**HW6 -Tweet Emotion Detection using SLMs**

This assignment aims to classify tweet data into multiple emotions called anger, anticipation, disgust, fear, joy, love, optimism, pessimism, sadness, surprise, trust. The main objective of this task is to predict multiple emotions conveyed in a single tweet using different models.

Sentiment analysis in today’s digital age has become crucial to the companies in determining the user behaviour and also determine the opinion of masses about a specific topic. In this assignment, I am using three models called DistilBert, Roberta and Flan T5 to do the multi-label classification of emotions on tweet data. By employing these three models my aim is to evaluate their performance and effectiveness in doing the multi-label classification.

**Data Source:**

The dataset is collected from a Kaggle competition which has two files ‘train.csv’ and ‘test.csv’. The ‘train.csv’ file is used to train the models as well as the evaluate the performance of the models. This train dataset has 7724 tweets with 11 emotions namely anger, anticipation, disgust, fear, joy, love, optimism, pessimism, sadness, surprise, trust.

**Data Preparation:**

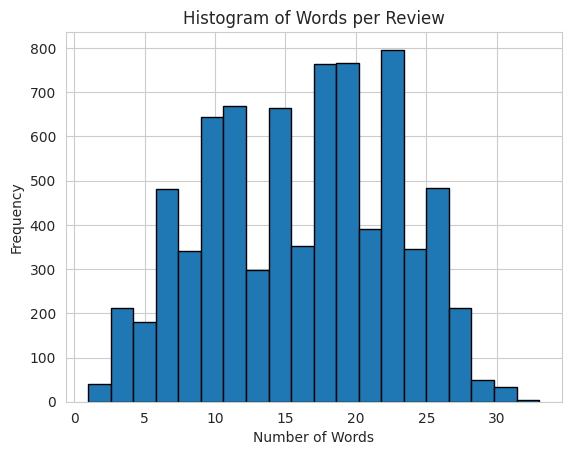
The 7724 tweets from the train.csv file is cleaned by using custompreprocessor from spaCy. This cleaning involves removing html tags if there are any.

From the train.csv, the column ‘tweet’ is taken as feature and the columns anger, anticipation, disgust, fear, joy, love, optimism, pessimism, sadness, surprise, trust are taken as labels and are converted into an array of labels for each tweet.

Since this is a multi-label classification problem, loss function is calculated using binary cross-entropy loss and that requires the labels to be of float type. Therefore, the array of labels is converted to float type.

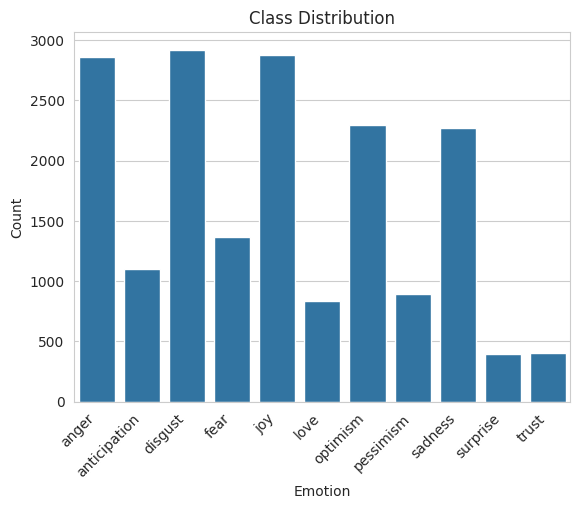
**Exploratory Data Analysis:**

Analysis of the length of the text:



From the above histogram, we can see that the max length of tweets is 35 which is less than the maximum tokens that the proposed models take which is 512 tokens. Therefore, there is no need for truncation of any tweet to fit the capabilities of the proposed models.

Class Distribution Analysis:



From the above bar graph, we can see that there is class imbalance among the distribution of labels across the tweet data. It can be observed that that the emotions like anger, disgust, joy are more occurring i.e. more than 2500 times as compared to the emotions like surprise and trust which are occurring less than 500 times. This highlights the class imbalance present in the data and it is important to take the class imbalance into consideration for better performance of the model.

