Investigating Numeracy Learning Ability of a Text-to-Text Transfer Model

Kuntal Kumar Pal, Chitta Baral
Department of Computer Science
Arizona State University, Tempe, Arizona, USA,
kkpal@asu.edu, chitta@asu.edu

Abstract

The transformer-based pre-trained language models have been tremendously successful in most of the conventional NLP tasks. But they often struggle in those tasks where numerical understanding is required. Some possible reasons can be the tokenizers and pre-training objectives which are not specifically designed to learn and preserve numeracy. Here we investigate the ability of text-to-text transfer learning model (T5), which has outperformed its predecessors in the conventional NLP tasks, to learn numeracy. We consider four numeracy tasks : numeration, magnitude order prediction, finding minimum and maximum in a series, and sorting. We find that, although T5 models perform reasonably well in the interpolation setting, they struggle considerably in the extrapolation setting across all four tasks.

1 Introduction

Recent advances in transfer learning in NLP have led to the emergence of pre-trained models which show a much stronger contextual representation of words than earlier static word embeddings. They have all performed extremely well in conventional NLP tasks. Yet, they fail to capture a better understanding of numbers. Numbers are integral part of natural language texts which can change the meaning of a sentence. So there is a need for NLP models which can identify numbers represented in any surface forms like words, floats or strings (Numeration), understand its values in various context (Magnitude Order Prediction), compare their values with others (List-MinMax) or able to rearrange a series of numbers based on its values (Sorting).

The transfer-learned models are pre-trained on huge amount of natural language texts with specially designed tasks and tokenizers to create stronger word-embeddings. This causes the numbers embedded in the texts to lose their meaning and inherent rules of numeracy guiding them (Thawani et al., 2021; Nogueira et al., 2021). This is possibly the reason they perform worse in numerical reasoning tasks on numbers absent in training data (Nogueira et al., 2021; Wallace et al., 2019).

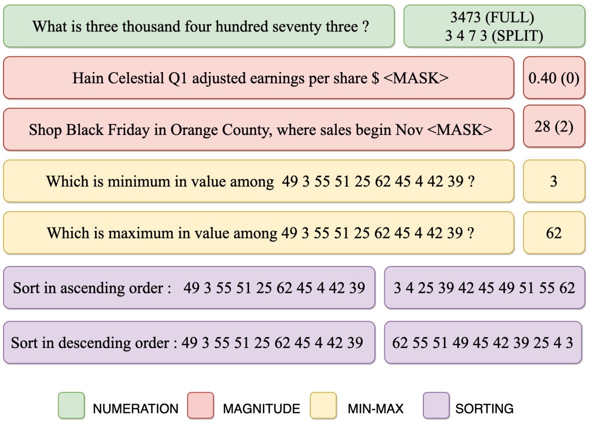

What is three thousand four hundred seventy three ?
3473 (FULL)
3 4 7 3 (SPLIT)
Hain Celestial Q1 adjusted earnings per share $ <MASK>
0.40 (0)
Shop Black Friday in Orange County, where sales begin Nov <MASK>
28 (2)
Which is minimum in value among 49 3 55 51 25 62 45 4 42 39 ?
3
Which is maximum in value among 49 3 55 51 25 62 45 4 42 39 ?
62
Sort in ascending order : 49 3 55 51 25 62 45 4 42 39
3 4 25 39 42 45 49 51 55 62
Sort in descending order : 49 3 55 51 25 62 45 4 42 39
62 55 51 49 45 42 39 25 4 3
NUMERATION
MAGNITUDE
MIN-MAX
SORTING

Figure 1: Examples of Numeracy Tests

In this paper, we test this numeracy learning ability of a text-to-text transfer learning generative model, T5 (Raffel et al., 2020) which has outperformed its predecessors in conventional NLP tasks. The text-to-text format of input and output helps the model to generalize all the NLP tasks as a unified model. We use four numeracy tests both in interpolation (training and testing on same range of data) and extrapolation settings (training on lower and testing on higher range of data) and study how much numeracy skill it can acquire. Figure 1 shows some examples of each of the numeracy tests.

