CMPSC 497 Final Project

Pranav Suby

April 25, 2025

1 Dataset Curation

For this project, I initially aimed to collect research papers from various different fields and extract the research problem and approach from those papers. However, after numerous attempts, the output from the extraction would always yield many errors, since I did not have an effective way to find out what the approach is besides searching through a few key phrases and hoping that they were an actual approach. Most of those ended up not being an actual approach, so I decided to switch to trying to find a dataset online, which lead me to https://www.kaggle.com/datasets/anshulmehtakaggl/200000-abstracts-for-seq-sentence-classification/data.
In this dataset, I used the 200k abstracts folder from the repository. The labeling was a little bit misleading, so for every line that was labeled objective, I combined them all and made it into the problem statement. If there was no line labeled objective, I used the background line, since both of them ended up being very similar when I looked at random samples in the data. For the approach, I combined all the lines with the method label. If an abstract had no method label or no background/objective label, it would be skipped. From this, I had 190,653 abstracts for the training set, and 2,499 abstracts for the testing set. The datasets looked like the following:

Problem Statement	Approach Statement
problem 1	approach 1
⋮	⋮

This data is all stored in the data folder, the raw data in the raw folder, and processed in the processed folder. This folder could not be added to the github due to the files being too big. After the data was formatted properly, I began the LLM training.

2 LLM Selection and Training

Due to lack of computing power, I was not able to try out many advanced LLMs. Also, since I'm using an M2 mac, I wasn't able to do anything with GPU, so everything was CPU trained and thus a lot slower than it could be. I first tested distilGPT2, which runs based on GPT2. I had the max step size be 5000, since it was an older model, however that did take a long time to run. With distilGPT2, it tended to repeat words often, so I added a no repeat ngram cap of 3. After adding this, the model produced more coherent results, but the results did not have a great connection to the actual research problem. I tried with a max step size of 3000 to but with no real difference in performance.
This lead me to switch to EleutherAI gpt neo, with 125 million parameters. Although this was more taxing on my computer, it was able to successfully complete. With this, I played around with both 3000 and 1000 as the max step size, and found no noticeable difference.
To adapt GPT-Neo-125M to our task, I used LoRA adapters, which freeze the base model's 125 million weights and introduce low-rank update matrices in key attention and output projection layers. LoRA was configured with rank $r = 8$, $\alpha = 32$, and dropout $= 0.05$, targeting the q_proj, k_proj, v_proj, and out_proj

modules. Training used a learning rate of 2×10^{-5}, a per-device batch size of 2 with gradient accumulation over four steps (effective batch = 8), and a total of 3,000 optimizer steps. All training ran in FP32 (no mixed precision) under PyTorch MPS, with gradient checkpointing disabled (the model fit comfortably) and a fixed seed of 42 to ensure reproducibility.

3 Evaluation Metrics

I used three different evaluation metrics for this project: Bleu, Rouge, and Bert.

Bleu	
Score	0.021

Table 1: Bleu Score

The Bleu score was only 2.1%, which is low, however, since Bleu is most effective at caputuring n-gram overlap, and the LLM just aims to create a solution to the research problem, this low value for Bleu is okay.

Rouge	
ROUGE-1	0.211
ROUGE-2	0.034
ROUGE-L	0.124

Table 2: Rouge Scores

The ROUGE-1 score shows that the model uses about 21% of the same 1-grams, however the low ROUGE-2 score means that the model ends up struggling with making domain relevant sentences fragments, and instead just has some of the key words individually. The ROUGE-L score shows that 12% of the tokens appear in the same order as the test case approaches, which is to be expected.

Bert	
Precision	0.218
Recall	0.280
F1	0.246

Table 3: Bert Scores

The 24.6% F1 score shows that the model has some contextual similarity; about a quarter. Since recall is higher than precision, and both of them are above 20%, the model covers a lot of the concepts used in the training data, but also uses new content.
Overall, this means that the model uses domain related vocabulary, but struggles with more in depth reasoning. Some of this could be attributed to the model creating a novel approach, however it would reason to believe that the ROUGE-2 scores would be higher if this were the case.

4 Thoughts

Overall, given the limited computing power and model choice, I believe the model performed well. I think future work should use a larger model, maybe Llama 3. I also think that a training set with multiple different fields might be helpful, which could be identified with a different model extracting the problem and approach from the paper. That way, it would also be easier to identify if the model is using relevant words, because irrelevant words would be of a different subject matter than the problem statement. Since

all of the data was medical jargon, I did not fully understand everything. I learned a lot about how exactly to train an LLM, and some of the pitfalls that can happen with training. I also learned how much harder it is to run these models without GPU power. I also learned about how to interepret the evaluation metrics, which will help me when I make my own personal LLMs.