The Emotional Twitterverse

A Comparative Study of Naive Bayes, Bidirectional-LSTM, and BERT Algorithms for Emotion Classification of Tweets with Lexiconned and Non-Lexiconned Data

Radhika Mardikar, Christian Montecillo

Abstract

In this report, we build a model to classify the emotion of a tweet or short text in 7 distinct categories: joy, love, sadness, anger, fear, surprise, and neutral. We used two datasets to build our train, validation and test sets. The first dataset is sourced from Kaggle[5] and the second is sourced from the CARER paper[1]. Using these datasets, we built a lexicon for emotions to feed into our models. We experimented with a baseline Naive Bayes model, Bi-LSTM model, and then a final BERT model. The Naive Bates and BERT models were run with and without a lexicon. We evaluate our models on the weighted F1-score to account for precision and recall for all emotions. On a clean version of the training and test set, the final BERT model had an overall 0.97 weighted F1-score with precision and recall above 0.93 for all emotions. On a mixture training set of a mixture of data, the BERT model had an overall weighted F1-score of 0.95, with above 0.9 precision and recall for all emotions except neutral. While neutral had high precision in this experiment, its recall was quite low. This model has many applications for sentiment analysis, mental health monitoring, and beyond.

Introduction

Social media platforms have become an essential part of our daily lives. We've used these platforms as a way to find community and belonging, to voice our thoughts, to stay in touch, and as news sources. Twitter, in particular, has emerged as a powerful tool for people to share their thoughts and feelings. In this paper, we aim to explore emotion classification of Tweets by training various models. While sentiment analysis has traditionally focused on three categories (positive, negative, and neutral), we will classify more complex emotions. We will limit ourselves to seven distinct emotions that are part of the dataset we have chosen. To achieve this, we will explore the work of Ekman, who uses six basic categories for all emotions from which others can be derived. Current methods of emotion detection include using standard machine learning models like SVM, Naive Bayes, and decision trees to classify emotions. Some involve using mechanical turks (from Amazon) to create a lexicon of emotion words to create a set of rules, and others involve a hybrid model which leverages the lexicon in machine learning models. Using a Bi-LSTM model and BERT model have also been explored. In this paper, we build a small scale version of some of these models to explore their limitations and capabilities.

The potential applications of this technology are vast, ranging from customer service to mental healthcare and entertainment. We will explore creating our own lexicon mapping of emotions and words and feeding them into standard ML algorithms, BERT, and using a Bi-LSTM classifier for emotion classification. We will evaluate the performance using weighted F1 scores as reported in literature review. We use F1 because it balances precision and recall and we use the weighted version to account for multiclass classification. Ultimately, we hope to see how the results of this paper can be used in the various applications stated above.

Background & Related Work

We owe much of the dataset cleaning to the work of the CARER paper[1] listed in our reference section. Similar to the work that Yu et al.,[4] did, in our base model we used a Bi-LSTM network to sequentially process each word in the input with a standard softmax output layer for classification. Yu[4] uses a shared attention-based Bi-LSTM layer to transform the input sentences into a shared hidden representation H_c, and also employs another task-specific Bi-LSTM layer to get the target-specific hidden representation H_t. Our implementation is less complex with only one Bi-LSTM layer.
In the Tzacheva paper[6], an existing NRC lexicon was applied to Twitter data to detect emotions and find actionable rules. These were fed into a SVM model. We can reapply this logic to extract predictive words from the data (create our own lexicon) and feed it into a standard ML model and a language model. The Seal paper[7] takes this a step further by matching words to the lexicon and if a word doesn't exist, to look up synonyms on WordNet. They were limited by the lexicon so we will need to build a large enough lexicon. The Huang paper[8] experiments with a BERT model for emotion detection from TV dialogue. This is slightly different from our project, but we will use the bert-base-cased (since capitalization makes a difference) for this task with a simple architecture.

Methods

Details of Datasets

Initial Dataset from Kaggle

We began by using a dataset of Tweets and emotion labels from Kaggle[5] with 40,000 rows of text and 13 emotion labels. Our analysis revealed that the dataset was imbalanced, with most labels being neutral or worry, and anger being the least represented. We found a non-trivial amount of Tweets were mislabeled, and ultimately decided not to use this dataset for training (ref. Figure A1 in the Appendix). We incorporated a cleaned version of this dataset for experimentation in Phase 3 described in later sections of this paper.

Second Dataset from CARER Paper

We sourced our second dataset from Saravia et. al. (CARER)[1]. The researchers used a dataset consisting of short text and emotion labels. This dataset had 416,809 records, and its distribution is shown in Figure A2. Unlike the previous dataset, the CARER dataset had consistent labeling. To create a more universal model, appropriately labeled neutral statements were added as negative controls from the Kaggle dataset. After preprocessing, the final dataset contained 379,604 rows (Figure A3). This dataset formed the basis for other datasets used in our experiments, which we describe in more detail.

Splitting Our Dataset

We split our full dataset into a 10% test set and a 90% training set. The training set was further split into 75% training and 25% validation. The final training set had 256k rows, the validation set had 85k rows, and the 10% test set had 37k rows. To maintain proportionality and account for the imbalance of emotion labels, we stratified the datasets. We also derived a balanced test set with 860 records for each emotion from the 10% test dataset. We conducted experiments with smaller training and validation datasets but kept the test sets constant. See Figure A4 for more details and distribution of the full datasets.

Enriching the Data with Lexicon Table

Inspired by Acheampong et. al's review[3], we created our own version of a lexicon and supplied it to the models. Filtering the dataset to only include a word if it was in the lexicon means that we could create a very predictive model. To create our lexicon, we tokenized the dataset and exploded it to determine the number of times a word appeared per emotion. We took the amount of times a word appeared in that specific emotion and the amount of times it appeared in the overall dataset and calculated the percentage of that word per emotion – the predictiveness of that word for a specific emotion. We then filtered the lexicon to only include words appearing more than 30 times in the whole dataset with at least a 65% predictive power for a specific emotion in order to filter out typos, and abnormal words. These filters were chosen after some analysis leaving us with 463 words in the lexicon (Figure A5).

Experiments

Phase 1: Creating a baseline

Base Model with Naive Bayes

The first model was Naive Bayes because of its versatility in terms of working with text and because it works much differently from more sophisticated language models. Specifically it evaluates each word's probability of occurrence independently rather than conditionally. For the first Naive Bayes experiment, we fed in a count-vectorized version of the 136k training set and evaluated the model on the test set. We expected decent overall performance >70% F1 score. For the second Naive Bayes experiment, we fed in a count-vectorized version of the lexiconned 136k training set by filtering out any word in the training set not present in the lexicon. This amplified the important words by getting rid of any word that may dampen the signal. We also filtered the test set to include only lexiconned words. We expected performance to be much higher for individual sectors than previously, though neutral would likely take a hit since there are no high signal neutral words. For the third Naive Bayes experiment, we fed in a count-vectorized version of the reverse-lexiconned 130k training set. This is the opposite of the previous experiment, meaning that we included only those words NOT present in the lexicon. We applied the same treatment to the test set. We expected this model to perform very poorly since we are taking out anything that would have a predictive signal.

