Carrie Magee
Intro to Natural Language Processing
12/11/2023

Individual Report

Introduction

For our final project, our group drew inspiration from the research paper titled *"Can NLP Models Really Solve Math Word Problems?* (Patel et al., 2022)" This paper aimed to address the need for an effective evaluation dataset for math word problem (MWP) models, highlighting concerns with prior models on commonly used datasets like *Math Word Problem Repository* (MWAPS) and *Academia Sinica Diverse MWP Dataset* (ASDiv-a). These datasets, typically featuring simplistic (elementary level) standardized math word problems, initially showcased high accuracy. However, researchers later exposed the models' limitations by revealing their proficiency even after removing the question part of the word problem (e.g., "How many apples does Jack have now?").

Aligned with the paper's approach, we set out to develop a model capable of translating MWPs into numeric equations, specifically focusing on more complex scenarios. The SVAMP dataset, created by Microsoft researchers to address limitations in other datasets, was designed with more intricate question formatting. The goal was to test if models genuinely learned to solve MWPs or if they relied on cues unrelated to the essential mathematical components.

To tackle this challenge, each team member independently created models using different types of transformers. I opted for a GPT-2 transformer due to its adaptability in various NLP tasks, including text summarization and generation. Paul worked on a BERT and T-5 model, while Akshay successfully built his own LSTM model from scratch. After several weeks of individual model development and contemplating the possibility of combining all three in the final project, we decided to vote on the best-performing model.

During this evaluation, we assessed the models based on sample problems and their ability to generate accurate numeric equations. The T-5 model emerged as the most successful, prompting us to shift our focus to collectively fine-tuning that particular model.

Individual Contribution I

In my individual contributions, the GPT-2, while not ultimately selected for the final model, played a pivotal role in establishing the foundation for the T-5 model. Constructing the GPT-2 model involved addressing challenges, notably memory constraints during training. To overcome this hurdle, I implemented a data loader and reduced the batch size to 8, effectively resolving the CUDA memory issues.

Despite encountering initial setbacks, the GPT-2 model exhibited an impressively low loss rate, measuring less than 0. However, when subjected to sample math word problems, the model's performance fell short of accurate. It either reproduced the same word problem verbatim or truncated the final question. In response, I explored the use of Optuna to fine-tune hyperparameters, ultimately identifying the optimal settings as learning rate 1e-3 and batch size 16. Subsequently, attempts to improve performance included experimenting with an increased sample size. However, this adjustment proved challenging, as it led to significantly slower model training and, in some instances, exhausted memory resources. Despite these challenges, my endeavor with the GPT-3 model not only contributed valuable insights but also informed subsequent decisions in the development of our final T-5 model.

Individual Contribution II

Link to Individual Contribution Code: https://github.com/darkivist/Final-Project-Group1/tree/Carrie/Code

At the project's inception, I assumed an administrative role, overseeing the planning of meetings and task delegation—a responsibility I maintained throughout the project's duration. In the early stages, I played a crucial role in guiding the team's orientation by providing example code for LSTM models, a consideration predating our exploration of transformers. Additionally, I spearheaded extensive data preprocessing efforts, deciphering the nuances of various datasets and ensuring they were appropriately formatted for our models. Managing multiple variations of the same dataset posed a significant challenge, particularly in terms of integrating masks for numbers used within word problems. To address this, I developed code to replace the masks with actual numbers.

Transitioning to the creation of the GPT model, my initial step involved thorough research to identify the most suitable transformer for our specific problem. Opting for GPT-2, I leveraged the strength and flexibility it offered for NLP tasks. The original model was crafted

using the GPTLMHead model, coupled with the distilgpt2 pre-trained transformer and tokenizer from Hugging Face, showcasing promising potential. However, as I embarked on the model training phase, I encountered CUDA memory issues, prompting a redesign of the code to incorporate a custom dataset and DataLoader from PyTorch. This adjustment proved instrumental in overcoming the memory challenges, allowing for a more seamless training process. Ultimately, the GPT model did not work because it did not align with what was needed for the project which was text summarization as opposed to text generation.We opted for T-5 as our final model because of the way it is trained to generate text based on it's input text which is important for math word problems. It also may have been due to the training data or the lack thereof at first before locating the augmented version of the MAWPS dataset.