Our contributions in this paper are: (1) Extensive study on three versions of T5 models (small, base, large) on four numeracy tests in interpolation and extrapolation settings. (2) Reporting interesting observations in the behavior of each model version across multiple experimental settings through detailed manual error analysis. The synthetically generated data and codes are publicly available[1] for future numeracy analysis in similar settings.

[1] https://github.com/kuntalkumarpal/T5Numeracy

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 3095–3101
November 7–11, 2021.

2 Numeracy Tests

We perform four essential numeracy tests to explore model's ability to understand numerical values.

Motivation: These four elementary tasks are simple and easy for the models, since they do not need to generate a completely new number in a different numerical range (like in mathematical tests : multiplication, division, exponentiation). Here we evaluate whether the models learn the numeracy tasks or they simply learn bias from the number range seen in training data.

2.1 Numeration

The probability of a number represented in multiple surface forms (word, scientific, float, integer) increases with the increase in the volume of pre-training corpus of the language models. It is impractical for an end-to-end NLP model to semantically parse these numbers accurately and convert into a single representation to retain its value or reason with. This task tests the model's ability to understand word representation of a number and to decode into integer form.

2.2 Magnitude Order Prediction

The task is to identify the order of magnitude of a missing (masked) number which fits the context of a natural language text. This task is important in numerical commonsense reasoning (Lin et al., 2020) and prompt-based methods (Liu et al., 2021). Here, we do not expect the model to predict the exact number that fits the context as this may vary in different domains. Instead, this task tests the model's ability to understand a missing number's context and predict its appropriate range.

2.3 List-MinMax

We test the model's ability to understand numerical values and compare among them. Given a series of n positive numbers, the task is to find the minimum and the maximum number. This is the basis of many question answering and commonsense numerical reasoning dataset like SQuAD (Rajpurkar et al., 2016), DROP (Dua et al., 2019) and NUMBERGAME (Mishra et al., 2020). We simplify the task by generating templates so that the models can concentrate on understanding the task rather than getting confused by the language complexities.

# TRAIN →		4.9K		1.3K		0.9K	
TP	**Model**	**IN**	**EX**	**IN**	**EX**	**IN**	**EX**
FL	T5-SM	45.31	0.08	1.90	0.01	0.33	0.00
	T5-BS	92.16	1.03	66.47	0.45	37.20	0.42
	T5-LG	98.06	1.91	89.49	1.96	79.48	1.58
SP	T5-SM	69.67	39.35	26.89	1.10	0.23	0.01
	T5-BS	99.50	11.31	81.21	22.44	73.61	31.06
	T5-LG	100.00	10.05	99.97	7.35	91.59	12.92

Table 1: **Numeration** EM scores w/ split (SP) and w/o split (FL) representation on 4.9K, 1.3K, 0.9K train-data in Interpolation (IN) and Extrapolation (EX) settings.

2.4 Sorting

In addition to understanding the values of each number in a series, the model will have to rearrange them in the correct order through this task, making it even harder than List-MinMax. Even if a model is successful in the previous test, it is necessary to identify whether it has actually compared among all the numbers in the series. Hence, sorting a list of n numbers in ascending and descending orders ensures that the model compares all the numbers and rearrange them into two different sequences.

3 Experiments

3.1 Experimental Setup:

We use T5-SM (small, 60M parameters), T5-BS (base, 220M), T5-LG (large, 770M) and positive integers for the experiments. The results are average of three random seeds. We perform experiments in two settings: *interpolation* (training and testing on same numerical range) and *extrapolation* (training on lower and testing on higher numerical range). The latter helps us to analyze whether a model has learnt the task, or it has exploited bias in the numerical range of the training data.

3.2 Data Preparation:

Numeration: We create a dataset keeping in mind that at least few examples of all unique words needed to represent each number, are present in the training data (Trask et al., 2018). In Table 1, interpolation samples are from [0,10K) and 99K extrapolation samples are from [10K,1000K). We use *num2words*[2] for generating word-form of each integer. To simulate fewer shot setting, we carefully craft two smaller training sets taking only 20% and 10% data. We show two number representation schemes with split-digits (SP) and without