Base Model with Bidirectional LSTM

Our second model was a Bi-LSTM network, known for performing well with six emotion classes[1]. To account for our additional 'neutral' class, we used this architecture with seven classes for text that did not have a distinct emotion. We chose the Bi-LSTM architecture because it is an older architecture compared to the BERT model used in later phases of our project, but is better suited to language tasks than the Naive Bayes model. We expected it to perform well, though we anticipated that it would not perform as well as BERT or other more recent state-of-the-art models.
To determine the `max_length` for our tokenizer we plotted the Tweets and set `max length` to 200 tokens for our experiments for generalizability (Figure A6). Per Gowda and May[2], 8,000 words is an optimal vocabulary size for datasets between 30K and 1.3M rows which we tested by using a larger vocabulary size of 11,039 words constructed by filtering out words that occurred less than 10 times in the overall dataset. Our model consists of three layers: an embedding layer, a bidirectional LSTM layer, and a dense output layer for classifying our seven emotions. The embedding layer converts the input sequence into a dense vector representation, using a variable number of input dimensions (11,039 and 8,000 based on vocabulary size), an output dimension of 8, and a fixed `max_len` input length. The bidirectional LSTM

layer processes the output of the embedding layer both forwards and backwards using 10 hidden units, capturing both past and future context for the sequence. The output layer is a dense layer with 7 neurons and a softmax activation function, representing the 7 classes of the classification task. We compile the model using the `sparse_categorical_crossentropy` loss function, and the `adam` optimizer.

Phase 2: Experimenting with State of the Art

Next, we move on to experimenting with BERT. For these experiments, we use the Bert-base-cased pretrained model. We constructed a base model with one hidden layer of dimension 201, one dropout layer with a dropout rate of 0.3, and one classification layer using the softmax activation function since we have a total of 7 classes to predict. Variations of hidden layer dimensions were tested before choosing the reported parameters. We use the `sparse_categorical_crossentropy` loss function, and `adam` optimizer. For our test set, we evaluate overall model performance using the weighted F1-score.
We ran the same set of experiments as on Naive Bayes on BERT to compare performance. However, one crucial difference was training time. Where Naive Bayes could compute ~140k rows of training data almost instantaneously, BERT would take over 3 hours for 1 epoch. This was definitely not sustainable for our purposes and we scaled BERT's dataset down to 17k rows for training. To test the effects of more training data on weighted F1 accuracy, we ran the first experiment (described in the following paragraph) on 34k rows, and 68k rows of training data.
First, we fed in 17k rows of training data with no lexicon filtering into BERT. We expected to perform decently and better than the baseline models since BERT has an inherent "understanding" of language and has a CLS token to help inform prediction.
Second, we passed in a lexiconned version of the 17k rows meaning that we include only those words that are present in the lexicon. We expected this to perform around the same as before, but poorly on neutral since there are no words that point explicitly to the neutral category. This experiment was a little tricky to hypothesize since there are strong arguments for better performance or worse performance. Since we are amplifying the signal for each category, we could say that it becomes easier to learn the difference between prediction classes. On the other hand, by removing transitional words, we decrease the ability of the model to map it to real language patterns, which could make predictions more difficult.
Finally, we passed in a reverse lexiconned version of the 17k rows, meaning that we only included words not present in the lexicon. We expected this to perform very poorly.

Phase 3: Breaking our Models

To further explore our models' robustness, we introduced some disruption by incorporating the Kaggle model, which had questionable labeling quality. We merged the Kaggle dataset with our 34k dataset, randomized the combined data, and divided it into smaller sets using stratification. The resulting dataset was a mix of CARER and Kaggle data with a 50-50 split. It is important to note that the Kaggle dataset had 13 emotion labels, while our CARER dataset only had six. To combine the datasets, we mapped the Kaggle emotion labels to the CARER labels inspired by Plutchik's wheel of emotions[3] (Figure A7).
We converted Kaggle emotion labels to CARER emotion labels by assigning them as follows: `{'empty': 'sadness', 'enthusiasm': 'joy', 'worry': 'fear', 'fun': 'joy', 'hate': 'anger', 'happiness': 'joy', 'boredom': 'neutral', 'relief': 'joy'}`. Then, we removed duplicate rows of 'text' regardless of their emotion label, combined the remaining data, shuffled it, and split it into training (20,324 records) and validation (12,322 records) sets while ensuring a proportional representation of emotion labels in each set. We used our original imbalanced and balanced datasets as our test set to maintain consistency throughout our experiments. We then ran a subset of our experiments again using this new, combined dataset.

Results and Discussion:

Phase 1: Baseline results

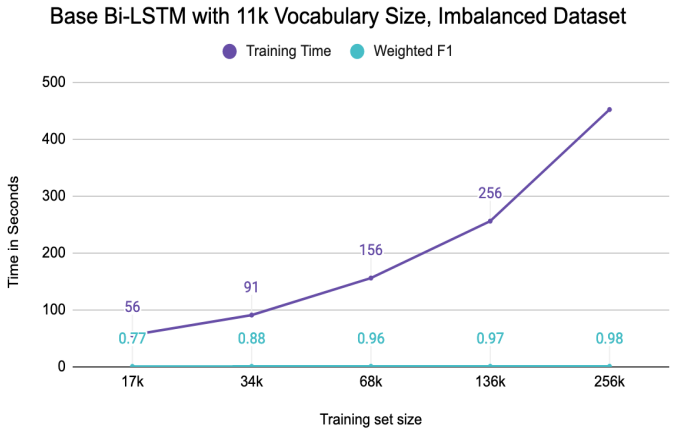
Naive Bayes Results

All confusion matrices and classification reports are in our Appendix.
For the first Naive Bayes experiment with the full 136k dataset, the model does well (~89%) on the weighted F1 metric. However, breaking it down by individual class, we see that it performs quite poorly on the neutral and surprise category. Looking at the support for each of those categories, it is likely due to a class imbalance, however since our final model is not a Naive Bayes model, we did not utilize any upsampling techniques. This goes slightly against our hypothesis, since we did not anticipate a lower support resulting in this low of a recall score. Running this same experiment on the 17k drops the F1 score a bit, as expected because there is less data. For the second experiment, filtering the large 136k training set to only include words from the lexicon gives much better precision and recall for individual sectors with only a slight improvement in overall weighted F1 score. The main reason the overall F1 score does not increase is because neutral is almost 0. This is expected because the lexicon does not contain any neutral words since there are no explicit signal amplifier words in that category. These results match our hypothesis.The third experiment with reverse lexicon fails quite spectacularly, just as expected, since we take out any word that could be indicative of a particular category.

Base Bi-LSTM Results

Our Bi-LSTM model achieved a validation accuracy of 98.02% on the 256k training dataset using a vocabulary size of 11k. With 8k vocabulary, the model achieved a validation accuracy of 98.15%. We tested the model on imbalanced and balanced test sets, and obtained a 98% weighted F1 score for both vocabulary sizes. This model performed well due to the clean dataset and the network architecture's suitability for short texts. The best performing experiment was with 8k vocabulary size tested against the imbalanced dataset. An 8k input size is optimal for this size dataset, and we observed that the greater the number of training records, the better the performance. Reducing the volume of training data had an impact on performance as seen in Figure 1a below. See Figure A8 for additional details.

