CSE4037 - Deep Learning

J Component Report

A project report titled

Emotional Analysis in Tamil

By

19MIA1014	Lokesh Kanna P
19MIA1076	Arun Venkat S J
19MIA1097	John Chacko

Computer Science and Engineering with Specialization in Business Analytics

Submitted to

Dr. R. Rajalakshmi

School of Computer Science and Engineering

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

April 2022

DECLARATION BY THE CANDIDATE

I hereby declare that the report titled "**Emotional Analysis in Tamil"** submitted by me to VIT Chennai is a record of bona-fide work undertaken by me under the supervision of **Dr. R. Rajalakshmi, Associate Professor, SCOPE, Vellore Institute of Technology, Chennai.**

Signature of the Candidate

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. R. Rajalakshmi,** School of Computer Science and Engineering for her consistent encouragement and valuable guidance offered to us throughout the course of the project work.

We are extremely grateful to **Dr. R. Ganesan, Dean,** School of Computer Science and Engineering (SCOPE), Vellore Institute of Technology, Chennai, for extending the facilities of the School towards our project and for his unstinting support.

We express our thanks to our **Head of the Department** for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the courses.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

BONAFIDE CERTIFICATE

Certified that this project report entitled "**Emotional Analysis in Tamil"** is a bona-fide work of **Arun Venkat S J (19MIA1076), Lokesh Kanna (19MIA1014), John Chacko (19MIA1097)** carried out the "Emotional Analysis in Tamil" - Project work under my supervision and guidance for CSE4037 - Deep Learning

Dr. R. Rajalakshmi

SCOPE

TABLE OF CONTENTS

ABSTRACT

Our project aims to perform an emotion analysis of social media comments in Tamil. The emotional analysis is the classification task of mining emotions in texts, which finds use in various natural language applications such as reviews analysis in e-commerce, public opinion analysis, extensive search, personalized recommendation, healthcare, and online teaching. The goal of this task is to identify whether a given comment contains which emotions.

In this project, we report the analysed findings in the "Emotion Analysis in Tamil." from the dataset collected The task aims to detect and recognize types of feelings through the expression of texts like Joy, Anger, Trust, Disgust, etc. We implemented transformer-based models like MuRIL, XLM-R and M-BERT to tackle the problem.

MuRIL performed the best with a macro-averaged f1 score of 0.31.

INTRODUCTION

In today's Internet world, humans express their Emotions, Sentiments and Feelings via text/comments, emojis, likes and dislikes. Understanding the true meanings behind the combinations of these electronic symbols is very crucial and this is what we are trying to achieve.

We have enormous amount of data to maintain it is required to classify large textual data according to the emotions which will help to standardize the platform, make it easier to reduce the toxic environment in social platforms which improves user experience. Remarkably, machine intelligence and deep learning are planting roots at most unimaginable and orthodox areas as well. ***"It's not what you say, but how you say it"***, Interactions, facts and feelings shape our relationships. Expressions matter, as do the sentiment behind each encounter and the emotions raised. Emotion is entwined with the literal meaning of words used. This fact/feeling principle applies to both inter-personal and business relationships.

Emotion analysis is the process of identifying and analyzing the underlying emotions expressed in textual data. Emotion analytics can extract the text data from multiple sources to analyze the subjective information and understand the emotions behind it.
Tamil is one of the longest-surviving classical languages in the world and is the 18th most spoken language all over the world (75 million). So, developing a method to analyze emotion of Tamil text will benefit many.

In our project we have different methods to analyze emotions from the social media comments. To do so we have used transformer-based approaches.

LITERATURE SURVEY

1. The first paper proposed creating a code-mixed Tamil-English dataset using sentiment-annotated corpus containing 15,744 English typed Tamil (Tanglish) comment from YouTube posts. Then benchmarking by classify the input dataset into Positive, Negative, Mixed feelings, Neutral, Other language using various classification algorithms (LR, SVM, K-NN, DT, RF, MNB, 1DConv-LSTM, BERT) but performed poorly on the code-mixed dataset. 67% belong to Positive class got more positive sentiment than others as the people who watch trailers are more likely to be interested in movies and this skews the overall distribution. But they achieved a high inter-annotator agreement in terms of Krippendorff's alpha value of 0.6 from voluntary annotators.

2. In the second paper they used Troll Classification dataset of Tamil Memes and classified it into troll or non-troll using Multimodal deep learning model using Vision transformer for images and Bidirectional Encoder representations from Transformers (BERT) for captions of memes. The model scored a F1 score of 0.46 on the test set and 1.0 on the validation set. Vast difference was observed due to high bias, the model achieved a perfect 1.00 weighted F1-score on the validation set mostly because of pre-processing of the images. The algorithm overfitted the train set undoubtedly. The reason behind the poor performance was due to the change in the distribution.

3. In third paper paper they used Dravidian–CodeMix-FIRE2020 Dataset which contains code-mixed Tamil and Malayalam as input to classify it as Positive, Negative, Mixed feelings, Unknown state as output for sentiment analysis and they obtained the best results for Tamil task and Malayalam task using AWD-LSTM model with ULMFiT framework using the FastAi library. Accuracy scores were better with Tamil compared to Malayalam due to comparatively larger database for Tamil. So, performance can be improved with huge datasets and tuning parameters.

S. No	TITLE	JOURNAL/ YEAR OF PUBLICATION	DATASET USED	METHODOLOGIES USED	METRICS USED	INTERPRETATION OF RESULTS
1)	Review on Sentiment analysis in Tamil texts	2018 CEUR Workshop Proceedings (CEUR-WS.org)	Social media comments	Features: TF of unigram and bigram. Classifiers: MNB, BNB, Logistic regression, RKS and SVM.	Different approaches were used for feature representation including presence of words, Bag of Words (BoW), Term Frequency (TF), Term Frequency- Inverse Document Frequency (TF-IDF) and Word2vec vectors and different classifiers were used for classification	This review paper aims to critically analyse the recent literature in the field of SA with Tamil text. Pre-processing, Corpus, Methodologies and success rates are taken in consideration for the review. Performance of SA model depends on the pre-processing steps such as negation handling techniques and stop words removal. They have concluded that SVM and RNN classifiers taking TF-IDF and Word2vec features of Tamil text give better performance than grammar rules-based classifications and other

						classifiers with features presence of words, TF and BoW.
2)	Deep Learning Based Sentiment Analysis in Code-mixed Tamil-English Text	2020 International Journal of Engineering Research & Technology (IJERT)	Dravidian-CodeMix-FIRE2020	Bi-LSTM model has been used for sentiment analysis in code-mixed Tamil-English text. For the baseline model, they used various machine learning approach such as Logistic regression, (LR), Support vector machine (SVM), Decision tree (DT), Random Forest (RF), Multinomial Naive Bayes (MNB), K-nearest neighbours (KNN).	Classifying the given text as positive, negative, neutral, mixed-feeling and non-Tamil.	The paper classified the given input comments s into Positive, Negative, Mixed-Feeling, Neutral and non-Tamil on the code-mixed data given by Dravidian Codemix-FIRE 2020 task. Deep Learning based Bi-LSTM model is used for classification in their implementation . F1-Score, Precision, Recall metrics are used for evaluation purpose.

3)	Findings of the Sentiment Analysis of Dravidian Languages in Code-Mixed Text	2021 CEUR Workshop Proceedings (CEUR-WS.org)	Dravidian-CodeMix-FIRE2021	Logistic regression model, an LSTM classifier, and a multilayer perceptron classifier	Classifying the given text as positive, negative, neutral, mixed-feeling and non-Tamil.	The paper used LSTM and traditional machine learning algorithms such as Naive Bayes (NB), K-Nearest Neighbours, etc. did not yield good results compared to the transformer-based models.
4)	Tamil English Language Sentiment Analysis System	2016 International Journal of Engineering Research & Technology (IJERT)	Tamil language reviews	Applied an iterative KNN strategy to propagate their polarity to other words.	Accuracy for this knowledge base can be detected using the Confusion Matrix.	The paper used Naive Bayesian classifier is used for supervised learning and select top 30 % informatively predicted reviews for training, with which the classifier exhibits the best performance.

5)	Corpus Creation for Sentiment Analysis in Code-Mixed Tamil-English Text	2020 European Language Resources association	Sentiment-annotated corpus containing 15,744 comment posts from YouTube	Classifiers: LR, SVM, K-NN, DT, RF, MNB, 1DConv-LSTM, BERT-Multilingual, DME and CDME.	Krippendorff α for measuring reliability coefficient developed to measure the agreement among observers. Classifies the input as Positive, Negative, Mixed feelings, other languages as output for sentiment analysis and measuring precison, recall and F-Score.	The paper is all about corpus creation using YouTube comments and benchmarking them using various machine learning algorithms which but performed poorly on the code-mixed dataset. Being analysed 67% belong to Positive class so it got more positive sentiment than others as the people who watch trailers are more likely to be interested in movies and this skews the overall distribution. But they achieved a high inter-annotator agreement in terms of Krippendorff's alpha value of 0.6 from voluntary annotators

6)	Sentiment Code-Mixed Text Classification in Tamil and Malayalam using ULMFiT	2020 CEUR Workshop Proceedings (CEUR-WS.org)	Dravidian-CodeMix-FIRE2020	AWD-LSTM model with pre-trained models for NLP – Universal Language Model Fine-tuning for Text Classification (ULMFiT) framework using the FastAi library. The FastAi library provides functions to create classification data bunch and Language model data bunch. In language modelling, The RNN learns about the next word from the previous word.	Classifies the input as Positive, Negative, Mixed feelings, Unknown state as output for sentiment analysis	This paper dealt with code-mixed Tamil and Malayalam as input to classify it as Positive, Negative, Mixed feelings, Unknown state as output for sentiment analysis and they obtained the best results for Tamil task and Malayalam task whose performance can be improved by huge datasets and tuning parameters.