**Tokenization:**

In order to process the text data to feed the model it must be tokenized i.e., converted into tokens and assign a numerical id to the tokens to give as inputs to the model. In my models, I am using DistilBert Tokenizer by initializing it using the ‘distil-bert-uncased’ checkpoint. In the tokenization function for all the models, I am taking a set of texts and implementing tokenization through truncation and then map the train data to tokenize the text into batches.

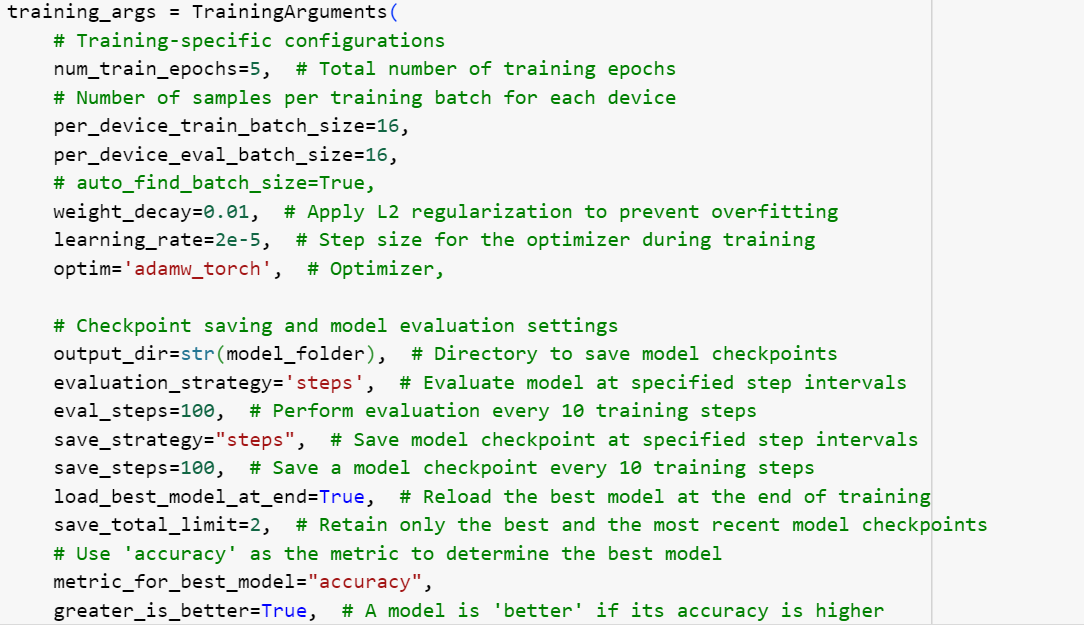
**Model Training:**

**Experiment 1: Using Distil Bert**

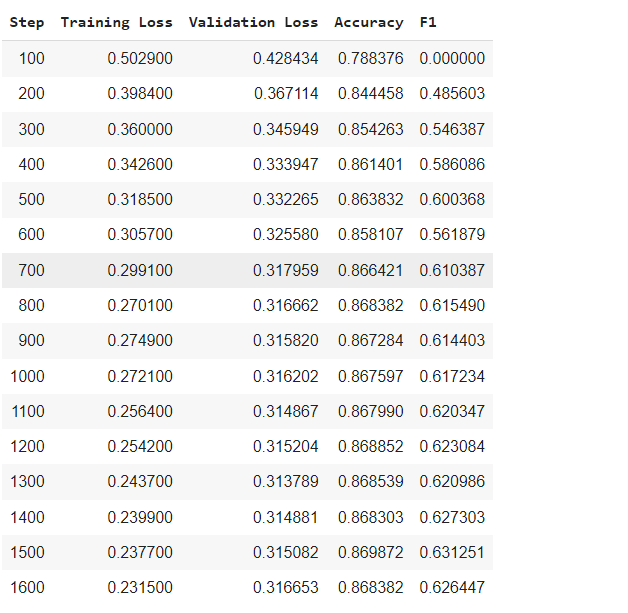
A pretrained DistilBert model is used with AutoModelForSequenceClassification. This variant includes a classification head atop the pretrained model, facilitating easier training with the base model. It is necessary to specify the number of labels to predict, which in this case is 11, as this determines the number of outputs for the classification head.

For tracking metrics during training, a compute\_metrics() function is defined for the trainer. This function will compute the accuracy and macro-F1 score of the model.