Once our team decided to go with a T-5 transformer for translating math word problems into equations. I decided to help with researching what types of metrics would best exemplify how well our model does at its given task. BLEU score seemed like a good option since it's meant to judge how well a model generates text but the score was very low. Considering the low BLEU score, I sought out other metrics which included ROUGE score. The ROUGE score is good for text summarization because it is able to measure the quality of generated summaries compared to the true summary output. I included the code for the ROUGE score on the testing script that was drafted. In addition to the test script, I also created a small test csv for the testing script from the SVAMP. My last individual contribution included outlining the group final paper and writing the introduction, description of dataset, and part of the NLP model explanation.

Results

As mentioned in the prior paragraph, my group and I did extensive research on the best metrics for our type of problem. Obviously, since we were working with numeric equations it seemed appropriate to use accuracy as a metric of evaluation. For the Flan-T5-Base model used for our equation output, the accuracy came out to be about 24%. Similarly, when we attempted to use the T5-Small model for our equation output, the accuracy came out to be around 9%. Upon further analysis of the accuracy, it was found that the Flan-T5-Base model outperformed the T5-Small model across each type of operation (Figure 1). This may be due to the fact that the Flan-T5-Base was created with greater parameters and has strong adaptability for more complex problems. In relation, the accuracy of our models across the various operations revealed an interesting trend. Figure 2 reveals that math word problems involving division tended to be better

predicted by our models for both the equation and numeric answer. That trend may be due to certain words like “sharing”, or “between” are more commonly used when talking about division and thus, more often than not to be picked up by the model.

Figure 1.

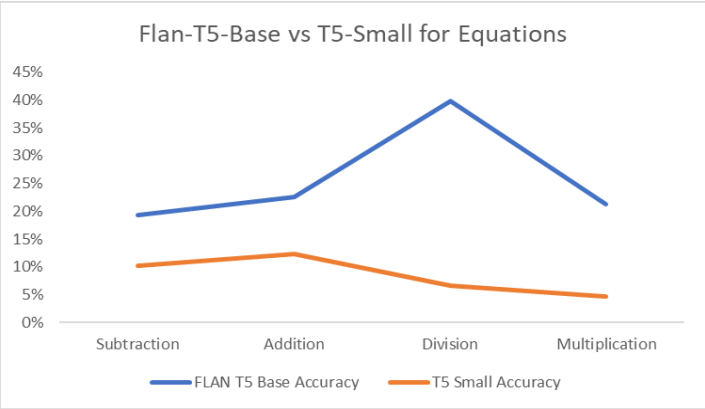
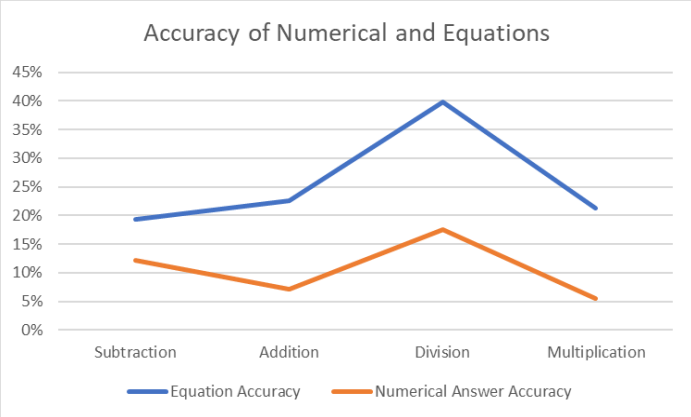
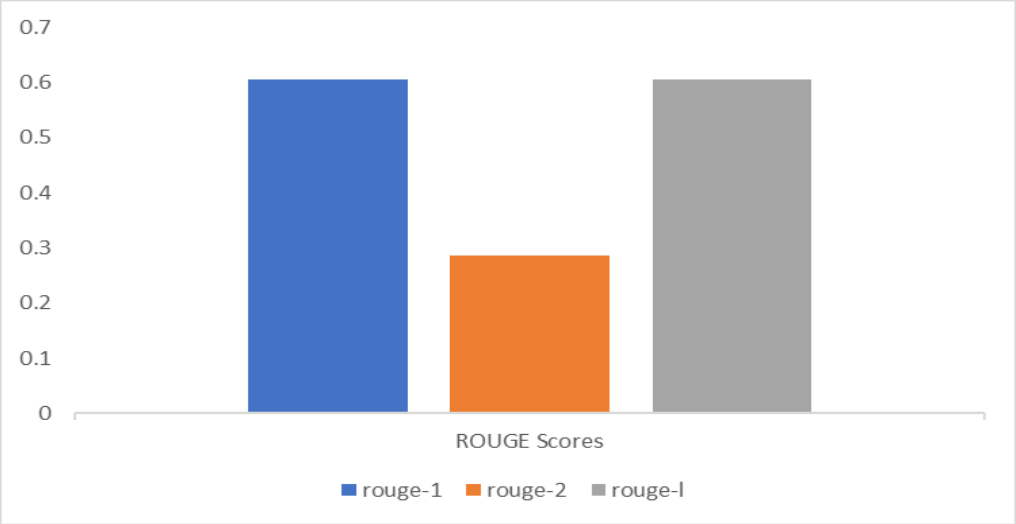