PROPOSED METHODOLOGY

To classify social media comments into different emotions, we used transformer-based models. So, we have taken the dataset from shared task on Emotion Analysis in Tamil-ACL 2022 which aims to classify the social media comments into categories of emotions. The Tamil sentiment analysis dataset contains eleven categories of emotions (***That is, neutrality, joy, ambiguity, trust, disgust, anger, expectation, sadness, love, surprise, and fear*** [11 Classes]). The training, development and testing datasets are of ***14,208***, ***3,552*** and ***4,440*** data points respectively. Each data point in the training data has a Tamil text and a corresponding label in English.

The data sets consist of Tamil and Tamil-English codemixed data, and we have used three transformers MuRIL, XLM-RoBERTa and M-BERT.

MuRIL is a language model built explicitly for Indian languages and trained on large amounts of Indic text corpora. MuRIL, short for Multilingual Representations for Indian Languages, is none other than a free and open-source Machine learning tool specifically designed for Indian languages. Google's Indian Research Unit has launched it in the year 2020. It helps to build local technologies in vernacular languages with a common framework. Google has trained the MuRIL tool using an existing language learning model called BERT (Bidirectional Encoder Representations from Transformers).

M-BERT or multilingual BERT is pre-trained on 104 languages using masked language modeling (MLM) objective. BERT is a transformers model pretrained on a large corpus of multilingual data in a self-supervised fashion. This means it was pretrained on the raw texts with an automatic process to generate inputs and labels from those texts.

XLMRoBERTa is a multilingual version of RoBERTa. XLM-R was trained on 2.5TB of newly created clean Common Crawl data in 100 languages. It outperforms previously released multi-lingual models like mBERT or XLM on tasks like classification, sequence labeling and question answering.

The training was stopped early if the f1 score did not improve for three consecutive epochs since we used HuggingFace for training withSimpleTransformers.

The punctuations, URL patterns, and stop words has been removed and for better contextual understanding, we replaced textual equivalents of emojis. For example, The Tamil equivalent of the word laughter instead of the laughing emoji. The performance of all transformer models except for MuRIL has seen a boost after Data cleaning. MuRIL and all ensemble models worked best without data cleaning.

There is a significant class imbalance in the data. To reduce the imbalance, we used the following techniques:

- Over-sampling,
- Over-under sampling,
- Synthetic minority over-sampling (SMOTE)
- Assigning class weights.

In over-under sampling, we under-sample the classes having more instances than expected and over-sample those having lesser instances than expected while keeping the length of the dataset constant.

Over-under sampling worked best for all transformer models. Assigning class weights to the input boosted the performance of the M-BERT - Logistic Regression ensemble model.

RESULTS AND DISCUSSION

MuRIL outperformed all other models with a macro-averaged f1 score of 0.31 and a weighted-average score of 0.37 whereas **M-BERT** performed better with a macro-averaged f1 score of 0.27 and a weighted-average score of 0.36 and worst with **XLM-R** with a macro-averaged f1 score of 0.01 and a weighted-average score of 0.01.

So, MuRIL performed the best (11 classes)

The results obtained are given Table below.

Classifier	**Macro avg f1**	**Weighted avg f1**
MuRIL	0.31	0.37
XLM-R	0.01	0.01
M-BERT	0.27	0.36

CONCLUSION

After we have addressed exciting known and unknown problems in various code-mixed research papers, the goal of our project has to classify the emotions in social media comments in Tamil and we used transformer-based models like MuRIL, XLM-RoBERTa and M-BERT.

The performance has improved by changing the value of parameter and by handling of data imbalances and change in distributions to be carefully handled. We observed size and source of the dataset affects the performance of the model.

Out of these models, for task MuRIL outperformed all other models with a macro-averaged f1 score of 0.31.

It is observed that the models classify emotions like Joy, Sadness, Neutral, and sentences having ambiguity well. However, the models classify more complex emotions like anger, fear, and sadness with much less accuracy.

Apart from using various machine learning methods, linguistic information or hierarchical meta-embedding or genetic algorithm-based ensembling can be tried can be utilized to improve the performance of the models.

.....

APPENDIX

Implementation / Code

In [1]:
```
import pandas as pd
import re
from collections import Counter
import matplotlib.pyplot as plt
from statistics import mean, median, mode
```

In [2]:
```
df_train = pd.read_csv(r'ta-emotion10-train.csv', sep='\t', names=['emotion','commen
df_train.head()
```

Out[2]:

	emotion	comment
0	Neutral	நாளைக்கு அரிசிக்கு இந்த நிலமை வந்தா 🙄
1	Anger	மானம் கேட்ட அன்புமணி
2	Neutral	தவறு இஸ்ரேல் இருக்காது இதை நான் கூறவில்லை ஹமாஸ...
3	Joy	கொங்கு நாட்டு சிங்கம் உன்மையும் நேர்மையும் உலை...
4	Neutral	இவர் யார்? ஒவ்வொரு வார்த்தையும் முன்னுக்கு பின...

In [3]:
```
df_train.shape
```

Out[3]:
```
(14208, 2)
```

In [4]:
```
df_dev = pd.read_csv(r'ta-emotion10-dev.csv', sep='\t',names=['emotion','comment'])
df_dev.head()
```

Out[4]:

	emotion	comment
0	Joy	அருமை அற்புதம் பிரமாதம் நண்பரே வாழ்த்துக்கள் ந...
1	Anticipation	வேல்ராஜ் வேலையா தான் இருக்கும்
2	Joy	அண்ணன் கிட்டுக்கு வாழ்த்துக்கள் 👍 👍
3	Trust	ஆமா நானும் இதான் யோசித்தேன் 🤣🤣
4	Anticipation	மொத்த மக்களும் ஒன்னு சேர்ந்தாதான் இந்த அரசாங்க...

In [5]:
```
df_dev.shape
```

Out[5]:
```
(3552, 2)
```

In [6]:
```
df_test = pd.read_csv(r'task_a_test.csv', sep='\t',names=['emotion','comment'])
df_test.head()
```

Out[6]:

	emotion	comment
0	Ambiguous	நம்மூரில் நம்மொழியில் வழிபாடு செய்ய இவ்வளவு இட...
1	Disguist	தமிழ் நாட்டிற்க்கு வெளியே போய் வாழ்ந்து பாருங்...
2	Disguist	ஆழி ரொம்ப சொம்பு தூக்காத திமுகவிற்கு
3	Ambiguous	நா என்ன சொன்னேன்.

	emotion	comment
4	Joy	மிக நல்ல அரசியல் கலாச்சாரம் நம்ம முதல்வர் 🙏 🙏 🙏 🙏 ...

In [7]:
```
df_test.shape
```

Out[7]: (4440, 2)

In [8]:
```
import unicodedata as ud

latin_letters= {}

def is_latin(uchr):
    try: return latin_letters[uchr]
    except KeyError:
         return latin_letters.setdefault(uchr, 'LATIN' in ud.name(uchr))

def only_roman_chars(unistr):
    return all(is_latin(uchr)
           for uchr in unistr
           if uchr.isalpha()) # isalpha suggested by John Machin
```

In [9]:
```
count = 0
for index, row in df_train.iterrows():
    if not only_roman_chars(row['comment']):
        print(index, row['comment'], row['emotion'])
        print('\n\n')
        count += 1
print(1 - count/len(df_train))
```

0 நாளைக்கு அரிசிக்கு இந்த நிலமை வந்தா 🙄 Neutral

1 மானம் கேட்ட அன்புமணி Anger

2 தவறு இஸ்ரேல் இருக்காது இதை நான் கூறவில்லை ஹமாஸ் நிறுவனர் யாசின் மாலிக் கூறியது Neutral

3 கொங்கு நாட்டு சிங்கம் உன்மையும் நேர்மையும் உலைப்பும் இருந்தால் எல்லாம் நம்மை தேடி வரும் வாழ்த்துக்கள் ஐயா இது அவருடைய தனிப்பட்ட வாழ்க்கை நன்றி Joy

4 இவர் யார்? ஒவ்வொரு வார்த்தையும் முன்னுக்கு பின் முரணாக உள்ளது Neutral

5 தினமும் ஸ்டாலின் செருப்ப தொடைத்து கொடுக்குற வன் தான் இந்த ராசா Disgu ist

6 உண்மையை உணர வைத்த உத்தமர்! Trust

7 அட முட்டா கூ எத்தனை கிராமத்தில் டா இருக்கு வெண்ணை எல்லாமே மாநக ரில் மட்டும் தான்டா இருக்கு மாநகரில் மட்டுமே அடித்தட்டு மக்களிடம் பண பு ழுக்கம் அதிகமா இருக்கு அந்த பணத்தை டாஸ்மாக் மூலமா பணம் பெறுவதே இ துல 80% குடிகார மக்கள் இதே கிராமத்தில் வைத்தால் டாஸ்மாக் வருமானம் வராது இதுதாண்டா அவர்களின் எண்ணம் இந்த மயிரு தெரியாம குற்றவாளி நம்பர் 1 பா ராட்டி கிட்டு இருக்க Disguist

