**ASSIGNMENT 2: SENTIMENT CLASSIFICATION**

* *Aim:*

Use Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) for sentiment classification.

* *Github Implementation:*
* *Observations:*

1. Dataset Overview –

The dataset used for the same is the Yelp Restaurant Revire Datset which consisted of millions of rows. However, for simplicity purposes, we selected the *top 100k rows*. During data analysis, we encountered data imbalance across classes, and thus to balance the classes I choose the top *10000 entries per sentiment for training-testing purposes*. Post the data selection, basic pre-processing techniques like tokenization, stopwords removal, lemmatization, and stemming were applied to the final dataset which was then used to create the *VOCAB*  for the model. Moreover, since the length of every sentence varies, for model training we had to make sure that every sentence has the same length. Upon further feature analysis, I came up with a *sentence length max to be 500 words* where sentences having a length greater than this are clipped, and less than this are padded with *pad* tokens. A *custom word-2-vec model* was trained, which was used to generate the word embeddings for all the words.

*NOTE: The data and the embeddings used across both models remains the same for comparative purposes*

1. Convolution Neural Network (CNN) –

CNNs are a kind of deep learning model that is regularly utilized for image classification. Nonetheless, they can likewise be utilized for text classification tasks, like sentiment analysis. In sentiment analysis, the goal is to classify a piece of text as positive, negative, or neutral. CNNs can be utilized for this task by first changing over the message into a grouping of vectors, where every vector addresses a word or expression. The CNN then, at that point, applies a progression of convolutions to these vectors, which permits it to learn local patterns in the text. These examples can then be utilized to classify the text as certain, negative, or neutral.

The benefits of involving CNNs in sentiment analysis are as follows:

* They can learn local patterns in the text, which can help identify sentiment.
* They can handle long sequences of text, which is important for sentiment analysis.
* They can generalize well to new data.

The drawbacks of involving CNNs r sentiment analysis:

* They can be computationally expensive to train.
* They can be sensitive to the choice of hyperparameters.
* They can be difficult to interpret.
* Model Specifics:

1. Hyperparameters

* Embedding Size: 500
* No of filters: 10
* Activation Function: Tanh/Relu
* No of Epochs: 5
* Optimizer: Adam
* Lerarning Rate: 0.001
* Loss Function: CrossEntropyLoss

The total parameters for the below-given model are around *13445163*. However, out of these, the trainable parameter is around *55163*. The model was trained for 5 epochs only.

Text

Description automatically generated

Figure 1: CNN Architecture

The table below shows the comparative analysis of the performance of the CNN model with TanH and Relu as activation functions for the same set of hyperparameters and input

Chart

Description automatically generated with medium confidence Chart

Description automatically generated

Fig 2: Tanh as Activation Function Fig 3: Relu as Activation Function

From the above-given testing results, I observe that the if *Relu* is used as an activation function it performs slightly better than the *TanH*. This is because it does not suffer from the *vanishing gradient problem*, thereby increasing the performance of the model. The graphs show the training and validation loss across the 5 epochs. I observed throughout training that the model is learning and thus the loss is decreasing. However, after a certain epoch, the model is kind of overfitting and I believe with the proper set of hyperparameters we can overcome this overfitting and increase the robustness. Also, the classification report shows the metrics based on the testing set. On an overview level, the model can achieve an accuracy of *77%* and *78%* with Tanh and Relu as activation functions respectively.