Training the model: A training arguments class is used to define the training parameters which can be used to fine-tune the training and evaluation process.

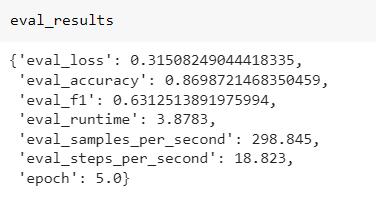


With this training arguments, the trainer is instantiated and it gave the following output:

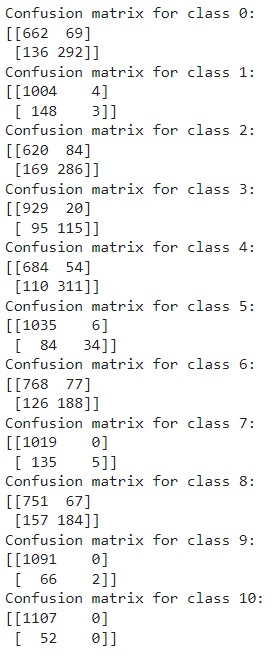


As we can see that the F1-score after 5 epochs is 62.6% which is a relatively good score but this score can be improved as I have not taken the class imbalance into consideration for the current model.

Validation results:



The confusion matrix for the DistilBert Model is as following:



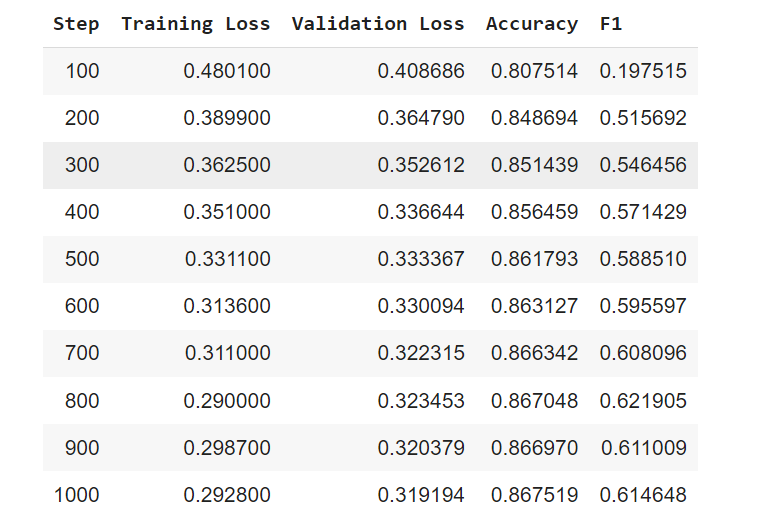
It can be seen that the model is not performing well on the minority classes like 9 and 10 which are surprise and trust. From the confusion matrix we can see that the true positives for these minority classes are low. This can be improved through cost-effective learning by giving more weights to these minority class while calculating the binary cross entropy loss function.

**Experiment 2: Using Roberta**

For this 2nd Experiment, tokenization was one using Roberta Tokenizer using the 'distilroberta-base' checkpoint. The model is pretrained with ‘distilroberta-base’ using same auto model for sequence classification using the 1st experiment.

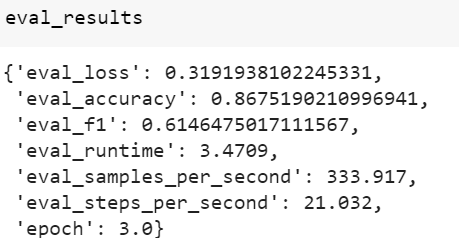
Same compute\_metrics function is used to for the 2nd experiment and the training process used for the distilbert is also implemented for Roberta.

The output of the second model is as follows:



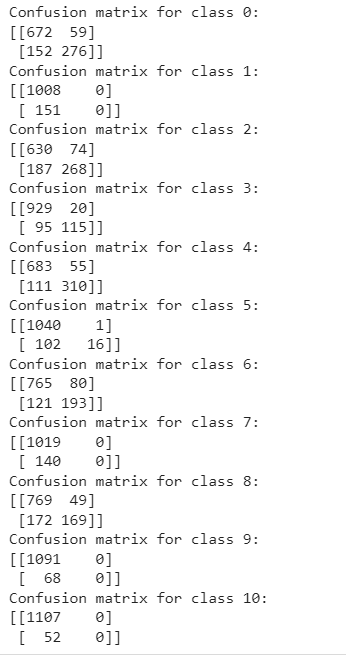
From the output it can be seen that the model is converging towards a stable solution indicated by the decreasing trend of training and validation loss. The F1-score at the model’s best checkpoint of 1000 is 61.46% which can be further improved by taking the class imbalance into consideration.

