Hi!

(Himanshu Gangwal, 22B0956) This is my project for LS : NN and LLM. The more specific technical details are up in the next portion of the text, on an overall it was a nice project to work on, In order to reach a well working model, I started with exploring the huggingface library as a whole, got to know in depth about a lot of things like pipe-lining, tokenization, training, also on the biggest scale I got to understand how big LLMs such as ChatGPT etc. work and are trained, that was too mind numbing, I would be so hard to handle such large datasets and to ensure that the model actually works. One of the most fascinating things I got to learn in the course was the architecture behind these models, at the very base its just mathematics, and allowing higher level of abstractions in the form of code is too difficult, the intricacies of the libraries such as PyTorch and TF are unbelievable, I too attempted to write a transformer myself, its a hard thing to do, but when the model really, learns its a great pleasure.

The Task

For this project I took up the task of sentiment analysis, i.e. give a text input the model classifies the piece of text as either positive or negative, along side a score.

The Base

I have used DistilBERT for the above task, the foremost reason is that, after some hours of reading and differentiating between the models, I got to know the training time depends highly on the number of parameters we would like to train, DistilBERT has around 66M such parameters, as you know to train all the parameters at once, say for a single iter of processing batch, it could so happen that the GPU vram turns out to be less than what we require to store and update the params, also it is too slow to work with for such a project. A nice way to go about it while maintaining almost complete accuracy for such a project is to freeze layers. I've quite briefly explained the same in the createModel.ipynb file provided alongside.

The Dataset

For this project we choose to us the IMDb movie review dataset, it contains 50k labelled reviews to learn from, for fine-tuning our model.

The Dumb Errors

The project for me, like every time was mostly fixing errors, rather than the Utopian where I am just coding peacefully. Uhh, anyways the errors are I think one of the best ways to learn how the code works and to reach deeper level of understand about the programs we write. For the project one of the most important thing to takeaway was to understand how parallelism offered by the gpu or the tpus enhance the training performance, when I tried to write the program for the first time I made the mistake of using BERT, after an hour of tinkering around and reading some articles on Medium I got to know why the training takes a lot of time, even for a single epoch, I got to know the methods like freezing the layers etc. to reduce the learnable(in the fine-tuning process)and thus get a better gain, then I got to know about DistilBERT. . . .

Finally. . .

At the end it was a nice course and quite fun too. I got to know a lot of cool mathematics and the implementation in code is a lot cooler to think about. . . Bye See Yaaaa.

To run the app, simply run the gradioApp.py file. Do check the path of the saved model if you are using the createModel.ipynb to create the model.