[2] https://github.com/savoirfairelinux/num2words

		LIST MINIMUM						LIST MAXIMUM					
# ELEMENTS		3		5		10		3		5		10	
Range	**Model**	**IN**	**EX**	**IN**	**EX**	**IN**	**EX**	**IN**	**EX**	**IN**	**EX**	**IN**	**EX**
< 99	T5-SM	90.5	0.6	86.5	0.1	65.9	0.0	80.4	0.5	71.6	0.3	74.7	0.1
	T5-BS	96.2	33.9	99.1	13.0	98.2	2.8	92.3	22.7	96.8	6.0	90.4	1.1
	T5-LG	100.0	22.2	99.4	2.8	100.0	0.5	100.0	29.6	100.0	13.6	100.0	2.0
< 999	T5-SM	72.6	41.8	55.5	22.2	49.9	9.7	65.3	38.4	54.8	17.5	40.0	5.2
	T5-BS	91.5	67.2	92.1	42.6	80.4	27.1	89.1	65.3	90.8	47.2	88.3	25.0
	T5-LG	98.3	70.1	96.1	49.3	87.4	34.7	96.1	61.2	97.8	58.7	95.2	35.3
< 9999	T5-SM	59.1	44.7	43.5	30.4	30.7	17.1	51.2	47.0	36.0	27.0	20.9	11.1
	T5-BS	89.6	68.8	86.9	53.8	85.4	38.1	87.1	58.6	83.1	43.4	81.6	29.9
	T5-LG	97.1	81.3	93.7	71.8	94.0	58.2	96.2	84.9	94.9	76.4	94.9	59.1

Table 2: **List-MinMax** (series length: 3, 5, 10) in three different number ranges evaluated as Interpolation (IN) and Extrapolation (EX) exact-match scores on 1K test data.

Datasets →	AT		MC	
Models ↓	μ**F1**	m**F1**	μ**F1**	m**F1**
LR	62.49	30.81	71.25	60.80
CNN	69.27	35.96	77.17	58.49
GRU	70.92	38.43	78.25	58.08
BiGRU	71.49	39.94	80.16	62.74
CRNN	69.50	36.15	78.00	64.62
CNN-capsule	63.11	29.41	75.89	59.22
GRU-capsule	70.73	33.57	77.36	**64.71**
BiGRU-capsule	71.49	34.18	77.97	64.34
BiLSTM-DICE	75.56	**46.80**	-	-
T5-SM	69.87	31.36	66.11	34.68
T5-BS	78.06	40.04	72.22	47.44
T5-LG	**81.40**	44.64	**80.29**	59.16

Table 3: **Magnitude Order Prediction** for Market Comments (MC) and Article Titles (AT) datasets of numeracy600K in micro-F1 (μF1) and macro-F1 (mF1). Best score is in bold and second-best is underlined.

Train on →	AT		MC	
Models ↓	μ**F1**	m**F1**	μ**F1**	m**F1**
BiGRU	25.59	10.58	31.38	11.08
T5-SM	28.88	12.04	37.35	10.81
T5-BS	35.53	14.48	31.51	12.25
T5-LG	**50.18**	**21.24**	**38.43**	**12.32**

Table 4: **Cross Domain** (Extrapolation) Tests of Order Prediction. Train on MC, test on AT and vice-versa.

split (FL) hypothesizing that for a generative model it would be easier to correctly generate individual digits instead of full integer at once.

Magnitude Order Prediction: For this task we work on Numeracy600K (Chen et al., 2019) dataset. We consider this as a mask prediction task. We train models to find the exact number that fits the mask. Then, we map the predicted numbers into its magnitude order, save the model based on best magnitude order and calculate the evaluation metrics on test data. Since this is a generation task we reject those answers which are not valid floating point numbers. The baseline results in Table 3 are from (Chen et al., 2019; Sundararaman et al., 2020). We also consider extrapolation setting by showing the cross-domain performance (train on market comments and test on article title and vice-versa) in Table 4.