8 அருமை நண்பா ...தமிழன் என்பதே பெருமை .. Joy

9 உன்மை... அனைத்து கட்சிகளிலும் வாரிசு அரசியல் ஆட்கொண்டுள்ளது Neutral

10 தமிழ் சினிமாவில் கஞ்சா கருப்பு போன்று இவரும் ஒரு வலம் வருவார் Neutral

11 காவல் துறையினரை அனைவரும் கடவுளாக பார்க்க வேண்டும்.... Anticipation

12 @உலகப் பொதுமறை வாழ்த்துக்கள் நண்பா கொரொனாவை ஒரு நெருக்கம் நி றைந்த மக்கள் கட்டு படுத்தி இருக்கிறார்கள் என்பது எங்களுக்கு மகிழ்ச்சி ஏ னென்றால் சென்னை இங்கே கதருது நண்பா Neutral

13 நமது குரு அவர்களிடம் கேட்டு கொடுக்கப்படும் ..நன்றி 🙏🙏🙏 Trust

14 அருமை. தலைசிறந்த முதல்வர். இன்றைய இந்தியாவிற்கு நீங்கள் ஒரு மு ன்னோடி. Trust

15 மீண்டும் பெட்டி ஒலி சூப்பர் Joy

16 நல்ல விஷயம் வாழ்த்துக்கள் Joy

17 டேய் சும்மா இருடா நாங்களே லோன எப்படி கட்டுறது தெரியலநீ வேற Anger

18 நண்பா அடுத்து சூரத் போவீங்களா. போறதா இருந்தா உங்களோட நானும் வ ரலாமா Ambiguous

19 நம்முடைய வாக்காளர்கள் கல்விக்கூடங்களில் இருக்கிறார்கள்.. சோர்வடைய வேண்டாம் தோழர்களே.. வெல்வான் விவசாயி.. 💪 Joy

20 எங்கடா? நாம்மேல்லாம் இந்தியன் என்று சொல்லுகிறீர்கள் சகமனிதனின் து ன்பத்தில் பங்கெடுக்கவில்லை என்றாலும் பரவாயில்லை Neutral

14206 தமிழ் மற்றும் சமஸ்கிருதம் Ambiguous

14207 எனக்கும் அதே டவுட் தான் சகோ Ambiguous

0.0

In [10]:

```
count = 0
for index, row in df_dev.iterrows():
    if not only_roman_chars(row['comment']):
        print(index, row['comment'], row['emotion'])
        print('\n\n')
        count += 1
print(1 - count/len(df_dev))
```

0 அருமை அற்புதம் பிரமாதம் நண்பரே வாழ்த்துக்கள் நண்பரே வாழ்க வளமுடன் நலமுடன் தொடரட்டும் தங்களது பணி மதுரை சிவசங்கரன் ஆர்டிஸ்ட் திருவல்லிக்கேணி சென்னை Joy

1 வேல்ராஜ் வேலையா தான் இருக்கும் Anticipation

2 அண்ணன் கிட்டுக்கு வாழ்த்துக்கள் 👍👍 Joy

3 ஆமா நானும் இதான் யோசித்தேன் 🤣🤣 Trust

4 மொத்த மக்களும் ஒன்னு சேர்ந்தாதான் இந்த அரசாங்கத்தை 🔥அடக்க முடியும் ✊😠 Anticipation

5 இவர் சொன்னது உன் மை Neutral

6 எந்த ஊர் சொல்லுங்க அக்கா Ambiguous

7 சூப்பர் வாழ்த்துக்கள் சகோ நீங்கள் நேரில் பார்க்கலாம் எங்கலாள்.பார்கா. முடியாது வாழ்த்துக்கள் Joy

8 அதுக்கு செத்து போக சொல்லுங்க செத்து போற 😥😥 ... இது மட்டும் முடியாது 😥😩😩😩😣 Sadness

9 எல்லாம் கடத்தல் தங்கமாக இருக்க வாய்ப்புள்ளது. இது எல்லாமே கொஞ்ச நாள்ல வெளி வரும்... கேரளாவும் இப்படித்தான் Disguist

10 25 வருடம் பேட்டரிக்குமா Ambiguous

11 தலைவரே நாலு முடிய புடிங்கி போடுங்க 😂😂 Joy

12 வேற லெவல் மிதுன் 🤣🤣🤣🤣🤣🤣🤣🤣🤣🤣🤣🤣 Neutral

13 @பாரி வேந்தன் என்னடா இது. என்ன சொல்ல வர. Ambiguous

14 ப ட ம் சு ப் ப ர் Joy

15 சரி இப்ப என்ன சொல்ல வர்ரீங்க குடும்பத்தோட தற்கொலை பன்னிக்கனுமா?? ஏங்கடா டேய் சனத்த வாழவும் விட மாட்டிக்கிறீங்க சாகவும் விட மாட்டிக்கிறீங்க? எப்படி பரவுதுன்னு சொன்னியே நைனா!! சி சீ ஐநா! எவன் கிருமியை உற்பத்தி ப ன்னிவிட்டான்னு கண்டுபிடிச்சியா??? Ambiguous

16 அஅப்படி இருக்க தருமண் எப்படி சொர்க்கலோம் சென்றான். சூதாட்டத்தில் ம னைவியை வைத்து ஆடியது ததவறுதானே. Ambiguous

17 . வயது17. சிறுமி சொல்றீங்க? இந்த கொலைகார கொரோனா இவுனுகல வாழ வச்சு நல்லவுங்களை கொண்டு போய் தொலையுது Disguist

18 ரசிகர்களின் தரமற்ற செயல் Neutral

19 தமிழ் வாழ்க..பெரியார் வாழ்க💥பெரியார் பெயரை சொன்ன உடனே கத றிட்டாங்களே Neutral

20 ஏதோ தலையில் இருக்கின்ற முடியும் சேர்ந்து போவதற்குரிய வழி தான் இது இ தையெல்லாம் விட்டுவிட்டு இருக்கின்ற முடியோடு வாழ்ந்துவிட்டு மரணித்தால் போ தும் Sadness

21 இவன் ஒரு ஒன்னம் நம்பர் பிராடு போக போக புரிந்து கொள்வீர்கள் 😂😂😊😊 Joy

22 யார் இந்த பாடலை 2021-ஆம் ஆண்டில் பார்க்கிறீர்கள் Ambiguous

23 சமையல் முடிஞ்சதும் குக்கருடன் திங்குறதுக்கு முதல் ஆளா தலைவர் இ ருக்கிறார் Neutral

ங்கே நரகத்தை அனுபவிக்கின்றார்கள் மற்றும் முதியவர்களுக்கு இல்லம் அதிகமாக இருக்கின்றது/ முதியோர் இல்லங்கள் அதிகமாக இருக்கின்றது திருநாட்டில்

4434 நாளை நல்லாட்சி மிண்டும் தொடரும் கவலை வேண்டாம் ..பணிய தொடற அட்வான்ஸ் வாழ்த்துக்கள்

4435 குஜராத்தில் அதிகம் இறப்பது ஜெயின் சமூகம்தான் இவர்கள்தான் சங்குக ளுக்கு அதிகம் வாக்களித்தவர்கள் உதவி செய்பவர்கள் சைவம் சாப்பிடக்கூடிய நோய் எதிர்ப்பு சக்தி குறைந்தவர்கள்

4436 பணம் இருந்தால் மனம் இருந்தால் குணம் இருக்கனும் இந்த மூன்றும் இ ருந்தால் மனிதன் தெய்வம் ஆகலாம்

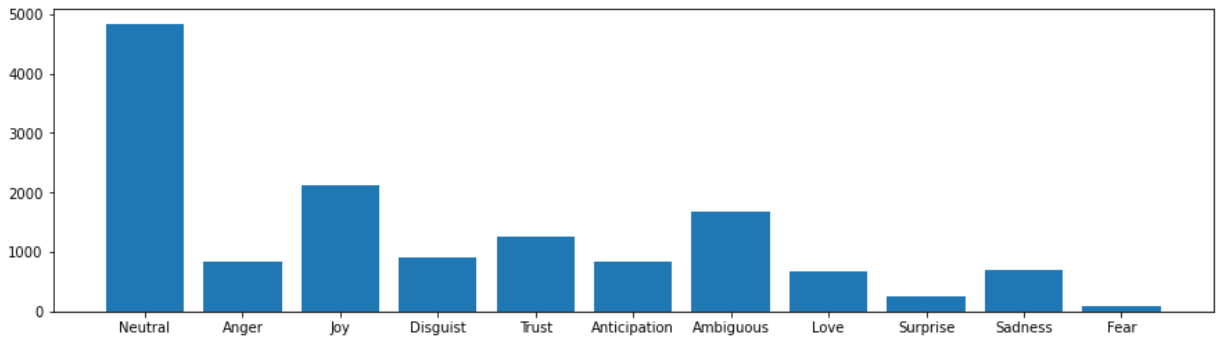
4437 இருப்பவர்கள் இடம் வாங்கி இல்லாதவர்களுக்கு கொடுப்பது தர்மம்...உ ண்மை...அன்பே சிவம் மருத்துவம் என்பது தொழில் அல்ல... அது கடவுளை விட மருத்துவரை நம்பும் நோயாளிகள் பக்தி நம்பிக்கை ...உழைப்புக்கு ஏற்ற ஊதியம் வழங்க வருவது அவரவர் விருப்பம்... நன்றி உள்ள மனிதர் வரவைத்து ...அசிங்க படுத்த வேண்டாம்...

