance of state-of-the-art architectural-change number decoders.

A.5 Neuron Probing

In this subsection, we further probe how numeracy is stored in the feed forward layers of language models. Previous work along these lines (Geva et al., 2021) have shown promise in interpreting the knowledge stored in language models by finding individual neurons in feed forward layers that are triggered by specific patterns of input. We apply this analysis to find some such neurons, if any, which can effectively and efficiently capture the magnitude of a masked number.

Figure 3 shows the Precision-Recall curves for the state-of-the-art DExp model on the task of predicting masked numbers has an exponent of 3, i.e. it is between 1000 and 10,000. We say a neuron has been triggered if it is among the top 50 activated ones (out of 3072) in that layer for the input mask token. Recall is then defined as the fraction of times when this neuron was triggered for all masked numbers with an exponent of 3. Precision is defined as the fraction of times when the exponent was 3 for all the times that the specific neuron was triggered. We find that some individual neurons, such as the 650th neuron in the 10th layer of finetuned DExp has a very high precision and recall. It alone can predict whether the order of magnitude is 3, with an F1 score of above 0.7.

The presence of such precise individual neurons that capture order-of-magnitude numeracy in DExp model further suggests why tokenizing the number line on the log scale is a naturally suited number representation. This analysis shows promise in interpreting results of number representations in language models and possibly even causing interventions to update its beliefs (Dai et al., 2022).

[3]We manually tune this hyperparameter so as to obtain a near-uniform distribution of number occurrences.

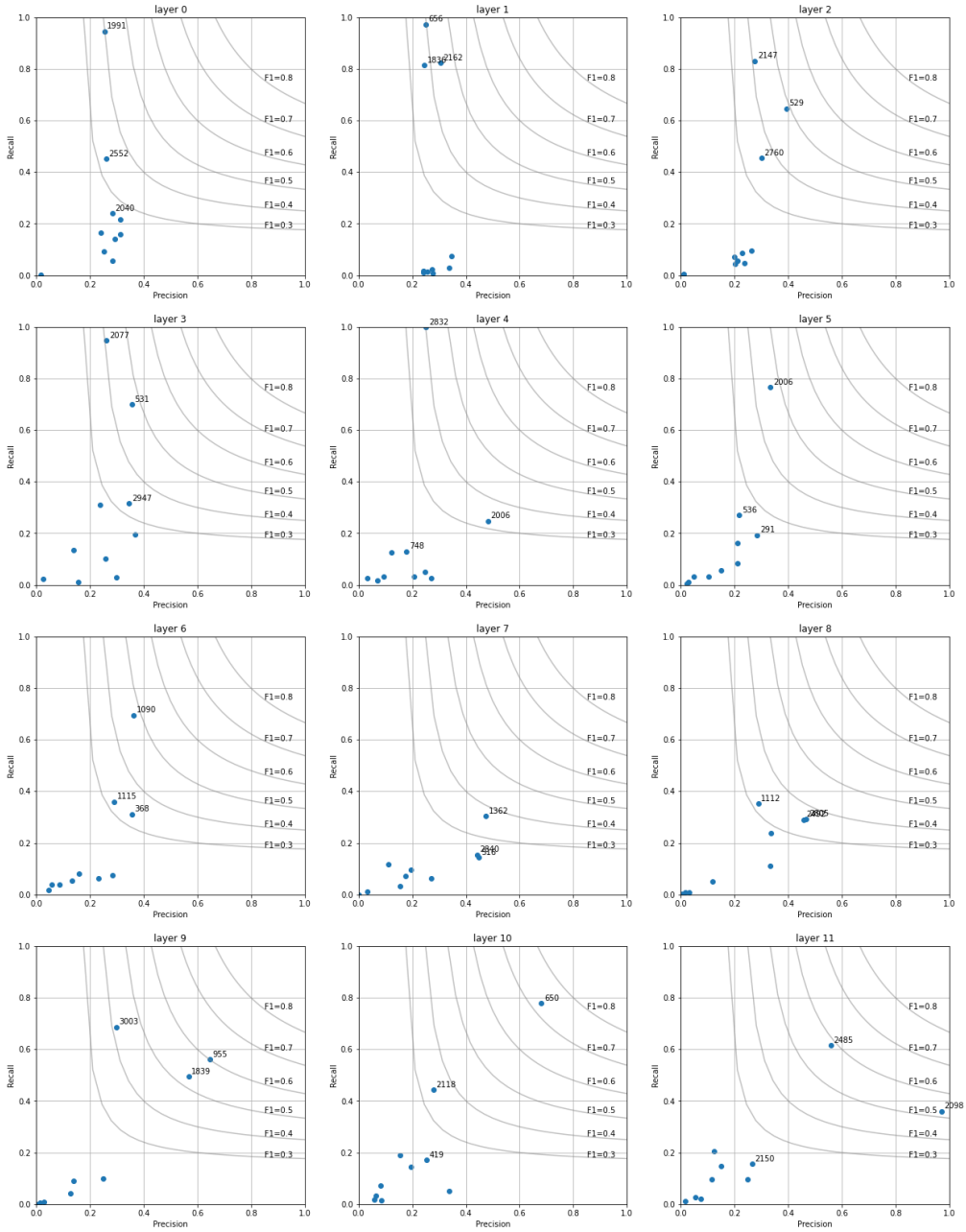

Figure 3: Precision Recall curve for the state-of-the-art (architecture-change) DExp model on the task of predicting masked numbers has an exponent of 3, i.e. it is between 1000 and 10,000. See Section A.5 for details.

B Limitations

Our findings and recommendations may not apply beyond the English language and the Hindu-Arabic Numeral system, which are by no means the only language / number systems in use today. We encourage follow-up work to take other systems into consideration, on the lines of Johnson et al. (2020) and Nefedov (2020).

Our recommended method of tokenizing on the number line is lossy by design. It collapses several numbers into large discrete bins, and is unlikely to be suitable for exact numeracy as is required for, say, math word problems. We note that an ideal number representation should capture both approximate and exact numeracy.