Phase 2: BERT Training Results



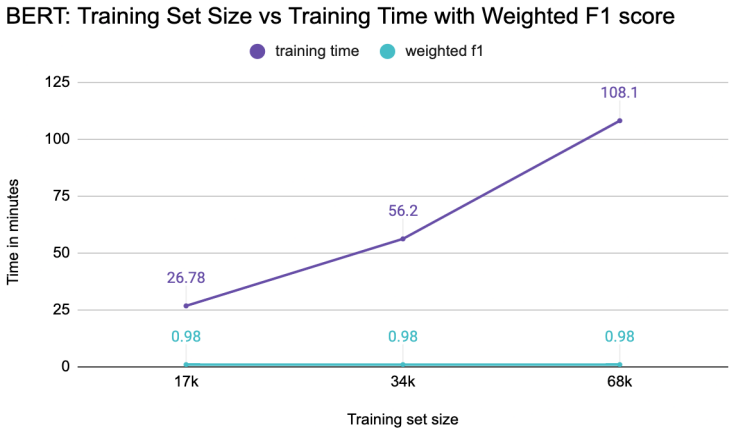


Figure 1: a) Bi-LSTM training time vs training set size vs F1 score b) BERT training time vs training set size vs F1 score

From Figure 1b, while training time increases by 2x with each increase in training set size, the evaluation metric stays the same, indicating that having smaller sets of data does not mean worse overall performance. Since the evaluation metric stays constant, we end the experimentation at 68k rows of training data. It is evident that using the 17k training set is enough for BERT so we use that for all future BERT experiments.
For the first experiment, the model has a weighted F1-avg score of 0.97 with high precision & recall (>0.90) for all sectors. This is a very strong model, as expected. These results also prove that no upsampling techniques were necessary to enhance model performance – this was not true for the Naive Bayes baseline. For the second experiment with the lexicon-filtered training set, the model had a weighted F1-avg score of 0.90, and precision & recall scores above 0.80 for all emotions. As expected, neutral is 0 since there is nothing in the lexicon to lead to a classification of that label. For the third experiment with reverse lexicon filtering, BERT performed quite poorly as expected. From these experiments it is obvious that BERT does not need a lexicon filtering to perform its classification task properly. Signal amplification does not seem necessary for this language model. Since this model is trained on clean data from the CARER dataset, we then test it on the Kaggle dataset which is poorly labeled. The model performs very poorly, around 0.04 weighted F1. It becomes apparent that the training set needs to include some data from a source that is not so clean and processed, so we move onto Phase 3.

Architecture Name	Training Time	F1
Phase 1: Base Bi-LSTM 8k vocabulary	447s	0.98
Phase 1: NB lexiconned		0.90254
Phase 1: NB full data		0.89205
Phase 2: Bert 17k lexiconned	1709s	0.90
Phase 2: Bert 17k full data	1607s	0.97
Phase 3: NB on carer, kaggle mix		0.78189
Phase 3: Bert on carer, kaggle	2162s	0.95

Table 1: Results

Phase 3: Combined dataset results

Naive Bayes Results

Introducing the Kaggle data to the training set did not result in a strong model. Almost every sector took a hit in either precision or recall, resulting in low weighted F1 score. This can be attributed to the low quality of the Kaggle dataset.

Base Bi-LSTM Results

Introducing the Kaggle data to the training of the model shows that the CARER-only trained model of size 17k did not outperform the combined-data-trained model of similar size indicating that for Bi-LSTM, one of the main contributing factors to our model's performance was the amount of clean training data provided.

BERT results

Running the BERT model on the combined Kaggle and Clean dataset results in an overall F1 score 0.95 with a precision and recall for each emotion above 0.8 except for the neutral emotion. Neutral has a high

precision but low recall. Many neutrals were classified as fear or joy, since we did not observe this problem on the CARER set, it can be attributed to dubious labeling on the Kaggle set.

Conclusion

In this paper, we built models to classify emotions of short texts or Tweets in 7 distinct categories. Based on the weighted F1-score, we see that the model performs its task quite well. Key findings are that BERT does not require large amounts of training data to be successful, and does not require additional text preprocessing (such as lexicon filtering) to accurately make its predictions. Another key learning is that with the Bi-LSTM model, having a larger vocabulary does not necessarily mean better performance; however, more training data will lead to better performance. In the future, we would expand this model out for more emotions and evaluate it on other datasets that may not be so clean. In the future, we would test the quality of different data sources by running a classifier, and apply this model to a use case like mental health detection.

References

1. Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized Affect Representations for Emotion Recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697, Brussels, Belgium. Association for Computational Linguistics.
2. Thamme Gowda and Jonathan May. 2020. Finding the Optimal Vocabulary Size for Neural Machine Translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3955–3964, Online. Association for Computational Linguistics.
3. Francisca Adom Acheampong, Chen Wenyu, and Henry Nunoo-Mensah. 2020. Text-based emotion detection: Advances, challenges,and opportunities. In Engineering Reports, Volume 2, Issue 7, July 2020.
4. Jianfei Yu, Luís Marujo, Jing Jiang, Pradeep Karuturi, and William Brendel. 2018. Improving Multi-label Emotion Classification via Sentiment Classification with Dual Attention Transfer Network. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1097–1102, Brussels, Belgium. Association for Computational Linguistics.
5. Ishant Juyal. 2021. Emotions In Text. Version 1. Retrieved 02-25-2023 from https://www.kaggle.com/datasets/ishantjuyal/emotions-in-text
6. Tzacheva, A., Ranganathan, J., Mylavarapu, S.Y. (2020). Actionable Pattern Discovery for Tweet Emotions. In: Ahram, T. (eds) Advances in Artificial Intelligence, Software and Systems Engineering. AHFE 2019. Advances in Intelligent Systems and Computing, vol 965. Springer, Cham. https://doi.org/10.1007/978-3-030-20454-9_5
7. Seal, D., Roy, U.K., Basak, R. (2020). Sentence-Level Emotion Detection from Text Based on Semantic Rules. In: Tuba, M., Akashe, S., Joshi, A. (eds) Information and Communication Technology for Sustainable Development. Advances in Intelligent Systems and Computing, vol 933. Springer, Singapore. https://doi.org/10.1007/978-981-13-7166-0_42
8. Huang, Y.H., Lee, S.R., Ma, M.Y., Chen, Y.H., Yu, Y.W. and Chen, Y.S., 2019. EmotionX-IDEA: Emotion BERT--an Affectional Model for Conversation. *arXiv preprint arXiv:1908.06264*.