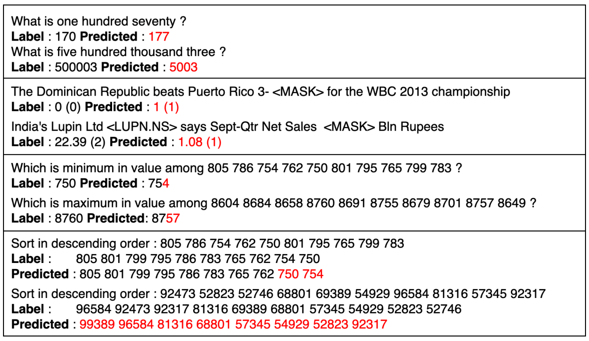
List Min-Max & Sort: We experiment on three different number ranges: [0,100), [0,1K), [0,10K). For interpolation tests, the numbers in the test data are from the same ranges. The extrapolation numbers are from the maximum of respective ranges to 100K. To prevent the model's bias on number lengths, we bring them closer following prior work (Wallace et al., 2019). We extend the experiment on a series of 3, 5 and 10 numbers (for each range) to study how each of the models behave with increasing series length. We consider the same data for sorting experiments as well. The results are in Table 2 for List-MinMax and Table 5 for List-Sort.

4 Results and Error Analysis

Table 1 shows, all versions of T5 benefit when they are trained with split representation. When trained with 4.9K data, T5-SM gains 24% points in interpolation evaluation where T5-LG gains only 2%. None of the models perform well on unseen number data ranges. In fewer shot interpolation settings however, only the T5-LG model maintains its performance beyond 90% which is not surprising because of its large parameter-space. We noticed that the best model could only partially decode numbers having multiple zeros (Figure 2). In the first example, the model predicts an extra seven and in the second (extrapolation), it ignored the key word 'hundred' as it attempts to fit this unseen

		LIST-SORT ASCENDING						LIST-SORT DESCENDING					
# ELEMENTS		3		5		10		3		5		10	
Range	Model	IN	EX	IN	EX	IN	EX	IN	EX	IN	EX	IN	EX
< 99	T5-SM	54.0	12.4	7.6	0.0	0.0	0.0	56.0	12.6	5.9	0.4	0.0	0.0
	T5-BS	80.6	12.2	87.2	0.0	0.4	0.0	84.3	12.9	75.5	0.0	6.2	0.0
	T5-LG	100.0	5.8	99.9	0.0	69.7	0.1	100.0	13.1	96.6	0.1	57.6	0.1
< 999	T5-SM	32.6	15.1	1.4	0.6	0.0	0.0	38.0	22.3	3.4	1.3	0.0	0.0
	T5-BS	74.7	45.7	64.0	8.0	12.5	0.0	73.1	42.0	62.6	9.6	16.8	0.1
	T5-LG	95.1	64.2	91.8	16.8	61.9	1.7	94.7	63.5	92.5	25.7	61.2	1.6
< 9999	T5-SM	23.4	17.1	1.0	0.1	0.0	0.0	30.4	21.2	0.7	0.4	0.0	0.0
	T5-BS	63.1	45.5	51.1	12.7	15.0	0.2	59.8	43.9	51.4	12.4	14.3	0.3
	T5-LG	94.5	76.0	87.4	43.2	74.6	12.6	94.2	76.1	86.1	44.4	75.6	11.9

Table 5: **List-Sort (Ascending & Descending)** on series lengths: 3, 5, 10 in three different integer ranges evaluated as Interpolation (IN) and Extrapolation (EX) exact-match scores on 1K test data.



What is one hundred seventy ?
Label : 170 **Predicted** : 177
What is five hundred thousand three ?
Label : 500003 **Predicted** : 5003

The Dominican Republic beats Puerto Rico 3- <MASK> for the WBC 2013 championship
Label : 0 (0) **Predicted** : 1 (1)
India's Lupin Ltd <LUPN.NS> says Sept-Qtr Net Sales <MASK> Bln Rupees
Label : 22.39 (2) **Predicted** : 1.08 (1)

Which is minimum in value among 805 786 754 762 750 801 795 765 799 783 ?
Label : 750 **Predicted** : 754
Which is maximum in value among 8604 8684 8658 8760 8691 8755 8679 8701 8757 8649 ?
Label : 8760 **Predicted**: 8757

Sort in descending order : 805 786 754 762 750 801 795 765 799 783
Label : 805 801 799 795 786 783 765 762 754 750
Predicted : 805 801 799 795 786 783 765 762 750 754
Sort in descending order : 92473 52823 52746 68801 69389 54929 96584 81316 57345 92317
Label : 96584 92473 92317 81316 69389 68801 57345 54929 52823 52746
Predicted : 99389 96584 81316 68801 57345 54929 52823 92317

Figure 2: Two incorrect predictions for each task.

data into a similar seen number range (4 digits).