Figure 2.



In addition to accuracy, ROUGE score also proved to be a helpful metric of evaluation for our model as mentioned above. ROUGE scores are able to quantify the linguistic performance of Large Language Models (LLM) such as our math word problem language model. There are three types of ROUGE scores that we were able to generate for our Flan-T5-Base model (Figure 3). For ROUGE-1, the measure of overlap of unigrams between the generated text and target text, came out to 0.605 for our model. Considering this high score, it can be concluded that there is a large overlap between generated equations and the reference equations at the level of individual words. The ROUGE-2 score, which measures the overlap of bigrams between generated text and target text, was 0.287. This score is quite low indicating that our model is not great at generating overlapping pairs of words for the reference equations. Lastly, the ROUGE-L score came out to be 0.605, meaning that the model does pretty good at generating text with overlapping long sequences. This metric in particular is a good measure of content overlap between the reference and generated equations. Overall, the results of the model in terms of accuracy and ROUGE are satisfactory. However, it is reassuring that the accuracy of the models used in the research paper that inspired our project also only had an accuracy of about 30%.

Figure 3.



Conclusion:

In summary, this project shows the robustness and efficacy of transformers for translating math word problems into numeric equations. After a series of trial and error for discovering the right model, we successfully employed a Flan-T5-Base model for accurate equation outputs and a T5-Small model for accurate numeric answers. The Flan-T5-Base model outperformed the T5-Small across the various ROUGE scores as well as having a larger accuracy. While we take pride in our achievements, there is still room for improvement. It would be beneficial to explore other datasets and transformer types. It may be helpful to train the model on more complex problems since we used elementary level math word problems. It would also be important to explore our options for hyperparameter tuning and optimization. With that being said, our primary focus lies in enhancing accuracy, aiming to perfect our math word problem translator to ensure precise generation of both the equation and the correct answer.

Code Percentage

About 56% of the code came from the internet.

Work Cited

Doe, P. (2021). Rouge Your NLP Results. Medium. *https://medium.com/@priyankads/rouge-your-nlp-results-b2feba61053a*

Koncel-Kedziorski, R., Roy, S., Amini, A., Kushman, N., & Hajishirzi, H. (2016, June). MAWPS: A math word problem repository. In Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: human language technologies (pp. 1152-1157).

Hugging Face. (2023). T5-small: Text-To-Text Transfer Transformer. Hugging Face. *https://huggingface.co/t5-small*

Hugging Face. (2023). T5-Flan-Base: Text-To-Text Transfer Transformer. Hugging Face. *https://huggingface.co/google/flan-t5-base*

Jacob, John. (2023). What is FLAN-T5? Exemplary AI Blog. *https://exemplary.ai/blog/flan-t5*

Patel, A. (2022). SVAMP: Structural Variant Annotation, Mapping, and Prediction. GitHub. *https://github.com/arkilpatel/SVAMP*

Patel, A., Bhattamishra, S., & Goyal, N. (2021). Are NLP models really able to solve simple math word problems?. arXiv preprint arXiv:2103.07191.

MathBot. (2023). MAWPS_Augmented.pkl. GitHub. *https://github.com/Starscream-11813/MathBot/blob/main/MAWPS_Augmented.pkl*

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. J. Mach. Learn. Res., 21(140), 1-67.