Appendix

	count	percentage
neutral	8638	21.5950
worry	8459	21.1475
happiness	5209	13.0225
sadness	5165	12.9125
love	3842	9.6050
surprise	2187	5.4675
fun	1776	4.4400
relief	1526	3.8150
hate	1323	3.3075
empty	827	2.0675
enthusiasm	759	1.8975
boredom	179	0.4475
anger	110	0.2750

Figure A1: Initial Kaggle dataset and its distribution

	counts	percentage
joy	141067	33.844519
sadness	121187	29.074948
anger	57317	13.751383
fear	47712	11.446970
love	34554	8.290128
surprise	14972	3.592053

Figure A2: Full CARER dataset and its distribution

	count	percentage
joy	129165	34.026248
sadness	115868	30.523388
anger	52260	13.766978
fear	39554	10.419806
love	24335	6.410628
surprise	9821	2.587170
neutral	8601	2.265782

Figure A3: Final CARER dataset and its distribution after deduplication and adding the ‘neutral’ emotion

Training dataset proportions

	count	percentage
joy	87186	34.026195
sadness	78211	30.523510
anger	35275	13.766821
fear	26699	10.419854
love	16426	6.410597
surprise	6629	2.587109
neutral	5806	2.265915

Validation dataset proportions

	count	percentage
joy	29062	34.026062
sadness	26070	30.523001
anger	11759	13.767548
fear	8900	10.420203
love	5475	6.410181
surprise	2210	2.587489
neutral	1935	2.265516

Test dataset proportions

	count	percentage
joy	12917	34.027028
sadness	11587	30.523432
anger	5226	13.766761
fear	3955	10.418587
love	2434	6.411844
surprise	982	2.586865
neutral	860	2.265483

Balanced test set

	count	percentage
anger	860	14.285714
fear	860	14.285714
joy	860	14.285714
love	860	14.285714
neutral	860	14.285714
sadness	860	14.285714
surprise	860	14.285714

Figure A4: Full training, validation, imbalanced test, and balanced test datasets.

cleaned_stopwords	emotions	counter_x	counter_y	percent_of_group
abused	sadness	1307	1348	0.969585
acceptable	joy	1416	1464	0.967213
aching	sadness	1423	1489	0.955675
acquired	joy	21	32	0.656250
adoring	love	84	91	0.923077
...	...	...	...	...
woefully	sadness	27	33	0.818182
wonderful	joy	1850	2421	0.764147
worthless	sadness	1507	1595	0.944828
worthwhile	joy	1436	1482	0.968961
wronged	anger	1347	1369	0.983930

Figure A5: Lexicon Table

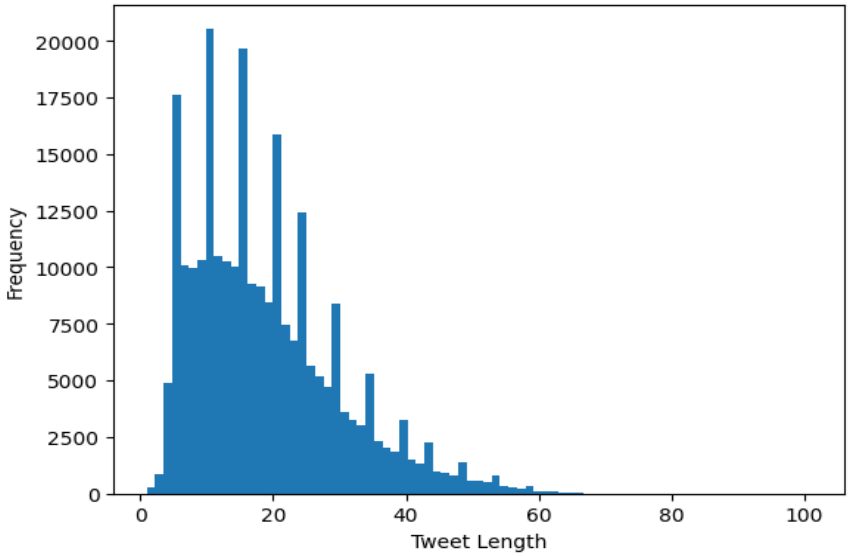


Figure A6: To determine the `max length` for our tokenizer we plotted the Tweets and discovered that the maximum words in a Tweet in the training data was 101 words

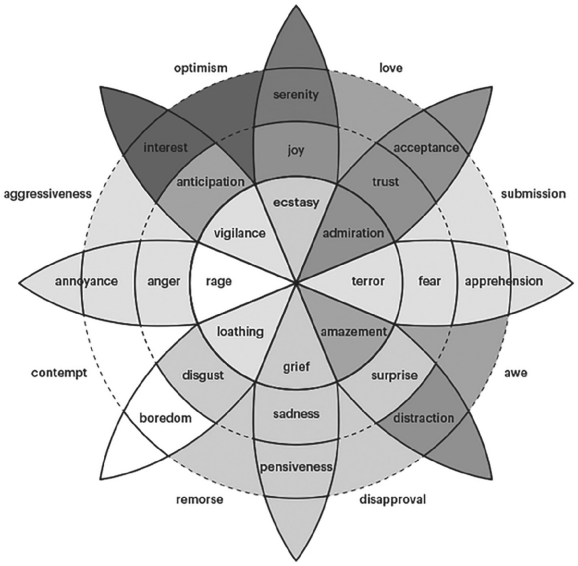


Figure A7: Plutchik's wheel of emotions

Training	Validation
17082	5125
34164	10250
68328	20499
136657	40998

Figure A8: Sizes of the datasets used in experiments.

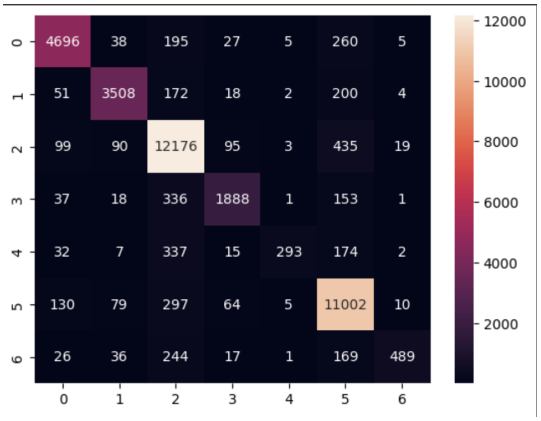
Classification Reports/Confusion Matrix Per Experiment:

Phase 1:

Naive Bayes

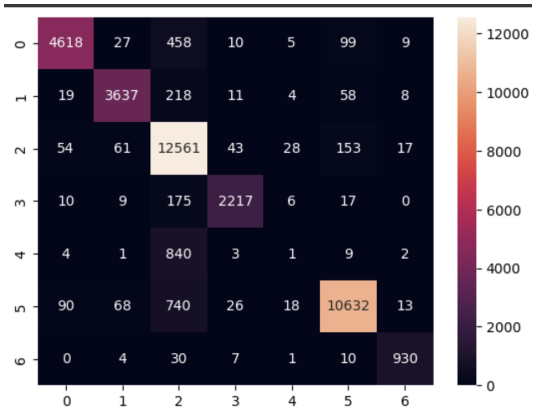
Experiment 1: On full training set

	precision	recall	f1-score	support
anger	0.92605	0.89858	0.91211	5226
fear	0.92903	0.88698	0.90752	3955
joy	0.88508	0.94263	0.91295	12917
love	0.88889	0.77568	0.82843	2434
neutral	0.94516	0.34070	0.50085	860
sadness	0.88776	0.94951	0.91760	11587
surprise	0.92264	0.49796	0.64683	982
accuracy			0.89703	37961
macro avg	0.91209	0.75601	0.80376	37961
weighted avg	0.89869	0.89703	0.89205	37961



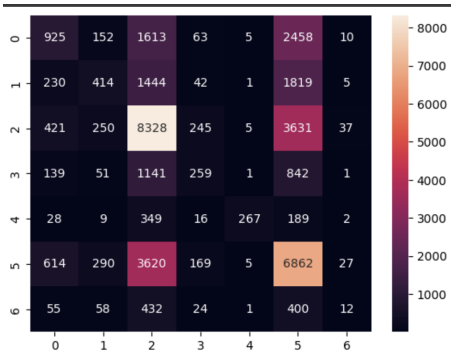
Experiment 2: Training set filtered by lexicon words:

	precision	recall	f1-score	support
anger	0.96309	0.88366	0.92166	5226
fear	0.95535	0.91960	0.93713	3955
joy	0.83617	0.97244	0.89917	12917
love	0.95684	0.91085	0.93328	2434
neutral	0.01587	0.00116	0.00217	860
sadness	0.96848	0.91758	0.94234	11587
surprise	0.94995	0.94705	0.94850	982
accuracy			0.91136	37961
macro avg	0.80654	0.79319	0.79775	37961
weighted avg	0.89854	0.91136	0.90254	37961