Validation Results:



Confusion Matrix of experiment 2:

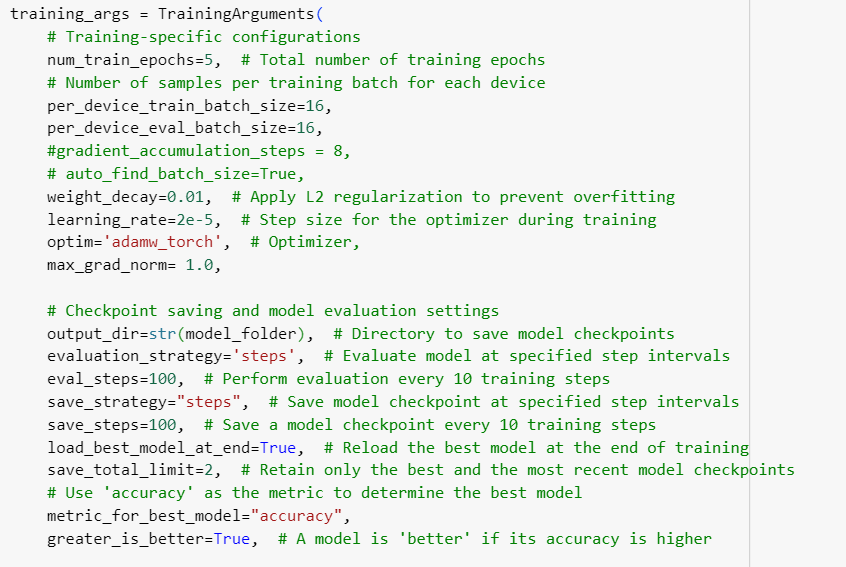
The confusion matrix of Roberta is showing similar results as compared to the first model Distilbert. It can be seen that Roberta is performing well on classes that are in majority like anger, disgust and joy while performing poorly or not able to do positive predictions on minority classes like surprise and trust. The current model can be further improved by taking the class imbalance into consideration.

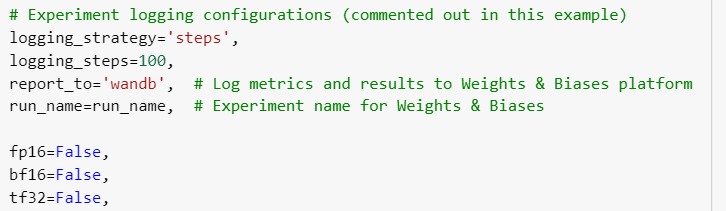


**Experiment 3: Using Flan-T5**

For the third experiment, google’s Flan-t5-base is used to pretrain the model using the Auto model for sequence classification. The tokenization for this model is done using AutoTokenizer where T5Tokenizer checkpoint is getting used. The T5 is an encoder-decoder model that takes text input and gives a text output. During the training, using AutoModelforSequenceClassification, the model will internally run the T5ForSequenceClassification which encodes the given input into tokens and creates a special token called <eos> token which is then fed to the T5classification head to do the multi-label classification.

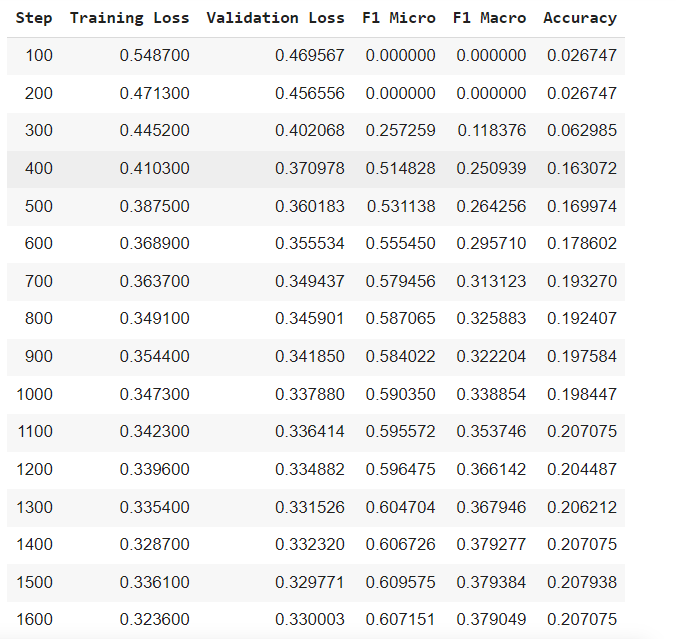
Following training arguments are defined to train the model:





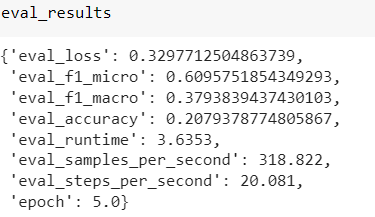
The compute\_metrics function is defined to calculate F1-micro, F1-macro and accuracy scores for the multi-label outputs.

The output of the second model is as follows:

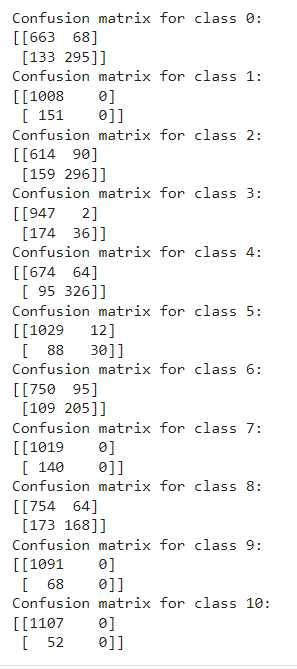


The best model was found to be at step 1500 where the accuracy is found to be 20.7% and the macro-F1 score is 37.93%. From the model output it can be seen that the F1-score is gradually increasing, it is still relatively low as compared to distilBert and Roberta. The accuracy is also increasing but remains relatively low, indicating that the model is still not performing optimally. This could be due to several reasons such as insufficient training data, model complexity, or suboptimal hyperparameters.

Validation Output:



Confusion Matrix:



**Metrics to consider:**

For comparing the three models, Macro F1 score and accuracy are being used to compare and evaluate the performance of the three models.

|  |  |  |
| --- | --- | --- |
| **Model** | **Macro-F1 Score** | **Accuracy** |
| DistilBert | 0.6312 | 86.98% |
| Roberta | 0.6167 | 86.8% |
| Flan-T5 | 0.3793 | 20.7% |

**W&B links of the projects:**

Experiment 1: <https://wandb.ai/vyshnavi-utd/emotion_distilbert>

Experiment 2: <https://wandb.ai/vyshnavi-utd/emotion_distilroberberta>

Experiment 3: <https://wandb.ai/vyshnavi-utd/emotion_flanT5_final>

**Comparison of the models and Performance evaluation:**

Based on the metrics, DistilBert appears to be the best-performing model for this multi-label classification task. The high Macro-F1 score and accuracy suggest that DistilBert is able to accurately classify the examples across all the labels, outperforming both Roberta and Flan-T5.

The differences in performances can be attributed to model’s training and inherent characteristics. DistilBert is a more compact and effective variant of the well-known BERT model, which has demonstrated strong performance on numerous natural language processing tasks, such as multi-label classification. In contrast, Roberta is a BERT variation that has undergone additional optimization and training on a broader corpus of data, potentially resulting in better performance.

Flan-T5, a large language model built on the Transformer architecture, performed noticeably worse than the other two models. This could be because it is a different kind of model and isn't as appropriate for this particular multi-label classification task as the BERT-based models. However, the performance of all the three models can be improved by fine-tuning them using the training arguments.

It is also observed from the three models that they are not performing well on predicting the positive labels for the minority classes like surprise and trust. This behaviour of the models can be corrected by taking class imbalance into consideration and giving more weights t the minority class during training.

**Further Improvements:**

* One of the improvements would include choosing a dynamic learning rate while training the model to improve the performance.
* Taking synthetic data for the minority classes like surprise and trust and training the model on the synthetic data to increase the predictability of these emotions by the models.