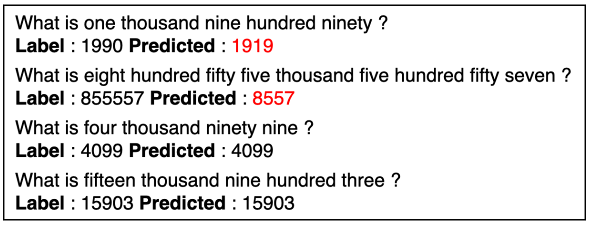
In magnitude order prediction (Table 3), T5-LG's performance improves by 5 μF1 in article title. For extrapolation (Table 4), all T5 versions beats previous estimates (BiGRU) by at most 25%. This shows that T5 can learn robust numeric representations based on contexts. Both the samples in Fig 2 are hard as they need prior explicit knowledge. Yet they are able to predict numbers in similar feasible ranges. This shows that the model is not randomly assigning magnitude but has learnt based on the domain and context. We found that, the best T5 model predicted an order of 1 instead of 2 for market and article data making a maximum error of 39.07% and 33.59% respectively.

Table 2 shows List-MinMax results. Both T5-BS and T5-LG perform over 80% across all ranges and series lengths. T5-SM however, degrades in performance as the range increases along with the list size. As the model learns more variations in numbers, the extrapolation performance increases to a max of 81% (List-Min) and 84.9% (List-Max). But the performance drops as series length increases. The best model predicted second minimum and maximum element in the examples of Fig 2.

From the sorting results (Table 5), we see T5-SM performance drops (18-22% from 2-3 digits, 8-9% from 3-4) as number ranges increase across series length of 3. T5-SM fails to generate a single correct order for a series of 10 elements and achieves less than 10% success in 5-element series across all ranges. This degrading performance can be attributed to its mere 60M parameter space. As the number of parameters keep increasing the models performs consistently across each of 3, 5, 10 elements in series, both for interpolation and extrapolation settings. With the increasing range of training data, the models become more robust to extrapolated numbers across all series lengths with 8-30% change in ascending order and 7-20% change in descending order. Finally, for sorting, we find a variety of incorrect predictions: missing order of one element, omission of one and two elements or repeating a particular element.

Overall, none of the models were able to perform well on extrapolation samples showing the inherent rules of numeracy is difficult for these models to learn. But, it also shows, more variations in numbers (increasing the range) help them perform better in extrapolation setting. The smaller model's limited parameter-space affects its performance in all four tasks whereas larger models are able to pick up some numeracy skills through training. We show more predictions in Figure 3, 4, 5, 6.

Analysis of NT5: We test with the NT5 (Yang et al., 2021) model on all our experiments and compared the results with T5-small. For the Numeration task with the split number representation NT5 performed 73.07 (accuracy), a 4% improvement over T5. The performance however did not improve for the MinMax and Sorting tasks. For 3-element sorting it dropped by 10-20%. In the Magnitude



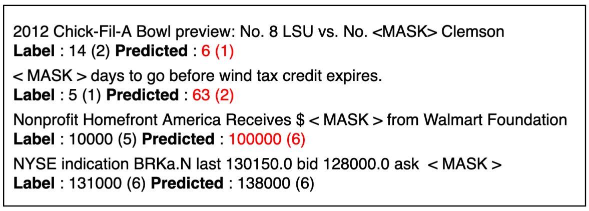
What is one thousand nine hundred ninety ?
Label : 1990 **Predicted** : 1919

What is eight hundred fifty five thousand five hundred fifty seven ?
Label : 855557 **Predicted** : 8557

What is four thousand ninety nine ?
Label : 4099 **Predicted** : 4099

What is fifteen thousand nine hundred three ?
Label : 15903 **Predicted** : 15903

Figure 3: Some predictions for Numeration task.