Experiment 3: Dataset filtered with reverse lexicon

	precision	recall	f1-score	support
anger	0.38350	0.17700	0.24221	5226
fear	0.33824	0.10468	0.15988	3955
joy	0.49200	0.64473	0.55810	12917
love	0.31663	0.10641	0.15929	2434
neutral	0.93684	0.31047	0.46638	860
sadness	0.42355	0.59222	0.49388	11587
surprise	0.12766	0.01222	0.02230	982
accuracy			0.44959	37961
macro avg	0.43120	0.27825	0.30029	37961
weighted avg	0.42956	0.44959	0.41201	37961



Bi-LSTM

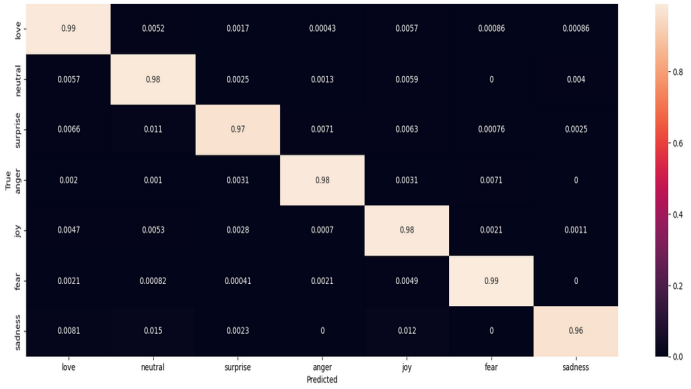
Experiment 1: Full Training set
Imbalanced Test Set

```
Classification Report:

              precision    recall  f1-score   support

        love       0.99      0.99      0.99     11587
     neutral       0.96      0.98      0.97      5226
    surprise       0.98      0.97      0.97      3955
       anger       0.95      0.98      0.97       982
         joy       0.99      0.98      0.99     12917
        fear       0.98      0.99      0.99      2434
     sadness       0.94      0.96      0.95       860

    accuracy                           0.98     37961
   macro avg       0.97      0.98      0.97     37961
weighted avg       0.98      0.98      0.98     37961
```



Balanced Test Set

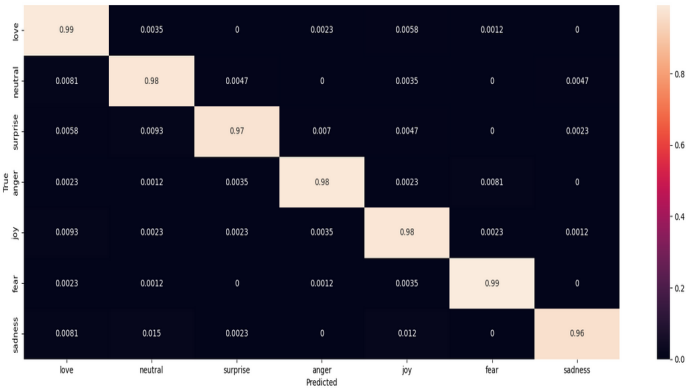
```
Classification Report:

              precision    recall  f1-score   support

        love       0.96      0.99      0.98       860
     neutral       0.97      0.98      0.97       860
    surprise       0.99      0.97      0.98       860
       anger       0.99      0.98      0.98       860
         joy       0.97      0.98      0.97       860
        fear       0.99      0.99      0.99       860
     sadness       0.99      0.96      0.98       860

    accuracy                           0.98      6020
   macro avg       0.98      0.98      0.98      6020
weighted avg       0.98      0.98      0.98      6020

Confusion Matrix:
```



Experiment 2: Reduced Dataset: 17k Training, 5k Validation
Imbalanced Test Set

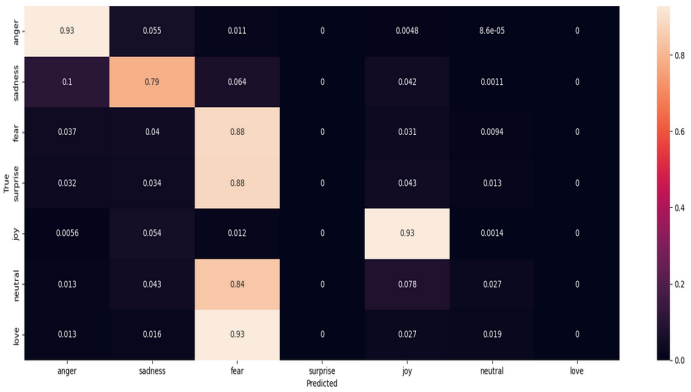
```
Classification Report:

              precision    recall  f1-score   support

       anger       0.93      0.93      0.93     11587
     sadness       0.71      0.79      0.75      5226
        fear       0.45      0.88      0.59      3955
    surprise       0.00      0.00      0.00       982
         joy       0.95      0.93      0.94     12917
     neutral       0.42      0.03      0.05      2434
        love       0.00      0.00      0.00       860

    accuracy                           0.80     37961
   macro avg       0.49      0.51      0.47     37961
weighted avg       0.78      0.80      0.77     37961

Confusion Matrix:
```

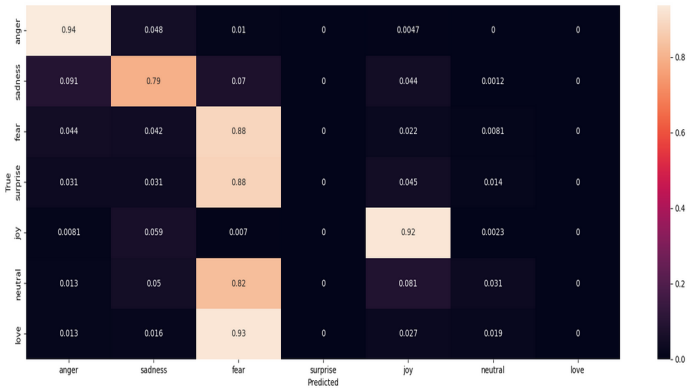


Balanced Test Set

Classification Report:

	precision	recall	f1-score	support
anger	0.82	0.94	0.88	860
sadness	0.76	0.79	0.78	860
fear	0.25	0.88	0.38	860
surprise	0.00	0.00	0.00	860
joy	0.80	0.92	0.86	860
neutral	0.42	0.03	0.06	860
love	0.00	0.00	0.00	860
accuracy			0.51	6020
macro avg	0.44	0.51	0.42	6020
weighted avg	0.44	0.51	0.42	6020

Confusion Matrix:

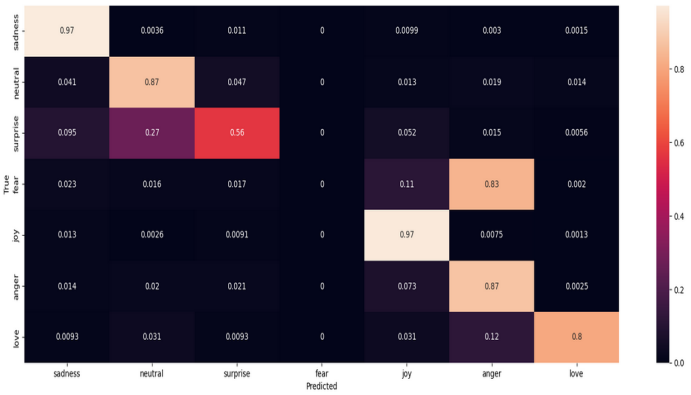


Experiment 3: Reduced Dataset: 34k Training, 10k Validation
Imbalanced Test Set

Classification Report:

	precision	recall	f1-score	support
sadness	0.93	0.97	0.95	11587
neutral	0.79	0.87	0.82	5226
surprise	0.80	0.56	0.66	3955
fear	0.00	0.00	0.00	982
joy	0.95	0.97	0.96	12917
anger	0.64	0.87	0.73	2434
love	0.83	0.80	0.82	860
accuracy			0.88	37961
macro avg	0.70	0.72	0.71	37961
weighted avg	0.86	0.88	0.86	37961

Confusion Matrix:



Balanced Test Set

Classification Report:

	precision	recall	f1-score	support
neutral	0.82	0.97	0.89	860
sadness	0.72	0.88	0.79	860
surprise	0.85	0.55	0.67	860
fear	0.00	0.00	0.00	860
joy	0.76	0.96	0.85	860
anger	0.46	0.85	0.60	860
love	0.96	0.80	0.87	860
accuracy			0.72	6020
macro avg	0.65	0.72	0.67	6020
weighted avg	0.65	0.72	0.67	6020

Confusion Matrix:

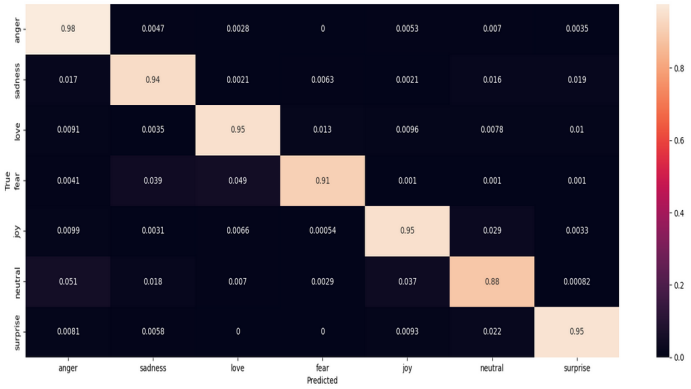
Experiment 4: Reduced Dataset: 68k Training, 20k Validation
Imbalanced Test Set

```
Classification Report:
              precision    recall  f1-score   support

       anger       0.97      0.98      0.97     11587
     sadness       0.96      0.94      0.95      5226
        love       0.95      0.95      0.95      3955
        fear       0.90      0.91      0.90       982
         joy       0.98      0.95      0.97     12917
     neutral       0.79      0.88      0.83      2434
    surprise       0.79      0.95      0.86       860

    accuracy                           0.95     37961
   macro avg       0.90      0.94      0.92     37961
weighted avg       0.95      0.95      0.95     37961

Confusion Matrix:
```



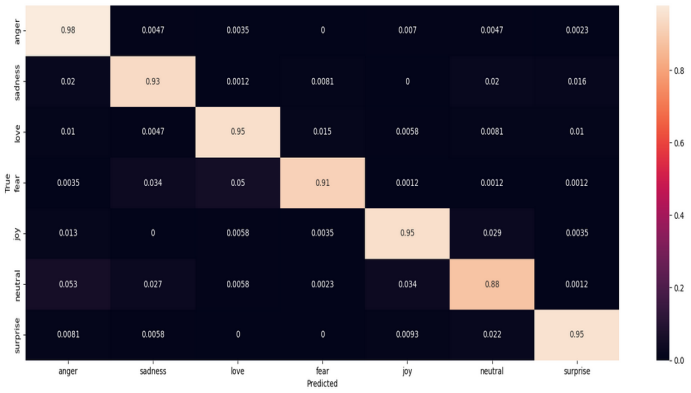
Balanced Test Set

```
Classification Report:
              precision    recall  f1-score   support

       anger       0.90      0.98      0.94       860
     sadness       0.93      0.93      0.93       860
        love       0.93      0.95      0.94       860
        fear       0.97      0.91      0.94       860
         joy       0.94      0.95      0.94       860
     neutral       0.91      0.88      0.89       860
    surprise       0.96      0.95      0.96       860

    accuracy                           0.93      6020
   macro avg       0.94      0.93      0.93      6020
weighted avg       0.94      0.93      0.93      6020

Confusion Matrix:
```



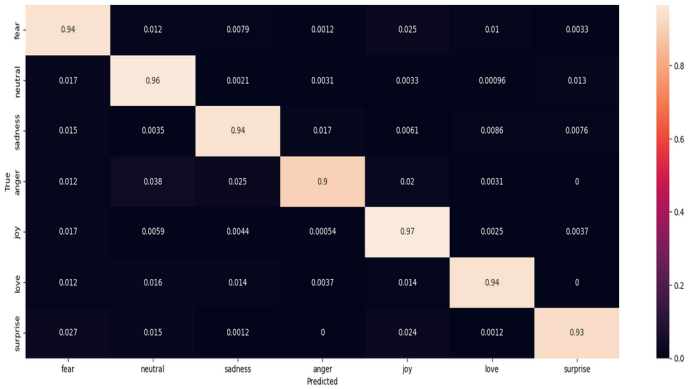
Experiment 5: Reduced Dataset: 136k Training, 40k Validation
Imbalanced Test Set

```
Classification Report:
              precision    recall  f1-score   support

        fear       0.96      0.94      0.95     11587
     neutral       0.94      0.96      0.95      5226
     sadness       0.94      0.94      0.94      3955
       anger       0.89      0.90      0.89       982
         joy       0.97      0.97      0.97     12917
        love       0.92      0.94      0.93      2434
    surprise       0.81      0.93      0.87       860

    accuracy                           0.95     37961
   macro avg       0.92      0.94      0.93     37961
weighted avg       0.95      0.95      0.95     37961

Confusion Matrix:
```



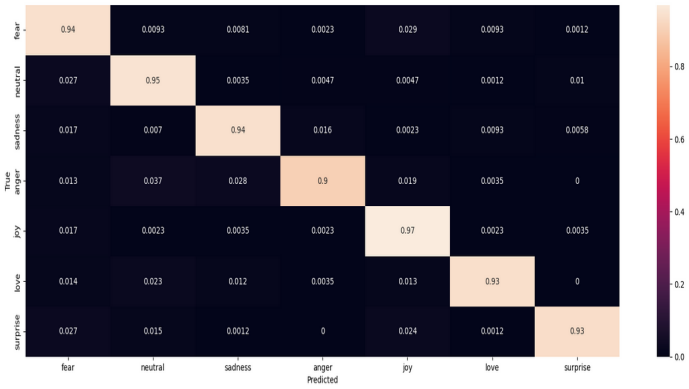
Balanced Test Set

```
Classification Report:
              precision    recall  f1-score   support

        fear       0.89      0.94      0.92       860
     neutral       0.91      0.95      0.93       860
     sadness       0.94      0.94      0.94       860
       anger       0.97      0.90      0.93       860
         joy       0.91      0.97      0.94       860
        love       0.97      0.93      0.95       860
    surprise       0.98      0.93      0.95       860

    accuracy                           0.94      6020
   macro avg       0.94      0.94      0.94      6020
weighted avg       0.94      0.94      0.94      6020

Confusion Matrix:
```



