Neural Network
Assignment 3
Protein Class prediction
Joon Jung

First, I wish I took screenshots as I was doing the assignment, but since there was no big visualization, I didn't really think to do it. When I went back to run the codes again, I either got Disk Quota Exceeded errors or CUDA memory ran out errors, which was weird because I ran them fine before, but I think I ran out of memory on home directory that I wasn't able to run the previous models results again so I can screenshot. But I do remember clearly what their accuracy numbers were. I luckily did take a screenshot of my latest model code.

I started off with base model code. The model that was provided first to us was "Rostlab/prot_bert". After I got it to work, I got really bad score and first thought it was my code. Others in the class assured me that it's not my code, it's the model that is bad. I ran it for about 200 epochs even, after all the parameter tuning, and got accuracy around 36%, which is still bad. 200 epochs was just to make sure that it is in fact that the model is bad, it's not that learning rate was very slow and maybe it would've been good if it learned for much longer epochs and repetition. Turns out it was not efficient model, even with 200 epochs. I did not run any more 200 epochs on other models, because it took such long time. If it did not perform the best within 20 epochs, even with that many parameters and v100 gpu training on them, I just assumed it is bad model and wasn't worth time like this one running 200 epochs on them, and looked up online for good models.

Next model that I got to work was "DistilBert" model from hugging face. This was from "transformers" library, and you also have to import "DistilBertTokenizer", and "DistilBertForSequenceClassification". This was with parameter tuning, got up to 67% accuracy with 30 epochs.

Other ones I have tried have either not worked due to error I got, or performed worse than DistilBert, and rather than diving into more parameter tuning, I rather went looking and researching into more advanced models that are available on hugging face.

The best performing one I came across came from a tip from a cohort. It was ESM-2 model made by Meta, former Facebook.

https://huggingface.co/facebook/esm2_t48_15B_UR50D

Explanation on ESM-2 model
"ESM-2 is a state-of-the-art protein model trained on a masked language modelling objective. It is suitable for fine-tuning on a wide range of tasks that take protein sequences as input."

This was the actual research paper on this model that was published.

https://www.biorxiv.org/content/10.1101/2022.07.20.500902v2

“Evolutionary-scale prediction of atomic level protein structure with a language model”

This model does not use regular tokenizer. You had to download and use “AutoTokenizer”. They actually explain and walk through the process to get this model working in their colab version of the code really well.

https://colab.research.google.com/github/huggingface/notebooks/blob/main/examples/protein_language_modeling.ipynb#scrollTo=1d81db83

There is multiple versions, or checkpoints to this model as it kept developing more. The most advanced model has 15 billion parameters. I wasn’t sure if it was the shear amount of the parameter size, but I was not able to run that model on Bridges-2, so I went for “less advanced” model (still very powerful) with 8 million parameters. Even that model, I did run some CUDA memory errors, but weirdly sometimes it would run. I was finally able to get this result for 20 epoch.

```
Epoch 19: 100%|████████████████████████████████████| 261/261 [02:06<00:00,  2.07it/s, v_num=9]
`Trainer.fit` stopped: `max_epochs=20` reached.
Epoch 19: 100%|████████████████████████████████████| 261/261 [02:06<00:00,  2.06it/s, v_num=9]
Training Accuracy: 0.9946965575218201
Validation Accuracy: 0.9968980550765991
```

I set the max_epochs=20 in the code, I did not consider that Epoch 0 is also counted, so it went up to Epoch 19, I actually wanted it to go through Epoch 20, so I should’ve set max epoch argument as 21, but I was quite satisfied with the result as it shows Training accuracy of 0.9946965575218201, and Validation Accuracy of 0.9968980550765991. On the leaderboard, it said 1.00 accuracy, and the private leaderboard gave me 0.99951 accuracy.