2012 Chick-Fil-A Bowl preview: No. 8 LSU vs. No. <MASK> Clemson
Label : 14 (2) **Predicted** : 6 (1)

< MASK > days to go before wind tax credit expires.
Label : 5 (1) **Predicted** : 63 (2)

Nonprofit Homefront America Receives $ < MASK > from Walmart Foundation
Label : 10000 (5) **Predicted** : 100000 (6)

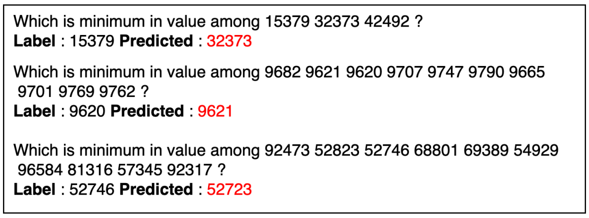
NYSE indication BRKa.N last 130150.0 bid 128000.0 ask < MASK >
Label : 131000 (6) **Predicted** : 138000 (6)

Figure 4: Magnitude Order Prediction Examples.

Order Prediction, we find the cross-domain (extrapolation) μF1 score increases by 5-7% while in-domain decreases by 3-6%. This might be because NT5 has seen more variety of contexts of numbers and can generalize well on this task.

5 Related Works

Numeracy Tests: Multiple numeracy tests have been proposed to evaluate the static word embeddings (Naik et al., 2019) like GloVe, Word2Vec, FastText and contextual embeddings (Wallace et al., 2019) like BERT through probing tasks like numeration, magnitude comparison, addition, list-maximum. Multilingual numeration (Johnson et al., 2020) tests have been performed by probing models like DistilBERT, XLM, and BERT. CNN, BiGRU models have been shown to perform well in magnitude order prediction (Chen et al., 2019) and T5 on addition and subtraction tasks (Nogueira et al., 2021) through training on similar texts. We, however focus on studying how much text-to-text transfer models (T5) can learn across four fundamental numeracy tasks in samples containing both in-domain and out-of-domain numerical ranges.
Specially Designed Models: NALU (Trask et al., 2018), NAU and NMU (Madsen and Johansen, 2020), numBERT (Zhang et al., 2020), GenBERT (Geva et al., 2020), NT5 (Yang et al., 2021) have emerged in the last few years to incorporate arithmetic skills into models through specially designed architecture or fine-tuning tasks which improves the performance in synthetic arithmetic or crowd-sourced numerical reasoning tasks like DROP.

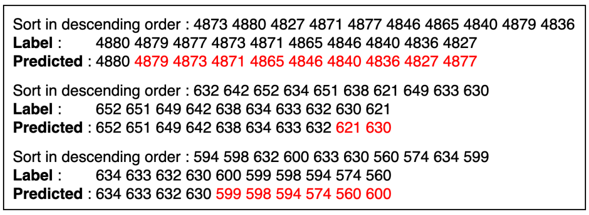


Which is minimum in value among 15379 32373 42492 ?
Label : 15379 **Predicted** : 32373

Which is minimum in value among 9682 9621 9620 9707 9747 9790 9665 9701 9769 9762 ?
Label : 9620 **Predicted** : 9621

Which is minimum in value among 92473 52823 52746 68801 69389 54929 96584 81316 57345 92317 ?
Label : 52746 **Predicted** : 52723

Figure 5: Some predictions for List-MinMax task.



Sort in descending order : 4873 4880 4827 4871 4877 4846 4865 4840 4879 4836
Label : 4880 4879 4877 4873 4871 4865 4846 4840 4836 4827
Predicted : 4880 4879 4873 4871 4865 4846 4840 4836 4827 4877

Sort in descending order : 632 642 652 634 651 638 621 649 633 630
Label : 652 651 649 642 638 634 633 632 630 621
Predicted : 652 651 649 642 638 634 633 632 621 630

Sort in descending order : 594 598 632 600 633 630 560 574 634 599
Label : 634 633 632 630 600 599 598 594 574 560
Predicted : 634 633 632 630 599 598 594 574 560 600

Figure 6: Some predictions for List-Sort task.