4438 அருமை அண்ணா மிக்க மகிழ்ச்சி நன்றி

4439 ஓம் சக்தி அம்மா பேசுவதே இப்போது தான் முதல் முறையாக பார்க்கிறேன். தைப்பூச இருமுடி விழா தேதி என்ன?.

0.0

In [12]:
```
# Class distribution in training set
D = Counter(df_train['emotion'])
plt.rcParams["figure.figsize"] = (15,4)
plt.bar(range(len(D)), list(D.values()), align='center')
plt.xticks(range(len(D)), list(D.keys()))
plt.show()
```

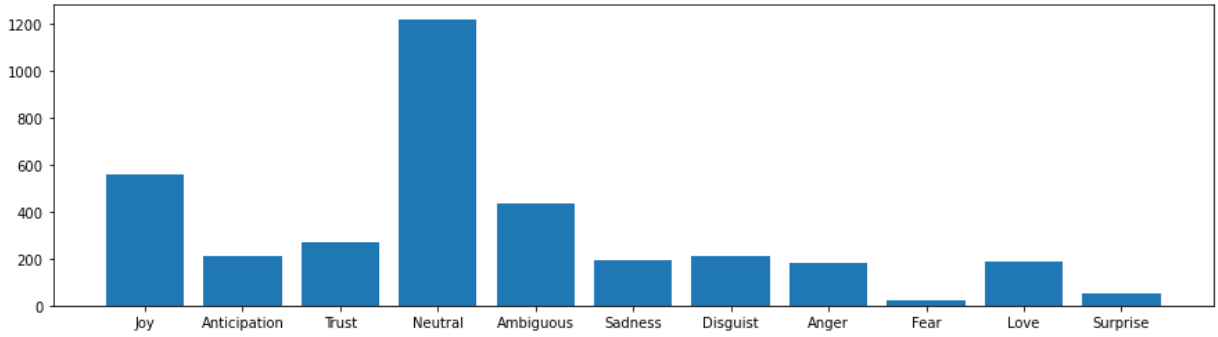


In [13]:
```
# Min, Max, Avg no of sentences per class in Training set
print('Min no. of sentences: ', min(D.values()))
print('Max no. of sentences: ', max(D.values()))
print('Avg no. of sentences: ', mean(D.values()))
print('Median of sentences: ', median(D.values()))
```

```
Min no. of sentences:  100
Max no. of sentences:  4841
Avg no. of sentences:  1291.6363636363637
Median of sentences:  834
```

In [14]:

```
# Class distribution in dev set
D = Counter(df_dev['emotion'])
plt.rcParams["figure.figsize"] = (15,4)
plt.bar(range(len(D)), list(D.values()), align='center')
plt.xticks(range(len(D)), list(D.keys()))
plt.show()
```



In [15]:

```
# Min, Max, Avg no of sentences per class in Dev set
print('Min no. of sentences: ', min(D.values()))
print('Max no. of sentences: ', max(D.values()))
print('Avg no. of sentences: ', mean(D.values()))
print('Median of sentences: ', median(D.values()))
```

```
Min no. of sentences:  23
Max no. of sentences:  1222
Avg no. of sentences:  322.90909090909093
Median of sentences:  210
```

In [16]:

```
def tokenize(s: str):
    return s.split()
```

In [17]:

```
# variation in length of sentences in train set
len_of_tokens = []
for index, row in df_train.iterrows():
    tokens = tokenize((row['comment']).lower())
    len_of_tokens.append(len(tokens))
print('Min no. of tokens: ', min(len_of_tokens))
print('Max no. of tokens: ', max(len_of_tokens))
print('Avg no. of tokens: ', mean(len_of_tokens))
print('Median of no. of tokens: ', median(len_of_tokens))
```

```
Min no. of tokens:  2
Max no. of tokens:  181
Avg no. of tokens:  9.713330518018019
Median of no. of tokens:  7.0
```

In [18]:

```
# variation in length of sentences in dev set
len_of_tokens = []
for index, row in df_dev.iterrows():
    tokens = tokenize((row['comment']).lower())
    len_of_tokens.append(len(tokens))
print('Min no. of tokens: ', min(len_of_tokens))
print('Max no. of tokens: ', max(len_of_tokens))
```

```
print('Avg no. of tokens: ', mean(len_of_tokens))
print('Median of no. of tokens: ', median(len_of_tokens))
```

```
Min no. of tokens:  2
Max no. of tokens:  189
Avg no. of tokens:  9.949042792792794
Median of no. of tokens:  7.0
```

In [19]:

```
# variation in length of sentences in test set
len_of_tokens = []
for index, row in df_test.iterrows():
    tokens = tokenize((row['comment']).lower())
    len_of_tokens.append(len(tokens))
print('Min no. of tokens: ', min(len_of_tokens))
print('Max no. of tokens: ', max(len_of_tokens))
print('Avg no. of tokens: ', mean(len_of_tokens))
print('Median of no. of tokens: ', median(len_of_tokens))
```

```
Min no. of tokens:  2
Max no. of tokens:  203
Avg no. of tokens:  9.705405405405406
Median of no. of tokens:  7.0
```

In [ ]:

M-BERT

Importing Libraries

In [3]:
```
import pandas as pd
from simpletransformers.classification import ClassificationModel, ClassificationArg
import sklearn
from sklearn.metrics import classification_report
from sklearn.metrics import f1_score
```

Loading Files From Dataset

In [4]:
```
df = pd.read_csv("ta-emotion10-train.csv",header=None,sep='\t')
df_eval = pd.read_csv("ta-emotion10-dev.csv",header=None,sep='\t')
df_test = pd.read_csv("task_a_test.csv",header=None,sep='\t')
```

In [5]:
```
df
```

Out[5]:

	0	1
0	Neutral	நாளைக்கு அரிசிக்கு இந்த நிலமை வந்தா 🥲
1	Anger	மானம் கேட்ட அன்புமணி
2	Neutral	தவறு இஸ்ரேல் இருக்காது இதை நான் கூறவில்லை ஹமாஸ...
3	Joy	கொங்கு நாட்டு சிங்கம் உன்மையும் நேர்மையும் உலை...
4	Neutral	இவர் யார்? ஒவ்வொரு வார்த்தையும் முன்னுக்கு பின...
...	...	...
14203	Trust	பெ மணியரசன் கூறுவதைஉணர்ந்து. செயலாற்றுவதேஇன்ற...
14204	Ambiguous	இன்னும் எத்தன நாள் வச்சி செய்வீங்க.
14205	Anticipation	அடுத்த ஏதோ தயார்பன்னிட்டான்
14206	Ambiguous	தமிழ் மற்றும் சமஸ்கிருதம்
14207	Ambiguous	எனக்கும் அதே டவுட் தான் சகோ

14208 rows × 2 columns

In [6]:
```
df_eval.rename(columns={0:'Labels',1:'Text'},inplace=True)
df_eval = df_eval[['Text','Labels']]
df.rename(columns={0:'Labels',1:'Text'},inplace=True)
df = df[['Text','Labels']]
df_test.rename(columns={0:'Labels',1:'Text'},inplace=True)
df_test = df_test[['Text','Labels']]
```

In [7]:
```
num_labels = len(df['Labels'].unique())
keys = list(df['Labels'].unique())
values = list(range(0, num_labels))
label_dict = dict(zip(keys,values))
```

```
df['Labels'] = df['Labels'].apply(lambda x:label_dict[x])
df_eval['Labels'] = df_eval['Labels'].apply(lambda x:label_dict[x])
df_test['Labels'] = df_test['Labels'].apply(lambda x:label_dict[x])
num_labels
```

Out[7]: 11

Balancing the imbalanced dataset

In [8]:
```
def oversample(df):
        classes = df['Labels'].value_counts().to_dict()
        most = max(classes.values())
        classes_list = []
        for key in classes:
            classes_list.append(df[df['Labels'] == key])
        classes_sample = []
        for i in range(1,len(classes_list)):
            classes_sample.append(classes_list[i].sample(most, replace=True))
        df_maybe = pd.concat(classes_sample)
        final_df = pd.concat([df_maybe,classes_list[0]], axis=0)
        final_df = final_df.reset_index(drop=True)
        return pd.DataFrame({'Text': final_df['Text'].tolist(), 'Labels': final_df['
```

In [9]:
```
def over_under_sample(df):
  unq_labels = list(set(df['Labels'].tolist()))
  texts = df['Text'].tolist()
  labels = df['Labels'].tolist()
  data_dict = dict()

  for l in unq_labels:
    data_dict[l] = []

  for i in range(len(texts)):
    data_dict[labels[i]].append(texts[i])

  req_len = len(labels)//len(unq_labels)

  for label in data_dict:
    if len(data_dict[label]) > req_len:
      data_dict[label] = data_dict[label][:req_len]

    new_texts = []

    new_labels = []
    for l in data_dict:
      new_texts += data_dict[l]
      new_labels += [l]*len(data_dict[l])
    return oversample(pd.DataFrame({'Text': new_texts, 'Labels': new_labels}))
```