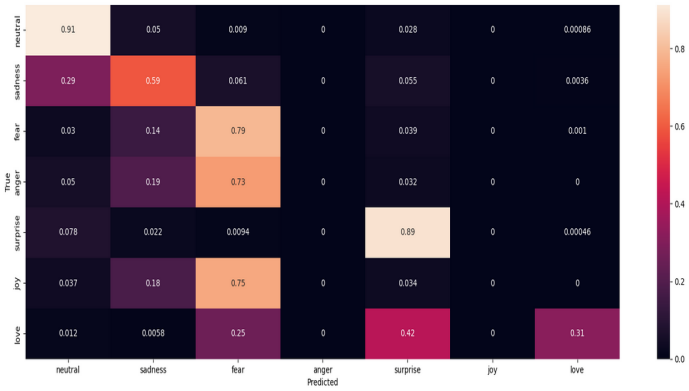
Experiment 6: Combined Kaggle + CARER Dataset, 20k Training, 12k Validation Imbalanced Test Set

```
Classification Report:
              precision    recall  f1-score   support

     neutral       0.79      0.91      0.85     11587
     sadness       0.60      0.59      0.60      5226
        fear       0.49      0.79      0.60      3955
       anger       0.00      0.00      0.00       982
    surprise       0.90      0.89      0.90     12917
         joy       0.00      0.00      0.00      2434
        love       0.87      0.31      0.46       860

    accuracy                           0.75     37961
   macro avg       0.52      0.50      0.49     37961
weighted avg       0.70      0.75      0.72     37961

Confusion Matrix:
```



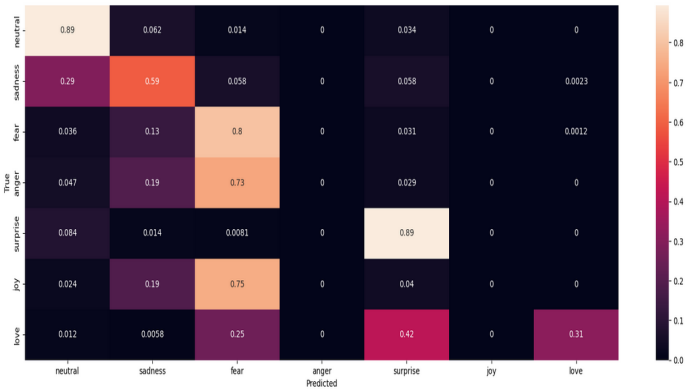
Balanced Test Set

```
Classification Report:
              precision    recall  f1-score   support

     neutral       0.64      0.89      0.75       860
     sadness       0.50      0.59      0.54       860
        fear       0.31      0.80      0.44       860
       anger       0.00      0.00      0.00       860
    surprise       0.60      0.89      0.72       860
         joy       0.00      0.00      0.00       860
        love       0.99      0.31      0.48       860

    accuracy                           0.50      6020
   macro avg       0.43      0.50      0.42      6020
weighted avg       0.43      0.50      0.42      6020

Confusion Matrix:
```

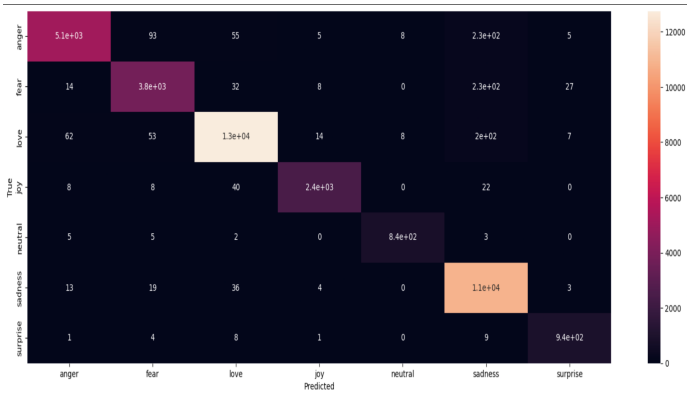


Phase 2:

BERT

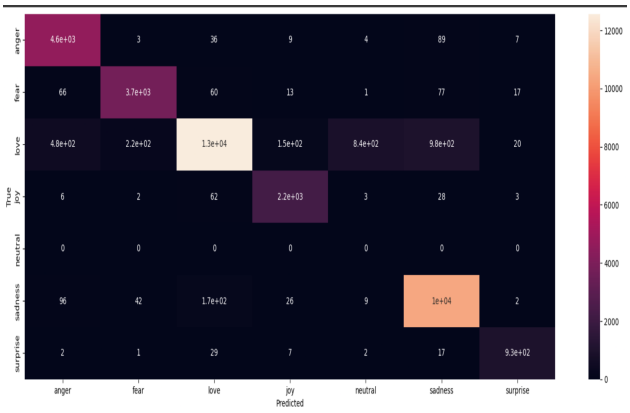
Experiment 1: Full Training set

	precision	recall	f1-score	support
0	0.93	0.98	0.95	5226
1	0.92	0.95	0.94	3955
2	0.97	0.99	0.98	12917
3	0.97	0.99	0.98	2434
4	0.98	0.98	0.98	860
5	0.99	0.94	0.97	11587
6	0.98	0.96	0.97	982
accuracy			0.97	37961
macro avg	0.96	0.97	0.97	37961
weighted avg	0.97	0.97	0.97	37961



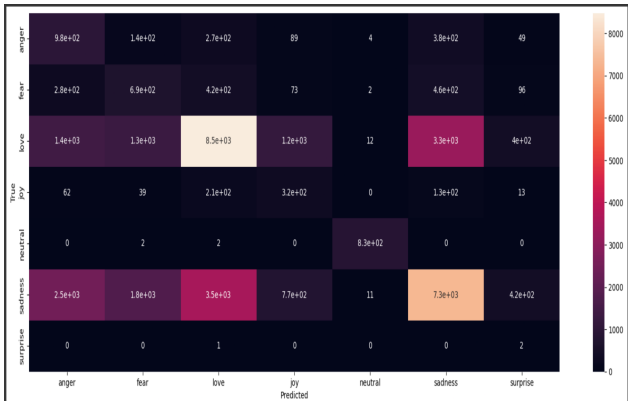
Experiment 2: Lexiconned training set

	precision	recall	f1-score	support
0	0.97	0.88	0.92	5226
1	0.94	0.93	0.94	3955
2	0.82	0.97	0.89	12917
3	0.96	0.92	0.93	2434
4	0.00	0.00	0.00	860
5	0.97	0.90	0.93	11587
6	0.94	0.95	0.95	982
accuracy			0.91	37961
macro avg	0.80	0.79	0.79	37961
weighted avg	0.89	0.91	0.90	37961