Numerical Embeddings: There are limited prior works in numeracy aware embeddings which show good performance in extrapolation setting. One approach (Jiang et al., 2019) represents numerals as a weighted average of prototype numeral embeddings obtained using either self organizing map or Gaussian Mixture models. DICE (Sundararaman et al., 2020) is a deterministic numeral embedding approach, independent of corpus, which preserves the relative magnitude between two numerals and their embeddings.

6 Conclusion & Future Works

We show that text-to-text models are able to learn numeracy quite well in an interpolation setting. Our extensive experiments show that T5 models struggle to learn with numbers outside training data ranges. We believe that, to make further progress in transfer learning, models need to achieve such elementary numeracy skills and this gap between interpolation and extrapolation performance needs to be reduced. We are of the opinion that, adding more data would not bridge this gap since domain of numbers is open. However, special pre-training objectives for digits rather than whole numbers can be designed to teach the inherent numeracy to models. In future, we intend to explore these objectives centered around preserving numeracy rules in transfer-learned models to generalize between in-domain and out-of-domain numbers.

Acknowledgement

The authors acknowledge support from DARPA grant number FA875019C0003 for this project.

Ethical Considerations

In this paper, we analyze performance of three publicly available T5 models on four numeracy tasks. For Magnitude Order Prediction task we use publicly available dataset, Numeracy600K. We synthetically create the data for rest of the tasks.

References

Chung-Chi Chen, Hen-Hsen Huang, Hiroya Takamura, and Hsin-Hsi Chen. 2019. Numeracy-600k: learning numeracy for detecting exaggerated information in market comments. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6307–6313.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*.

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.

Chengyue Jiang, Zhonglin Nian, Kaihao Guo, Shanbo Chu, Yinggong Zhao, Libin Shen, and Kewei Tu. 2019. Learning numeral embeddings. *arXiv preprint arXiv:2001.00003*.

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.

Bill Yuchen Lin, Seyeon Lee, Rahul Khanna, and Xiang Ren. 2020. Birds have four legs?! numersense: Probing numerical commonsense knowledge of pretrained language models. In *Proceedings of EMNLP*. To appear.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pretrain, prompt, and predict: A systematic survey of prompting methods in natural language processing.

Andreas Madsen and Alexander Rosenberg Johansen. 2020. Neural arithmetic units. *arXiv preprint arXiv:2001.05016*.

Swaroop Mishra, Arindam Mitra, Neeraj Varshney, Bhavdeep Sachdeva, and Chitta Baral. 2020. Towards question format independent numerical reasoning: A set of prerequisite tasks. *arXiv preprint arXiv:2005.08516*.

Aakanksha Naik, Abhilasha Ravichander, Carolyn Rose, and Eduard Hovy. 2019. Exploring numeracy in word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3374–3380.

Rodrigo Nogueira, Zhiying Jiang, and Jimmy Li. 2021. Investigating the limitations of the transformers with simple arithmetic tasks. *arXiv preprint arXiv:2102.13019*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Dhanasekar Sundararaman, Shijing Si, Vivek Subramanian, Guoyin Wang, Devamanyu Hazarika, and Lawrence Carin. 2020. Methods for numeracy-preserving word embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4742–4753.

Avijit Thawani, Jay Pujara, Pedro A Szekely, and Filip Ilievski. 2021. Representing numbers in nlp: a survey and a vision. *arXiv preprint arXiv:2103.13136*.

Andrew Trask, Felix Hill, Scott E Reed, Jack Rae, Chris Dyer, and Phil Blunsom. 2018. Neural arithmetic logic units. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. Do NLP models know numbers? probing numeracy in embeddings. 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 5307–5315, Hong Kong, China. Association for Computational Linguistics.

Peng-Jian Yang, Ying Ting Chen, Yuechan Chen, and Daniel Cer. 2021. Nt5?! training t5 to perform numerical reasoning. *arXiv preprint arXiv:2104.07307*.

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.

A Appendix

A.1 Data Statistics Experimental Setup

Numeration: We have 4906, 2097, 2997 in train, dev and test respectively. We make sure that all numbers within 10K are present in any of train, dev or test. For extrapolation we select 1K integers randomly from every 10K range from [10K,1000K) making it a total of 99K.

Magnitude Order Prediction: For this data we consider 450K, 50K and 100K samples for train, dev and test data respectively from each of market comments and article titles data.

