Problem statement - I*mplement a solution to forecast stock 'volatility' following earnings calls release of S&P1500 companies.*

Abstract - In recent times work has been done in order to predict various parameters of the stock market in order to understand it better. One of the key events that occur for every stock is its earning call, in which the company declares the performance and the profitability of the company in the past few months. This project is aspired to **predict the Volatility of a particular stock on the basis of the transcript of the speech given by the companies representative and the audio of the same**. Audio here plays an important role because it takes into consideration the speech pattern, confidence, and hidden truth in the speaker's mind while the words might say something else. This project is implemented **using 2 separate models, one for text encoder which encodes the text file**, BERT is used as an encoding method as it is trained on a larger dataset and would provide a better encoding. Secondly, an audio encoder taken in the .npy format files and passes it through a few Dense layers before making it ready for further layers. The model is trained on different loss functions and the results signify that the model is training well.

Overview of the method -

1) The first step all of this process was to **extract all the data that was scattered placed into one master dataset**. This was done using basic python functions in which the text file was extracted from the directory based on the name and date of the earning call. Further using the same name and date a numpy file was also taken from the dataset directory and was loaded. Both these data were added to a CSV file to keep it together for further pre-processing.
2) The next step was to **calculate the parameter to measure the volatility index**, referring to the paper - VolTAGE: Volatility Forecasting via Text Audio Fusion with Graph Convolution Networks for Earnings Calls - Ramit Sawhney et. al a parameter of stock volatility which is defined as the natural log of the standard deviation of return prices *r* in a window of 3 days. Return price was in turn calculated using the closing price of that particular stock which was taken from the Yahoo finance API. Functions to calculate all these values were made which was then iterated to calculate the value for all stock and then appended to the earlier CSV file.
3) The next step was to **preprocess** the data in the CSV file. The text data was preprocessed as per the BERT model requirement and was padded with the appropriate length. An embedding was made of the text file using functions provided with the BERT model, only 100 earning calls were taken taking into consideration the high amount of data and limited processing power. The numpy file in the CSV file was reshaped appropriately so as to make it able to pass through the model along with the other data.
4) Further, a model needed to be created that takes in the text file and the processed numpy file and **gives out the target value** (volatility index). The model needed to **concatenate** both the type of values and then further pass through the layers to predict the value. The model was created using the Keras and Tensorflow platform. The performance of the model was tested on the 2 losses, MAE and MSE, and was trained using Adam optimizer with an optimal learning rate of 0.01 and was trained for 20 epochs.

*The model is shown in figure 1.1 on the next page.

Performance of the method- The model was trained and tested upon 2 losses MAE and MSE results of both are provided below.

Loss function	Epochs	Training loss	Validation Loss
MSE	20	0.8281	1.1664
MAE	20	0.7552	0.8564

The model was also run for 10 and 20 epochs but the latter performed better with both the loss functions. The model was tested on different optimizers (SGD, RMSProp) but Adam outperformed every one of them.

Inferences -

1) We see that the loss has reduced during the training process which signifies that the model has been training and **no signs of gradient vanishing/exposing can be seen**.
2) We know that MSE penalizes more for any outlier values. Having looked upon both the loss values which come out to be nearly the same **it can be justified that the predictions by our model do not include any outlier points**.

Conclusion -

The model is training properly and the values predicted of the average log volatility **is not** any arbitrary large values away from the target values.

Scope for future work -

1) Currently the audio encoder works on the numerical values of the audio file. A spectrogram of the audio file could be plotted and **CNN could be run on that spectrogram** in order to capture the hidden intricate details, and then merged with text encoder as before.

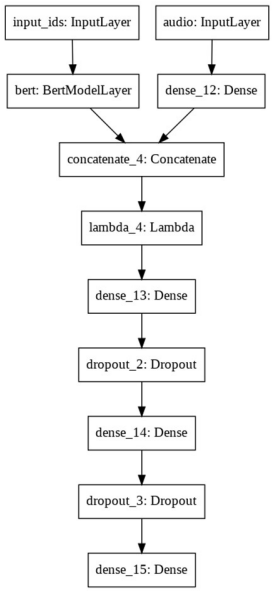

Figure 1.1 showing the model details

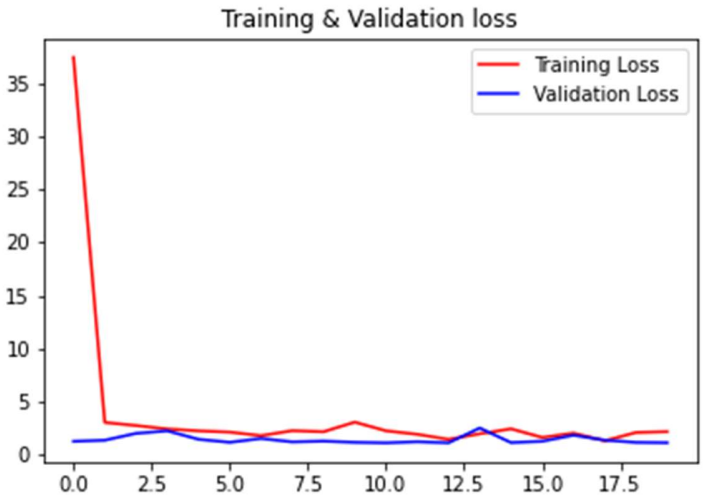

Figure 1.2 showing the Loss in MSE while training.