Cramming Language Models

Angie Fathalla(af4292) and Karan Vora(kv2154)

Executive Summary

How much accuracy can we get receive when we have limited resources within a small window of time? We proceeded to use a Bert_Base Model trained on limited resources. We manipulated previously created codes and tried to improve upon their work by fine tuning parameters and observing the results.

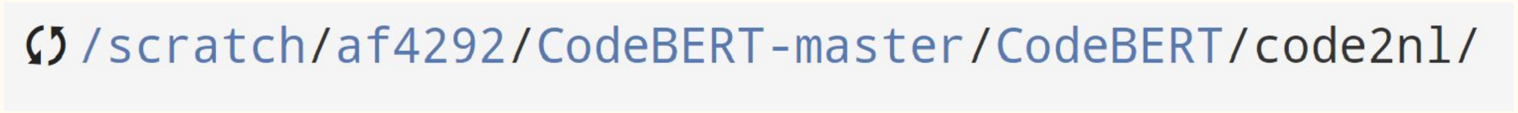
Executive Summary Continued

We collected as much information we could find on transformer architecture in this resource constricted deployment method.

The procedure was to upload and unzip CodeBert_Master:

CodeBERT-master

Within the directory:



```
/scratch/af4292/CodeBERT-master/CodeBERT/code2nl/
```

Are where the files are located in which we made the changes.

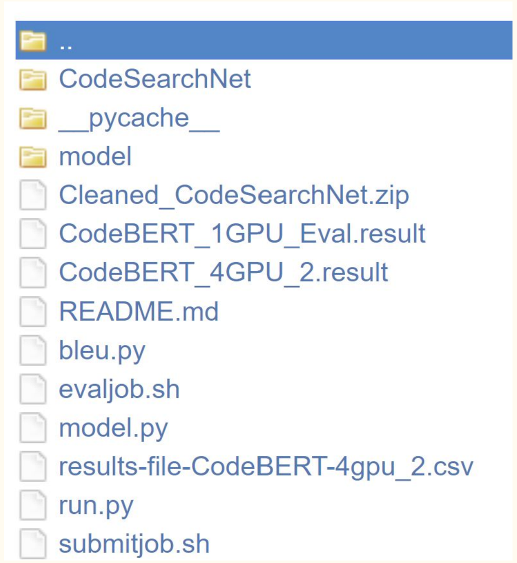
Executive Summary Continued

We made changes to the parameters within model.py file.

We then submitted submitjob.sh which took roughly 10 hours to train the model.

Afterwards we submitted evaljob.sh which took 2 hours.

We then recorded our results with the new parameters and try to explore whether we could make improvements to the accuracy.



```
..
CodeSearchNet
__pycache__
model
Cleaned_CodeSearchNet.zip
CodeBERT_1GPU_Eval.result
CodeBERT_4GPU_2.result
README.md
bleu.py
evaljob.sh
model.py
results-file-CodeBERT-4gpu_2.csv
run.py
submitjob.sh
```

Technical Challenges

Technical Challenges

1. We have limited access to resources.

2. We struggled to enable the codes to work on our machines.

3. Find what parts of the architecture to use to our advantage.

Solution Approach

Solution Approach

We found that...

Training LLMs is really difficult. Requires large amount of clean data for minor improvements.

Fine Tuning is even more difficult as the final performance might be worse.

The parameters that had already used in the original code had given the highest accuracy and our changes had considered: increasing the number of epochs, the batch size and the learning rate.

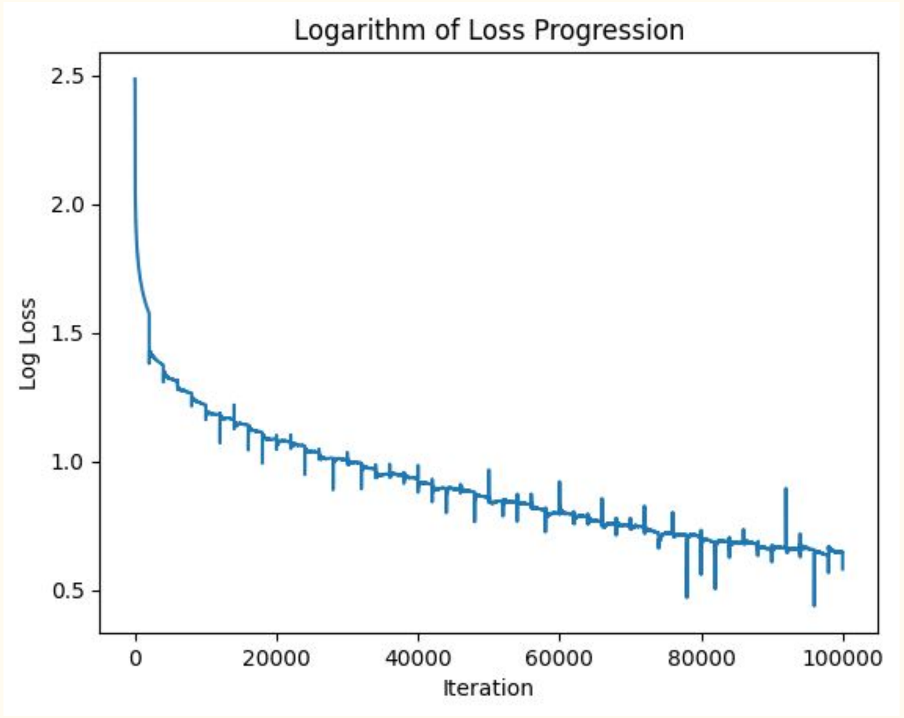
Main Results

Main Results

We used BLEU 4-Gram benchmark on our model and we got following scores

Programming Language	Score
Ruby	12.16
JavaScript	14.90
Go	17.99
Python	18.97
Java	17.15
PHP	25.02

Results



This is the logarithmic-plot of Loss Progression while training.

Observations/Conclusion

Observation/Conclusion

The results we received were because...

We had not considered that a larger batch size can lead to more memory consumption, and if too many epochs were given, both can potentially be unable to generalize well and lead to overfitting.When the learning rate is too high, the model can fail to converge, which will result in poor performance.

To get modest Improvements in final results, We need to provide with more data thus increasing the training time.

There is a hypothetical wall of training LLMs and even after providing with larger amount of data, smaller LLMs won't be able to improve much or will get even worse.

GitHub Link:
https://github.com/karanvora2599/ECE-GY-9143-HPML-Final-Project-Spring-2023