List-Sort: We consider both the task of arranging in ascending and descending orders since if a series is already sorted in ascending order the model can directly predict by copying it from the given input.

What is nine thousand one hundred sixty two ?
Label : 9162 **Predicted** : 9172

What is eight hundred twenty thousand six ?
Label : 820006 **Predicted** : 826

What is one thousand nine hundred sixty ?
Label : 1960 **Predicted** : 1959

What is three hundred thousand four hundred fifteen ?
Label : 300415 **Predicted** : 3415

Figure 7: More predictions for Numeration task.

A.2 Hyperparameters

For all the experiments we use maximum sequence length of 128 and 256 for question context. The maximum sequence length of the answers is kept as [5, 10, 20, 25] for different tasks. We ran for 20 epochs and save a model based on validation EM performance. Our training and validation batch size varies between [2, 4, 8, 16, 32] based on the experiment. We work on 4 Tesla V100 GPUs. We use AdamW optimizer and StepLR scheduler with step size of 2, learning rate of 5e-5 and gamma of 0.1.

2012 Chick-Fil-A Bowl preview: No. 8 LSU vs. No. <MASK> Clemson
Label : 14 (2) **Predicted** : 6 (1)

< MASK > days to go before wind tax credit expires.
Label : 5 (1) **Predicted** : 63 (2)

Nonprofit Homefront America Receives $ < MASK > from Walmart Foundation
Label : 10000 (5) **Predicted** : 100000 (6)

Gun ban advocates must decide if they're willing--and able--to kill < MASK >
Label : 50000000 (7) **Predicted** : 10000 (5)

NYSE indication BRKa.N last 130150.0 bid 128000.0 ask < MASK >
Label : 131000 (6) **Predicted** : 138000 (6)

NCR CORP - Updating its full year 2016 guidance for non-Gaap diluted eps to $2.85 from its previous guidance of $2.72 to $ < MASK >.
Label : 2.82 (1) **Predicted** : 2.85 (1)

Figure 8: More Magnitude Order Prediction Examples.

Which is minimum in value among 162 56 52 ?
Label : 52 **Predicted** : 56

Which is minimum in value among 630 628 627 ?
Label : 627 **Predicted** : 630

Which is minimum in value among 15379 32373 42492 ?
Label : 15379 **Predicted** : 32373

Which is minimum in value among 9682 9621 9620 9707 9747 9790 9665 9701 9769 9762 ?
Label : 9620 **Predicted** : 9621

Which is minimum in value among 92473 52823 52746 68801 69389 54929 96584 81316 57345 92317 ?
Label : 52746 **Predicted** : 52723

Figure 9: More predictions for List-MinMax task.

Sort in descending order : 4873 4880 4827 4871 4877 4846 4865 4840 4879 4836
Label : 4880 4879 4877 4873 4871 4865 4846 4840 4836 4827
Predicted : 4880 4879 4873 4871 4865 4846 4840 4836 4827 4877

Sort in descending order : 632 642 652 634 651 638 621 649 633 630
Label : 652 651 649 642 638 634 633 632 630 621
Predicted : 652 651 649 642 638 634 633 632 621 630

Sort in descending order : 594 598 632 600 633 630 560 574 634 599
Label : 634 633 632 630 600 599 598 594 574 560
Predicted : 634 633 632 630 599 598 594 574 560 600

Sort in descending order : 600 902 20 120 1237 1492 173 291 30 28
Label : 1492 1237 902 600 291 173 120 30 28 20
Predicted : 1492 1237 902 600 291 291 173 120 30 20

Sort in descending order : 40049 22125 38721 22513 44180 43313 19923 17563 18365 38121
Label : 44180 43313 40049 38721 38121 22513 22125 19923 18365 17563
Predicted : 44180 40049 43313 38121 38721 22513 22125 19923 18365 17563

Figure 10: More predictions for List-Sort task.

A.3 Results and Error analysis

Magnitude Order Prediction: We also experimented with zero-shot magnitude order predictions. We found 553 and 8783 exact-matches out of 100K test data using T5-large which shows that the performance is very poor without proper fine-tuning. We show some more predictions of the best performing T5 model in Figure 7, 8, 9, 10.