1. **Baseline**

   1.1 LSTM

   LSTM can achieve a better result by adjusting the batch size so that the training time can be shortened without worrying about the overfitting problem. We used batch normalization, N-gram, and multiple inputs to improve the model's performance. We found that using a proper batch size reduced training time and prevented overfitting. N-gram was used to generate more features and capture the context of words, with a 2-gram approach being found to be effective. Lastly, we found that batch normalization improved training accuracy.

   1.2 Logistic Regression

   Changing hyperparameters (C value) in logistic regression can control the regularization strength. Overfitting will occur when C is too small, and vice versa. We found that it is optimum to set C to be 10, allowing the logistic regression model using tifdf to perform better than the strong baseline.

2. **BERT**

   2.1 BERT model

   We adopted the Bidirectional Encoder Representations from Transformers (BERT), which is a common baseline model for Natural Language Processing (NLP) developed by Google.

   2.2 Hyperparameters

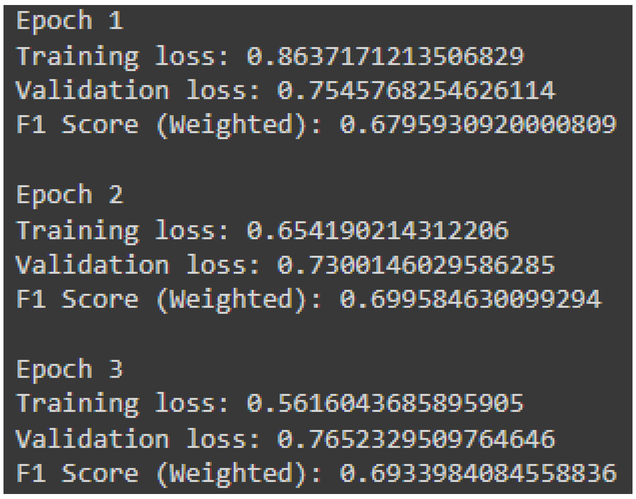
   2.2.1 Batch size

   It affects the how the Adam Optimizer operates. When it is too small, underfitting is likely to happen and it takes a long time to process. On the other hand, if it is too large, the system will collapse as the RAM size is inadequate. After trial and error, we used the batch size of 16 to obtain the best accuracy.

   2.2.2 Learning rate

   We used 1e-5 in our model as the result is usually better for a lower learning rate.

3. **Results**



```
Epoch 1
Training loss: 0.8637171213506829
Validation loss: 0.7545768254626114
F1 Score (Weighted): 0.6795930920000809

Epoch 2
Training loss: 0.654190214312206
Validation loss: 0.7300146029586285
F1 Score (Weighted): 0.699584630099294

Epoch 3
Training loss: 0.5616043685895905
Validation loss: 0.7652329509764646
F1 Score (Weighted): 0.6933984084558836
```