In [10]:
```
df = over_under_sample(df)
```

Model Training

In [11]:
```
model_args = ClassificationArgs()
```

In [12]:

```
model_args.overwrite_output_dir=True
model_args.eval_batch_size=8
model_args.train_batch_size=8
model_args.learning_rate=4e-5
```

In [13]:

```
model = ClassificationModel(
    'bert',
    'bert-base-multilingual-cased',
    num_labels=11,
    args=model_args,
    tokenizer_type="bert",
    tokenizer_name='bert-base-multilingual-cased'
)
```

```
Some weights of the model checkpoint at bert-base-multilingual-cased were not used w
hen initializing BertForSequenceClassification: ['cls.seq_relationship.bias', 'cls.p
redictions.transform.dense.weight', 'cls.predictions.transform.LayerNorm.bias', 'cl
s.predictions.decoder.weight', 'cls.predictions.transform.LayerNorm.weight', 'cls.pr
edictions.bias', 'cls.predictions.transform.dense.bias', 'cls.seq_relationship.weigh
t']
- This IS expected if you are initializing BertForSequenceClassification from the ch
eckpoint of a model trained on another task or with another architecture (e.g. initi
alizing a BertForSequenceClassification model from a BertForPreTraining model).
- This IS NOT expected if you are initializing BertForSequenceClassification from th
e checkpoint of a model that you expect to be exactly identical (initializing a Bert
ForSequenceClassification model from a BertForSequenceClassification model).
Some weights of BertForSequenceClassification were not initialized from the model ch
eckpoint at bert-base-multilingual-cased and are newly initialized: ['classifier.bia
s', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it for
predictions and inference.
```

In [14]:

```
for i in range(0,4):
    !rm -rf /kaggle/working/outputs
    model.train_model(df,eval_data=df_eval,acc=sklearn.metrics.classification_report
    result, model_outputs, preds_list = model.eval_model(df_test,acc=sklearn.metrics
    for j in result.values():
        print(j)
```

```
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:586: UserWarning: Dataframe headers not specified. Falling back to usin
g column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."
```

```
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:922: FutureWarning: Non-finite norm encountered in torch.nn.utils.clip_
grad_norm_; continuing anyway. Note that the default behavior will change in a futur
e release to error out if a non-finite total norm is encountered. At that point, set
ting error_if_nonfinite=false will be required to retain the old behavior.
  model.parameters(), args.max_grad_norm
/opt/conda/lib/python3.7/site-packages/torch/optim/lr_scheduler.py:134: UserWarning:
Detected call of `lr_scheduler.step()` before `optimizer.step()`. In PyTorch 1.1.0 a
nd later, you should call them in the opposite order: `optimizer.step()` before `lr_
scheduler.step()`.  Failure to do this will result in PyTorch skipping the first val
ue of the learning rate schedule. See more details at https://pytorch.org/docs/stabl
e/optim.html#how-to-adjust-learning-rate
  "https://pytorch.org/docs/stable/optim.html#how-to-adjust-learning-rate", UserWarn
ing)
```

```
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:1427: UserWarning: Dataframe headers not specified. Falling back to usi
ng column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."


0.2314039642759547
              precision    recall  f1-score   support

         0.0       0.53      0.13      0.21      1538
         1.0       0.17      0.33      0.22       244
         2.0       0.60      0.58      0.59       702
         3.0       0.14      0.19      0.16       277
         4.0       0.21      0.24      0.22       377
         5.0       0.24      0.43      0.31       271
         6.0       0.48      0.50      0.49       500
         7.0       0.15      0.30      0.20       196
         8.0       0.02      0.07      0.03        61
         9.0       0.15      0.28      0.20       241
        10.0       0.15      0.24      0.19        33

    accuracy                           0.30      4440
   macro avg       0.26      0.30      0.26      4440
weighted avg       0.40      0.30      0.30      4440

2.0023343197934262
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:586: UserWarning: Dataframe headers not specified. Falling back to usin
g column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."



/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:922: FutureWarning: Non-finite norm encountered in torch.nn.utils.clip_
grad_norm_; continuing anyway. Note that the default behavior will change in a futur
e release to error out if a non-finite total norm is encountered. At that point, set
ting error_if_nonfinite=false will be required to retain the old behavior.
  model.parameters(), args.max_grad_norm
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:1427: UserWarning: Dataframe headers not specified. Falling back to usi
ng column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."


0.2542749579578216
              precision    recall  f1-score   support

         0.0       0.52      0.24      0.32      1538
         1.0       0.18      0.27      0.22       244
         2.0       0.57      0.59      0.58       702
         3.0       0.14      0.19      0.16       277
         4.0       0.26      0.28      0.27       377
         5.0       0.24      0.39      0.30       271
         6.0       0.46      0.54      0.50       500
         7.0       0.17      0.27      0.20       196
         8.0       0.03      0.07      0.04        61
         9.0       0.21      0.34      0.26       241
        10.0       0.25      0.21      0.23        33

    accuracy                           0.34      4440
   macro avg       0.28      0.31      0.28      4440
weighted avg       0.40      0.34      0.35      4440

2.1303844348804373
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:586: UserWarning: Dataframe headers not specified. Falling back to usin
```

```
g column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."



/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:922: FutureWarning: Non-finite norm encountered in torch.nn.utils.clip_
grad_norm_; continuing anyway. Note that the default behavior will change in a futur
e release to error out if a non-finite total norm is encountered. At that point, set
ting error_if_nonfinite=false will be required to retain the old behavior.
  model.parameters(), args.max_grad_norm
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:1427: UserWarning: Dataframe headers not specified. Falling back to usi
ng column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."


0.2527226691495714
              precision    recall  f1-score   support

         0.0       0.52      0.26      0.34      1538
         1.0       0.17      0.21      0.19       244
         2.0       0.56      0.55      0.55       702
         3.0       0.15      0.23      0.18       277
         4.0       0.25      0.34      0.29       377
         5.0       0.27      0.40      0.32       271
         6.0       0.49      0.53      0.51       500
         7.0       0.15      0.24      0.19       196
         8.0       0.03      0.05      0.04        61
         9.0       0.22      0.32      0.26       241
        10.0       0.29      0.18      0.22        33

    accuracy                           0.35      4440
   macro avg       0.28      0.30      0.28      4440
weighted avg       0.40      0.35      0.35      4440

2.380400817888277
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:586: UserWarning: Dataframe headers not specified. Falling back to usin
g column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."



/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:922: FutureWarning: Non-finite norm encountered in torch.nn.utils.clip_
grad_norm_; continuing anyway. Note that the default behavior will change in a futur
e release to error out if a non-finite total norm is encountered. At that point, set
ting error_if_nonfinite=false will be required to retain the old behavior.
  model.parameters(), args.max_grad_norm
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:1427: UserWarning: Dataframe headers not specified. Falling back to usi
ng column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."


0.255306444322786
              precision    recall  f1-score   support

         0.0       0.53      0.29      0.37      1538
         1.0       0.19      0.22      0.20       244
         2.0       0.52      0.58      0.55       702
         3.0       0.15      0.17      0.16       277
         4.0       0.23      0.35      0.28       377
         5.0       0.28      0.35      0.31       271
         6.0       0.46      0.53      0.49       500
```

```
         7.0       0.15      0.24      0.19       196
         8.0       0.04      0.03      0.03        61
         9.0       0.25      0.33      0.29       241
        10.0       0.15      0.15      0.15        33

    accuracy                           0.36      4440
   macro avg       0.27      0.30      0.27      4440
weighted avg       0.39      0.36      0.36      4440

2.6724335396612013
```

In [15]:
```
predictions, raw_outputs = model.predict(df_test['Text'].to_list())
```

In [16]:
```
df_final = df_test.copy()
reverse_label_dict = {v:u for u,v in label_dict.items()}
reverse_label_dict
df_final['Predicted_Labels'] = predictions
df_final['Predicted_Labels'] = df_final['Predicted_Labels'].apply(lambda x:reverse_l
df_final['Labels'] = df_final['Labels'].apply(lambda x:reverse_label_dict[x])
df_final['pid'] = df_final.index
df_final = df_final[['pid','Predicted_Labels','Labels']]
```

In [17]:
```
df_final
```

Out[17]:

	pid	Predicted_Labels	Labels
0	0	Ambiguous	Ambiguous
1	1	Joy	Disguist
2	2	Anger	Disguist
3	3	Ambiguous	Ambiguous
4	4	Joy	Joy
...	...	...	...
4435	4435	Sadness	Neutral
4436	4436	Trust	Trust
4437	4437	Trust	Anticipation
4438	4438	Joy	Joy
4439	4439	Ambiguous	Ambiguous

4440 rows × 3 columns

In [18]:
```
score = f1_score(df_final['Labels'],df_final['Predicted_Labels'],average='macro')
print("The macro average f1 score is:" + str(score))
```

```
The macro average f1 score is:0.27459121367417355
```

In [ ]:

XLM-R

Importing Libraries

In [3]:
```
import pandas as pd
from simpletransformers.classification import ClassificationModel, ClassificationArg
import sklearn
from sklearn.metrics import classification_report
from sklearn.metrics import f1_score
```

Loading Files From Dataset

In [4]:
```
df = pd.read_csv("ta-emotion10-train.csv",header=None,sep='\t')
df_eval = pd.read_csv("ta-emotion10-dev.csv",header=None,sep='\t')
df_test = pd.read_csv("task_a_test.csv",header=None,sep='\t')
```

In [5]:
```
df
```

Out[5]:

	0	1
0	Neutral	நாளைக்கு அரிசிக்கு இந்த நிலமை வந்தா 🥲
1	Anger	மானம் கேட்ட அன்புமணி
2	Neutral	தவறு இஸ்ரேல் இருக்காது இதை நான் கூறவில்லை ஹமாஸ...
3	Joy	கொங்கு நாட்டு சிங்கம் உன்மையும் நேர்மையும் உலை...
4	Neutral	இவர் யார்? ஒவ்வொரு வார்த்தையும் முன்னுக்கு பின...
...	...	...
14203	Trust	பெ மணியரசன் கூறுவதைஉணர்ந்து. செயலாற்றுவதேஇன்ற...
14204	Ambiguous	இன்னும் எத்தன நாள் வச்சி செய்வீங்க.
14205	Anticipation	அடுத்த ஏதோ தயார்பன்னிட்டான்
14206	Ambiguous	தமிழ் மற்றும் சமஸ்கிருதம்
14207	Ambiguous	எனக்கும் அதே டவுட் தான் சகோ

14208 rows × 2 columns

In [6]:
```
df_eval.rename(columns={0:'Labels',1:'Text'},inplace=True)
df_eval = df_eval[['Text','Labels']]
df.rename(columns={0:'Labels',1:'Text'},inplace=True)
df = df[['Text','Labels']]
df_test.rename(columns={0:'Labels',1:'Text'},inplace=True)
df_test = df_test[['Text','Labels']]
```

In [7]:
```
num_labels = len(df['Labels'].unique())
keys = list(df['Labels'].unique())
values = list(range(0, num_labels))
label_dict = dict(zip(keys,values))
```

```
df['Labels'] = df['Labels'].apply(lambda x:label_dict[x])
df_eval['Labels'] = df_eval['Labels'].apply(lambda x:label_dict[x])
df_test['Labels'] = df_test['Labels'].apply(lambda x:label_dict[x])
num_labels
```

Out[7]: 11

Balancing the imbalanced dataset

In [8]:
```
def oversample(df):
        classes = df['Labels'].value_counts().to_dict()
        most = max(classes.values())
        classes_list = []
        for key in classes:
            classes_list.append(df[df['Labels'] == key])
        classes_sample = []
        for i in range(1,len(classes_list)):
            classes_sample.append(classes_list[i].sample(most, replace=True))
        df_maybe = pd.concat(classes_sample)
        final_df = pd.concat([df_maybe,classes_list[0]], axis=0)
        final_df = final_df.reset_index(drop=True)
        return pd.DataFrame({'Text': final_df['Text'].tolist(), 'Labels': final_df['
```

In [9]:
```
def over_under_sample(df):
  unq_labels = list(set(df['Labels'].tolist()))
  texts = df['Text'].tolist()
  labels = df['Labels'].tolist()
  data_dict = dict()

  for l in unq_labels:
    data_dict[l] = []

  for i in range(len(texts)):
    data_dict[labels[i]].append(texts[i])

  req_len = len(labels)//len(unq_labels)

  for label in data_dict:
    if len(data_dict[label]) > req_len:
      data_dict[label] = data_dict[label][:req_len]

    new_texts = []

    new_labels = []
    for l in data_dict:
      new_texts += data_dict[l]
      new_labels += [l]*len(data_dict[l])
    return oversample(pd.DataFrame({'Text': new_texts, 'Labels': new_labels}))
```

In [10]:
```
df = over_under_sample(df)
```

Model Training

In [11]:
```
model_args = ClassificationArgs()
```

In [12]:

```
model_args.overwrite_output_dir=True
model_args.eval_batch_size=8
model_args.train_batch_size=8
model_args.learning_rate=4e-5
model_args.use_multiprocessing=False
```

In [13]:

```
model = ClassificationModel(
    'xlmroberta',
    'xlm-roberta-base',
    num_labels=11,
    args=model_args,
    tokenizer_type="xlmroberta",
    tokenizer_name='xlm-roberta-base'
)
```

```
Some weights of the model checkpoint at xlm-roberta-base were not used when initiali
zing XLMRobertaForSequenceClassification: ['lm_head.dense.weight', 'lm_head.dense.bi
as', 'lm_head.decoder.weight', 'lm_head.layer_norm.bias', 'lm_head.layer_norm.weigh
t', 'roberta.pooler.dense.weight', 'roberta.pooler.dense.bias', 'lm_head.bias']
- This IS expected if you are initializing XLMRobertaForSequenceClassification from
the checkpoint of a model trained on another task or with another architecture (e.g.
initializing a BertForSequenceClassification model from a BertForPreTraining model).
- This IS NOT expected if you are initializing XLMRobertaForSequenceClassification f
rom the checkpoint of a model that you expect to be exactly identical (initializing
a BertForSequenceClassification model from a BertForSequenceClassification model).
Some weights of XLMRobertaForSequenceClassification were not initialized from the mo
del checkpoint at xlm-roberta-base and are newly initialized: ['classifier.out_proj.
bias', 'classifier.dense.bias', 'classifier.dense.weight', 'classifier.out_proj.weig
ht']
You should probably TRAIN this model on a down-stream task to be able to use it for
predictions and inference.


/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:460: UserWarning: use_multiprocessing automatically disabled as xlmrobe
rta fails when using multiprocessing for feature conversion.
  f"use_multiprocessing automatically disabled as {model_type}"
```

