Introduction

In this project, we reproduce the paper *Investigating the Limitations of Transformers with Simple Arithmetic Tasks* by Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. The work addresses the inability of standard pretrained transformer models to perform basic arithmetic such as addition. Motivated by the human capacity for abstraction and seeking to see if existing transformer models could attain the same ability, the authors of the original paper inject explicit digit-position representations into the model's input.

The authors fine-tune T5 models of various sizes on generated addition and subtraction datasets using six distinct input representations and show that positional markers enable near-perfect accuracy on numbers up to 60 digits. Finally, the paper demonstrates that, despite these representations, transformers fail to generalize to longer or even shorter sequences than those seen during training, revealing a fundamental limitation in their extrapolation capabilities.

Results

We aimed to reproduce Figure 1 from the original paper, which presents the test accuracies of various number format types on adding two numbers with between 2 to 30 digits. This result is significant because it highlights how the input representations affect the performance of transformers.

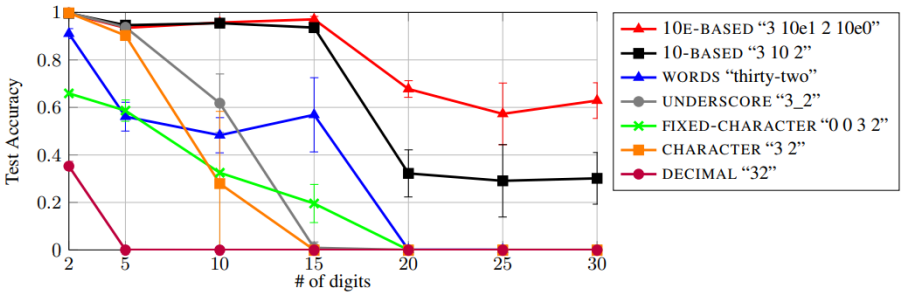


Figure 1: Accuracy of different number representations on the addition task.

As shown above, the input representation of the arithmetic can greatly boost the performance of the transformer, achieving near perfect accuracies in some scenarios. We chose to reproduce this due to its central role in demonstrating the paper's key claim: that transformers, when fine-tuned, can learn arithmetic operations to varying degrees depending on the input format.

Methodology

We generated our dataset through a process called balanced sampling. Given a maximum number of digits D (D = 30 in our implementation), sample a number d from [2, D] and further sample a number between [10^{d-1}, 10^{d-1} - 1]. Repeat this twice to get two numbers to add, and substitute them into the format "What is [number1] plus [number2]?". Using this sampling method and question format, a dataset of 10000 was created along with a validation set of 1000. Each dataset was then formatted to one of the 7 different input format types

Orthography	Example	Notes
DECIMAL	832	default representation
CHARACTER	8 3 2	ensures consistent tokenization
FIXED-CHARACTER	0 8 3 2	ensures consistent positions (e.g., max. 4 digits)
UNDERSCORE	8_3_2	underscores provide hints on digit significance
WORDS	eight hundred thirty-two	leverages pretraining
10-BASED	8 100 3 10 2	easy to determine digit significance
10E-BASED	8 10e2 3 10e1 2 10e0	more compact encoding of above

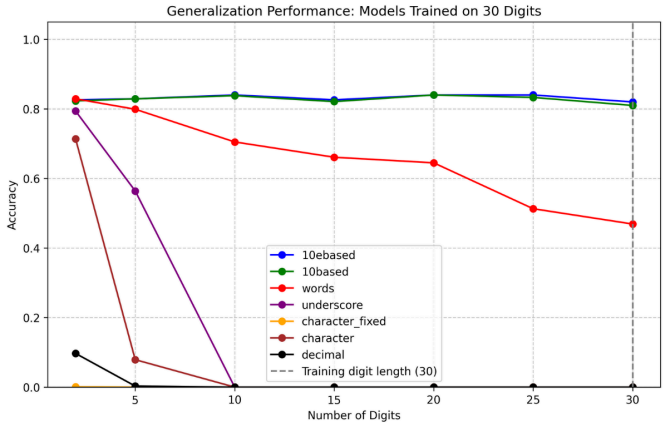
Table 1: Different ways of representing numbers explored in this work.

The 7 input formats are shown above in Table 1 (taken from the paper). They vary from no change to the number (decimal) to more tokenized versions and those with digit positions.

Each formatted dataset was used to finetune, validate, and test a T5-base (220M parameter) transformer model. Finetuning was done with an AdamW optimizer with a learning rate of 3e-4 to 4e-4, weight decay of 5e-5, and gradient clipping at 1.0. Batch size was 64 over 25 epochs. The model was validated every two epochs and the best accuracy checkpoint was used.

Compared to the original paper, we made some minor changes to speed training and reduce memory usage. Batch size was reduced from 128, but gradient accumulation steps was set to 4 to simulate larger batches. Additionally, more training data was used at the expense of less epochs, but the paper mentioned that they performed similar experiments and found that this configuration didn't affect overall performance much.

Results



Our results (shown above) align with the paper's: that 10e-based and 10-based formatting are the most effective with basic arithmetic tasks in transformers. Other formats did well with 2 digit numbers, but their performance plummeted with larger numbers, just like those in the paper.

While the reproduced accuracies are slightly lower, they still achieved decent accuracies around 80% and reflected the advantages of each format. We believe that most of the discrepancies come from using smaller batch sizes (due to memory constraints) and minor changes to the AdamW. Regardless, our results reflect the paper and show that input formatting can be a key element of using transformers on arithmetic, as they can drastically change performance.

Reflections

Transformers learn most accurately when introducing position tokens (e.g. "3 10e1 2") with 10e based words having the highest test accuracy. This result is exhibited even with varying amounts of data, epochs, and batch sizes, highlighting the robustness of this technique.

In the future, after validating the paper's findings, we want to extend to modern transformers, i.e. Llama3, and to chart progress (or persisting gaps) in LLM reasoning. In addition, expanding the range of values to decimal numbers, (e.g "10.49 + 7.77"), quantity of inputs, different operations (multiplication, modular arithmetic, and nested parentheses), on both fine-tuned T5-base and newer models could reveal limits of reasoning.

References

Nogueira, R., Jiang, Z., & Lin, J. (2021). Investigating the limitations of transformers with simple arithmetic tasks. *arXiv preprint arXiv:2102.13019*.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... & Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, *21*(140), 1-67.