In [14]:

```
for i in range(0,5):
    !rm -rf /kaggle/working/outputs
    model.train_model(df,eval_data=df_eval,acc=sklearn.metrics.classification_report
    result, model_outputs, preds_list = model.eval_model(df_test,acc=sklearn.metrics
    for j in result.values():
        print(j)
```

```
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:586: UserWarning: Dataframe headers not specified. Falling back to usin
g column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."


/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:922: FutureWarning: Non-finite norm encountered in torch.nn.utils.clip_
grad_norm_; continuing anyway. Note that the default behavior will change in a futur
e release to error out if a non-finite total norm is encountered. At that point, set
ting error_if_nonfinite=false will be required to retain the old behavior.
  model.parameters(), args.max_grad_norm
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:1427: UserWarning: Dataframe headers not specified. Falling back to usi
ng column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."
```

```
huggingface/tokenizers: The current process just got forked, after parallelism has a
lready been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
        - Avoid using `tokenizers` before the fork if possible
        - Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | fal
se)


/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1308: Unde
finedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in la
bels with no predicted samples. Use `zero_division` parameter to control this behavi
or.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1308: Unde
finedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in la
bels with no predicted samples. Use `zero_division` parameter to control this behavi
or.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1308: Unde
finedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in la
bels with no predicted samples. Use `zero_division` parameter to control this behavi
or.
  _warn_prf(average, modifier, msg_start, len(result))
0.0
              precision    recall  f1-score   support

         0.0       0.00      0.00      0.00      1538
         1.0       0.00      0.00      0.00       244
         2.0       0.00      0.00      0.00       702
         3.0       0.00      0.00      0.00       277
         4.0       0.00      0.00      0.00       377
         5.0       0.00      0.00      0.00       271
         6.0       0.00      0.00      0.00       500
         7.0       0.00      0.00      0.00       196
         8.0       0.00      0.00      0.00        61
         9.0       0.00      0.00      0.00       241
        10.0       0.01      1.00      0.01        33

    accuracy                           0.01      4440
   macro avg       0.00      0.09      0.00      4440
weighted avg       0.00      0.01      0.00      4440

2.3956859691722974
huggingface/tokenizers: The current process just got forked, after parallelism has a
lready been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
        - Avoid using `tokenizers` before the fork if possible
        - Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | fal
se)
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:586: UserWarning: Dataframe headers not specified. Falling back to usin
g column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."


/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:922: FutureWarning: Non-finite norm encountered in torch.nn.utils.clip_
grad_norm_; continuing anyway. Note that the default behavior will change in a futur
e release to error out if a non-finite total norm is encountered. At that point, set
ting error_if_nonfinite=false will be required to retain the old behavior.
  model.parameters(), args.max_grad_norm
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:1427: UserWarning: Dataframe headers not specified. Falling back to usi
ng column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."
huggingface/tokenizers: The current process just got forked, after parallelism has a
lready been used. Disabling parallelism to avoid deadlocks...
```

```
To disable this warning, you can either:
        - Avoid using `tokenizers` before the fork if possible
        - Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | fal
se)


/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1308: Unde
finedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in la
bels with no predicted samples. Use `zero_division` parameter to control this behavi
or.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1308: Unde
finedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in la
bels with no predicted samples. Use `zero_division` parameter to control this behavi
or.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1308: Unde
finedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in la
bels with no predicted samples. Use `zero_division` parameter to control this behavi
or.
  _warn_prf(average, modifier, msg_start, len(result))
0.0
              precision    recall  f1-score   support

         0.0       0.00      0.00      0.00      1538
         1.0       0.00      0.00      0.00       244
         2.0       0.00      0.00      0.00       702
         3.0       0.00      0.00      0.00       277
         4.0       0.00      0.00      0.00       377
         5.0       0.00      0.00      0.00       271
         6.0       0.00      0.00      0.00       500
         7.0       0.00      0.00      0.00       196
         8.0       0.01      1.00      0.03        61
         9.0       0.00      0.00      0.00       241
        10.0       0.00      0.00      0.00        33

    accuracy                           0.01      4440
   macro avg       0.00      0.09      0.00      4440
weighted avg       0.00      0.01      0.00      4440

2.399291772240991
huggingface/tokenizers: The current process just got forked, after parallelism has a
lready been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
        - Avoid using `tokenizers` before the fork if possible
        - Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | fal
se)
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:586: UserWarning: Dataframe headers not specified. Falling back to usin
g column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."


/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:922: FutureWarning: Non-finite norm encountered in torch.nn.utils.clip_
grad_norm_; continuing anyway. Note that the default behavior will change in a futur
e release to error out if a non-finite total norm is encountered. At that point, set
ting error_if_nonfinite=false will be required to retain the old behavior.
  model.parameters(), args.max_grad_norm
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:1427: UserWarning: Dataframe headers not specified. Falling back to usi
ng column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."
huggingface/tokenizers: The current process just got forked, after parallelism has a
lready been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
        - Avoid using `tokenizers` before the fork if possible
```

```
        - Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | fal
se)


/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1308: Unde
finedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in la
bels with no predicted samples. Use `zero_division` parameter to control this behavi
or.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1308: Unde
finedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in la
bels with no predicted samples. Use `zero_division` parameter to control this behavi
or.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1308: Unde
finedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in la
bels with no predicted samples. Use `zero_division` parameter to control this behavi
or.
  _warn_prf(average, modifier, msg_start, len(result))
0.0
              precision    recall  f1-score   support

         0.0       0.00      0.00      0.00      1538
         1.0       0.00      0.00      0.00       244
         2.0       0.00      0.00      0.00       702
         3.0       0.00      0.00      0.00       277
         4.0       0.08      1.00      0.16       377
         5.0       0.00      0.00      0.00       271
         6.0       0.00      0.00      0.00       500
         7.0       0.00      0.00      0.00       196
         8.0       0.00      0.00      0.00        61
         9.0       0.00      0.00      0.00       241
        10.0       0.00      0.00      0.00        33

    accuracy                           0.08      4440
   macro avg       0.01      0.09      0.01      4440
weighted avg       0.01      0.08      0.01      4440

2.382073919622748
huggingface/tokenizers: The current process just got forked, after parallelism has a
lready been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
        - Avoid using `tokenizers` before the fork if possible
        - Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | fal
se)
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:586: UserWarning: Dataframe headers not specified. Falling back to usin
g column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."


/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:922: FutureWarning: Non-finite norm encountered in torch.nn.utils.clip_
grad_norm_; continuing anyway. Note that the default behavior will change in a futur
e release to error out if a non-finite total norm is encountered. At that point, set
ting error_if_nonfinite=false will be required to retain the old behavior.
  model.parameters(), args.max_grad_norm
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:1427: UserWarning: Dataframe headers not specified. Falling back to usi
ng column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."
huggingface/tokenizers: The current process just got forked, after parallelism has a
lready been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
        - Avoid using `tokenizers` before the fork if possible
        - Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | fal
se)
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1308: Unde
finedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in la
bels with no predicted samples. Use `zero_division` parameter to control this behavi
or.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1308: Unde
finedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in la
bels with no predicted samples. Use `zero_division` parameter to control this behavi
or.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1308: Unde
finedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in la
bels with no predicted samples. Use `zero_division` parameter to control this behavi
or.
  _warn_prf(average, modifier, msg_start, len(result))
0.0
              precision    recall  f1-score   support

         0.0       0.00      0.00      0.00      1538
         1.0       0.00      0.00      0.00       244
         2.0       0.00      0.00      0.00       702
         3.0       0.00      0.00      0.00       277
         4.0       0.00      0.00      0.00       377
         5.0       0.06      1.00      0.12       271
         6.0       0.00      0.00      0.00       500
         7.0       0.00      0.00      0.00       196
         8.0       0.00      0.00      0.00        61
         9.0       0.00      0.00      0.00       241
        10.0       0.00      0.00      0.00        33

    accuracy                           0.06      4440
   macro avg       0.01      0.09      0.01      4440
weighted avg       0.00      0.06      0.01      4440

2.4047992328265764
huggingface/tokenizers: The current process just got forked, after parallelism has a
lready been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
        - Avoid using `tokenizers` before the fork if possible
        - Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | fal
se)
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:586: UserWarning: Dataframe headers not specified. Falling back to usin
g column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."


/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:922: FutureWarning: Non-finite norm encountered in torch.nn.utils.clip_
grad_norm_; continuing anyway. Note that the default behavior will change in a futur
e release to error out if a non-finite total norm is encountered. At that point, set
ting error_if_nonfinite=false will be required to retain the old behavior.
  model.parameters(), args.max_grad_norm
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:1427: UserWarning: Dataframe headers not specified. Falling back to usi
ng column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."
huggingface/tokenizers: The current process just got forked, after parallelism has a
lready been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
        - Avoid using `tokenizers` before the fork if possible
        - Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | fal
se)


0.0
```

```
              precision    recall  f1-score   support

         0.0       0.00      0.00      0.00      1538
         1.0       0.00      0.00      0.00       244
         2.0       0.00      0.00      0.00       702
         3.0       0.00      0.00      0.00       277
         4.0       0.08      1.00      0.16       377
         5.0       0.00      0.00      0.00       271
         6.0       0.00      0.00      0.00       500
         7.0       0.00      0.00      0.00       196
         8.0       0.00      0.00      0.00        61
         9.0       0.00      0.00      0.00       241
        10.0       0.00      0.00      0.00        33

    accuracy                           0.08      4440
   macro avg       0.01      0.09      0.01      4440
weighted avg       0.01      0.08      0.01      4440

2.400269214527027
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1308: Unde
finedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in la
bels with no predicted samples. Use `zero_division` parameter to control this behavi
or.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1308: Unde
finedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in la
bels with no predicted samples. Use `zero_division` parameter to control this behavi
or.
  _warn_prf(average, modifier, msg_start, len(result))
/opt/conda/lib/python3.7/site-packages/sklearn/metrics/_classification.py:1308: Unde
finedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in la
bels with no predicted samples. Use `zero_division` parameter to control this behavi
or.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [15]:

```
predictions, raw_outputs = model.predict(df_test['Text'].to_list())
```

```
huggingface/tokenizers: The current process just got forked, after parallelism has a
lready been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
        - Avoid using `tokenizers` before the fork if possible
        - Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | fal
se)
```

In [16]:

```
df_final = df_test.copy()
reverse_label_dict = {v:u for u,v in label_dict.items()}
reverse_label_dict
df_final['Predicted_Labels'] = predictions
df_final['Predicted_Labels'] = df_final['Predicted_Labels'].apply(lambda x:reverse_l
df_final['Labels'] = df_final['Labels'].apply(lambda x:reverse_label_dict[x])
df_final['pid'] = df_final.index
df_final = df_final[['pid','Predicted_Labels','Labels']]
```

In [17]:

```
df_final
```

Out[17]:

	pid	Predicted_Labels	Labels
0	0	Trust	Ambiguous
1	1	Trust	Disguist
2	2	Trust	Disguist

	pid	Predicted_Labels	Labels
3	3	Trust	Ambiguous
4	4	Trust	Joy
...	...	...	...
4435	4435	Trust	Neutral
4436	4436	Trust	Trust
4437	4437	Trust	Anticipation
4438	4438	Trust	Joy
4439	4439	Trust	Ambiguous

4440 rows × 3 columns

In [18]:
```
score = f1_score(df_final['Labels'],df_final['Predicted_Labels'],average='macro')
print("The macro average f1 score is:" + str(score))
```

```
The macro average f1 score is:0.014229905448506235
```

In [ ]:

MuRIL

Importing Libraries

In [3]:
```
import pandas as pd
from simpletransformers.classification import ClassificationModel, ClassificationArg
import sklearn
from sklearn.metrics import classification_report
from sklearn.metrics import f1_score
```

Loading Files From Dataset

In [4]:
```
df = pd.read_csv("ta-emotion10-train.csv",header=None,sep='\t')
df_eval = pd.read_csv("ta-emotion10-dev.csv",header=None,sep='\t')
df_test = pd.read_csv("task_a_test.csv",header=None,sep='\t')
```

In [5]:
```
df_eval.rename(columns={0:'Labels',1:'Text'},inplace=True)
df_eval = df_eval[['Text','Labels']]
df.rename(columns={0:'Labels',1:'Text'},inplace=True)
df = df[['Text','Labels']]
df_test.rename(columns={0:'Labels',1:'Text'},inplace=True)
df_test = df_test[['Text','Labels']]
```

In [6]:
```
num_labels = len(df['Labels'].unique())
keys = list(df['Labels'].unique())
values = list(range(0, num_labels))
label_dict = dict(zip(keys,values))
df['Labels'] = df['Labels'].apply(lambda x:label_dict[x])
df_eval['Labels'] = df_eval['Labels'].apply(lambda x:label_dict[x])
df_test['Labels'] = df_test['Labels'].apply(lambda x:label_dict[x])
num_labels
```

Out[6]: 11

Balancing the imbalanced dataset

In [7]:
```
def oversample(df):
        classes = df['Labels'].value_counts().to_dict()
        most = max(classes.values())
        classes_list = []
        for key in classes:
            classes_list.append(df[df['Labels'] == key])
        classes_sample = []
        for i in range(1,len(classes_list)):
            classes_sample.append(classes_list[i].sample(most, replace=True))
        df_maybe = pd.concat(classes_sample)
        final_df = pd.concat([df_maybe,classes_list[0]], axis=0)
        final_df = final_df.reset_index(drop=True)
        return pd.DataFrame({'Text': final_df['Text'].tolist(), 'Labels': final_df['
```

In [8]:
```
def over_under_sample(df):
```

```
  unq_labels = list(set(df['Labels'].tolist()))
  texts = df['Text'].tolist()
  labels = df['Labels'].tolist()
  data_dict = dict()

  for l in unq_labels:
    data_dict[l] = []

  for i in range(len(texts)):
    data_dict[labels[i]].append(texts[i])

  req_len = len(labels)//len(unq_labels)

  for label in data_dict:
    if len(data_dict[label]) > req_len:
      data_dict[label] = data_dict[label][:req_len]

    new_texts = []

    new_labels = []
    for l in data_dict:
      new_texts += data_dict[l]
      new_labels += [l]*len(data_dict[l])
    return oversample(pd.DataFrame({'Text': new_texts, 'Labels': new_labels}))
```

In [9]:

```
df = over_under_sample(df)
```

Model Training

In [10]:

```
model_args = ClassificationArgs()
```

In [11]:

```
model_args.overwrite_output_dir=True
model_args.eval_batch_size=8
model_args.train_batch_size=8
model_args.learning_rate=4e-5
```

In [12]:

```
model = ClassificationModel(
    'bert',
    'google/muril-base-cased',
    num_labels=11,
    args=model_args,
    tokenizer_type="bert",
    tokenizer_name='google/muril-base-cased'
)
```

```
Some weights of the model checkpoint at google/muril-base-cased were not used when i
nitializing BertForSequenceClassification: ['cls.predictions.decoder.bias', 'cls.seq
_relationship.weight', 'cls.predictions.bias', 'cls.predictions.transform.dense.weig
ht', 'cls.seq_relationship.bias', 'cls.predictions.transform.dense.bias', 'cls.predi
ctions.transform.LayerNorm.bias', 'cls.predictions.transform.LayerNorm.weight', 'cl
s.predictions.decoder.weight']
- This IS expected if you are initializing BertForSequenceClassification from the ch
eckpoint of a model trained on another task or with another architecture (e.g. initi
alizing a BertForSequenceClassification model from a BertForPreTraining model).
- This IS NOT expected if you are initializing BertForSequenceClassification from th
e checkpoint of a model that you expect to be exactly identical (initializing a Bert
ForSequenceClassification model from a BertForSequenceClassification model).
```

```
Some weights of BertForSequenceClassification were not initialized from the model ch
eckpoint at google/muril-base-cased and are newly initialized: ['classifier.weight',
'classifier.bias']
You should probably TRAIN this model on a down-stream task to be able to use it for
predictions and inference.
```

In [13]:

```
for i in range(0,3):
    !rm -rf /kaggle/working/outputs
    model.train_model(df,eval_data=df_eval,acc=sklearn.metrics.classification_report
    result, model_outputs, preds_list = model.eval_model(df_test,acc=sklearn.metrics
    for j in result.values():
        print(j)
```

```
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:586: UserWarning: Dataframe headers not specified. Falling back to usin
g column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."



/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:922: FutureWarning: Non-finite norm encountered in torch.nn.utils.clip_
grad_norm_; continuing anyway. Note that the default behavior will change in a futur
e release to error out if a non-finite total norm is encountered. At that point, set
ting error_if_nonfinite=false will be required to retain the old behavior.
  model.parameters(), args.max_grad_norm
/opt/conda/lib/python3.7/site-packages/simpletransformers/classification/classificat
ion_model.py:1427: UserWarning: Dataframe headers not specified. Falling back to usi
ng column 0 as text and column 1 as labels.
  "Dataframe headers not specified. Falling back to using column 0 as text and colum
n 1 as labels."


0.17738863009987985
              precision    recall  f1-score   support

         0.0       0.60      0.00      0.00      1538
         1.0       0.18      0.38      0.25       244
         2.0       0.66      0.07      0.13       702
         3.0       0.13      0.18      0.15       277
         4.0       0.24      0.11      0.15       377
         5.0       0.22      0.58      0.32       271
         6.0       0.49      0.61      0.54       500
         7.0       0.12      0.62      0.20       196
         8.0       0.03      0.36      0.06        61
         9.0       0.37      0.28      0.32       241
        10.0       0.09      0.33      0.14        33

    accuracy                           0.21      4440
   macro avg       0.29      0.32      0.21      4440
weighted avg       0.45      0.21      0.17      4440

2.0985558312218466



0.27101908014819803
              precision    recall  f1-score   support

         0.0       0.52      0.16      0.24      1538
         1.0       0.21      0.33      0.25       244
         2.0       0.65      0.50      0.56       702
         3.0       0.18      0.25      0.21       277
         4.0       0.26      0.40      0.31       377
```

```
         5.0       0.26      0.49      0.34       271
         6.0       0.49      0.59      0.54       500
         7.0       0.15      0.36      0.21       196
         8.0       0.05      0.11      0.07        61
         9.0       0.30      0.44      0.36       241
        10.0       0.32      0.27      0.30        33

    accuracy                           0.34      4440
   macro avg       0.31      0.35      0.31      4440
weighted avg       0.42      0.34      0.34      4440

2.1036428021955063



0.28114539933074395
              precision    recall  f1-score   support

         0.0       0.54      0.24      0.33      1538
         1.0       0.21      0.31      0.25       244
         2.0       0.59      0.55      0.57       702
         3.0       0.16      0.22      0.19       277
         4.0       0.26      0.39      0.31       377
         5.0       0.25      0.42      0.31       271
         6.0       0.49      0.58      0.53       500
         7.0       0.19      0.34      0.24       196
         8.0       0.06      0.05      0.05        61
         9.0       0.33      0.42      0.37       241
        10.0       0.23      0.21      0.22        33

    accuracy                           0.36      4440
   macro avg       0.30      0.34      0.31      4440
weighted avg       0.42      0.36      0.37      4440

2.420247815106366
```

In [14]:

```
predictions, raw_outputs = model.predict(df_test['Text'].to_list())
```

In [15]:

```
df_final = df_test.copy()
reverse_label_dict = {v:u for u,v in label_dict.items()}
reverse_label_dict
df_final['Predicted_Labels'] = predictions
df_final['Predicted_Labels'] = df_final['Predicted_Labels'].apply(lambda x:reverse_l
df_final['Labels'] = df_final['Labels'].apply(lambda x:reverse_label_dict[x])
df_final['pid'] = df_final.index
df_final = df_final[['pid','Predicted_Labels','Labels']]
```

In [16]:

```
df_final
```

Out[16]:

	pid	Predicted_Labels	Labels
0	0	Ambiguous	Ambiguous
1	1	Disguist	Disguist
2	2	Anger	Disguist
3	3	Ambiguous	Ambiguous
4	4	Joy	Joy

	pid	Predicted_Labels	Labels
...	...	...	...
4435	4435	Neutral	Neutral
4436	4436	Trust	Trust
4437	4437	Trust	Anticipation
4438	4438	Joy	Joy
4439	4439	Neutral	Ambiguous

4440 rows × 3 columns

In [17]:
```
score = f1_score(df_final['Labels'],df_final['Predicted_Labels'],average='macro')
print("The macro average f1 score is:" + str(score))
```

```
The macro average f1 score is:0.30695567159425036
```

In [ ]:

REFERENCE

1. Thilagavathi, R. and Krishnakumari, K., 2016. Tamil english language sentiment analysis system. Int J Eng Res Technol, 4, pp.114-118.
2. Chakravarthi, B. R., Muralidaran, V., Priyadharshini, R., & McCrae, J. P. (2020). Corpus creation for sentiment analysis in code-mixed Tamil-English text. arXiv preprint arXiv:2006.00206.
3. Peng, S., Cao, L., Zhou, Y., Ouyang, Z., Yang, A., Li, X., ... & Yu, S. (2021). A survey on deep learning for textual emotion analysis in social networks. Digital Communications and Networks.
4. Jada, P. K., Yasaswini, K., Puranik, K., Sampath, A., Thangasamy, S., & Thamburaj, K. P. (2021). Pegasus@ Dravidian-CodeMix-HASOC2021: Analyzing Social Media Content for Detection of Offensive Text. arXiv preprint arXiv:2111.09836.
5. bert-base-multilingual-cased · Hugging Face
6. Cross Lingual Models(XLM-R). A deep dive into XLM-R | by aman anand | Medium
7. MuRIL | A General Introduction to MuRIL - Analytics Vidhya