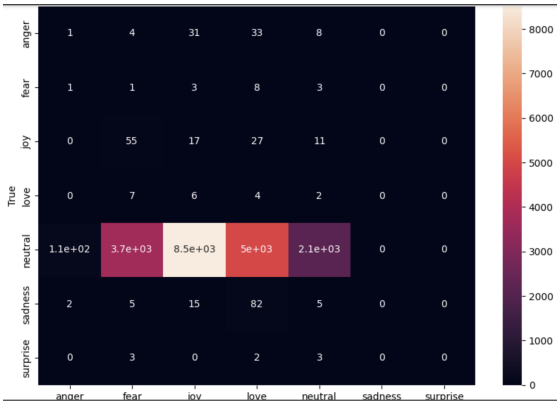
Experiment 3: Reverse Lexiconned training set

	precision	recall	f1-score	support
0	0.51	0.19	0.28	5226
1	0.34	0.17	0.23	3955
2	0.53	0.66	0.59	12917
3	0.41	0.13	0.20	2434
4	1.00	0.97	0.98	860
5	0.45	0.63	0.53	11587
6	0.67	0.00	0.00	982
accuracy			0.49	37961
macro avg	0.56	0.39	0.40	37961
weighted avg	0.49	0.49	0.46	37961



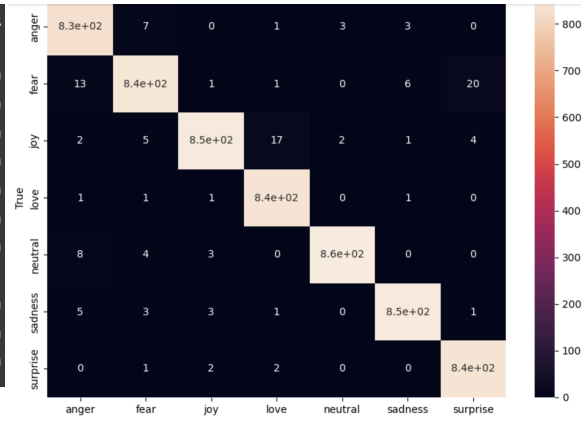
Experiment 4: Running best BERT on kaggle data set

	precision	recall	f1-score	support
0	0.01	0.01	0.01	110
1	0.06	0.00	0.00	3770
2	0.15	0.00	0.00	8580
3	0.21	0.00	0.00	5142
4	0.11	0.99	0.20	2177
5	0.00	0.00	0.00	0
6	0.00	0.00	0.00	0
accuracy			0.11	19779
macro avg	0.08	0.14	0.03	19779
weighted avg	0.15	0.11	0.02	19779



Experiment 5: Running best BERT on balanced test set

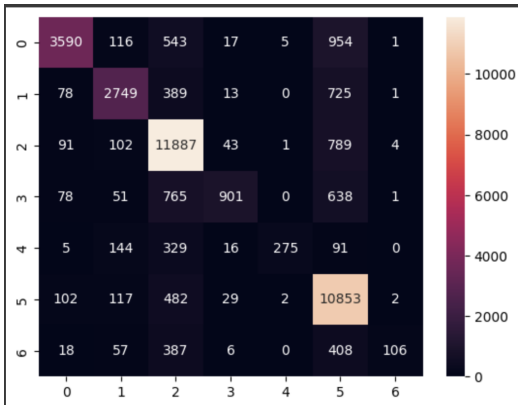
	precision	recall	f1-score	support
0	0.98	0.97	0.97	860
1	0.95	0.98	0.96	860
2	0.96	0.99	0.98	860
3	1.00	0.97	0.98	860
4	0.98	0.99	0.99	860
5	0.98	0.99	0.99	860
6	0.99	0.97	0.98	860
accuracy			0.98	6020
macro avg	0.98	0.98	0.98	6020
weighted avg	0.98	0.98	0.98	6020



Phase 3:

Naive Bayes on mixed CARER and Kaggle

	precision	recall	f1-score	support
anger	0.90611	0.68695	0.78145	5226
fear	0.82404	0.69507	0.75408	3955
joy	0.80415	0.92026	0.85830	12917
love	0.87902	0.37017	0.52096	2434
neutral	0.97173	0.31977	0.48119	860
sadness	0.75066	0.93665	0.83340	11587
surprise	0.92174	0.10794	0.19325	982
accuracy			0.79979	37961
macro avg	0.86535	0.57669	0.63181	37961
weighted avg	0.81557	0.79979	0.78189	37961



BERT on mixed CARER and Kaggle

	precision	recall	f1-score	support
0	0.98	0.95	0.97	5226
1	0.84	0.99	0.90	3955
2	0.96	0.98	0.97	12917
3	0.96	0.99	0.97	2434
4	0.85	0.15	0.26	860
5	0.99	0.97	0.98	11587
6	0.97	0.96	0.97	982
accuracy			0.96	37961
macro avg	0.94	0.86	0.86	37961
weighted avg	0.96	0.96	0.95	37961

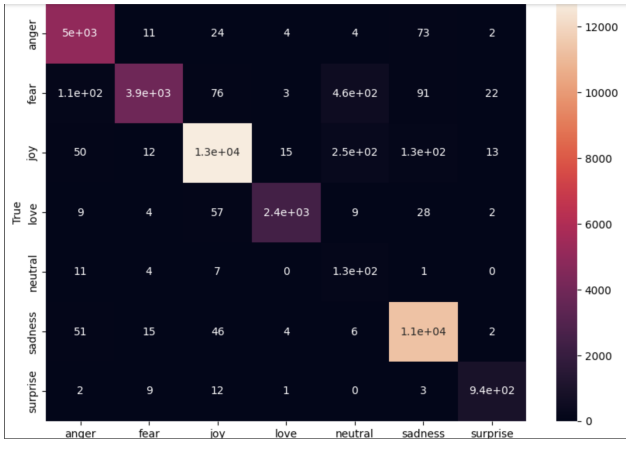


Table of Results

Architecture Name	Architecture Description	Data Permutation	Epochs	Training Time	val_accuracy	model.evaluate accuracy	F1	Comments
Base Bi-LSTM.01	Sequential model that has three layers: 1. An Embedding layer with input/vocabulary = 10,179 and max_len = 200. When doing EDA, 10179 was the number of words in the traiing set that showed up at least 10 times. We chose 200 for max_len. 2. A Bidirectional layer that wraps an LSTM layer with 10 hidden units. 3. A Dense output layer with our 7 classes and softmax activation.	1. Imbalanced training data (256232 records) 2. Imbalanced validation data (85411 records) 3. Imbalanced test data (37961 records) 4. balanced test data (6020 records)	5	11k: 452s 8k: 447s	11k: 0.9802 8k: 0.9815	1. imbalanced test, 11k: 0.9787 2. balanced, 11k: 0.9754 3. imbalanced test, 8k: 0.9817 4. balanced, 8k: 0.9791	1. imbalanced test, 11k: 0.98 2. balanced, 11k: 0.98 3. imbalanced test, 8k: 0.98 4. balanced, 8k: 0.98	With the number of records in the training and validation set this base model performed extremely well.
Base Bi-LSTM.02	Same as above	1. Imbalanced training data (17082 records) 2. Imbalanced validation data (5125 records) 3. Imbalanced test data (37961 records) 4. balanced test data (6020 records)	5	11k: 56s 8k: 65s	11k: 0.8070 8k: 0.7247	1. imbalanced test, 11k: 0.8011 2. balanced, 11k: 0.5100 3. imbalanced test, 8k: 0.7218 4. balanced, 8k: 0.3972	1. imbalanced test, 11k: 0.77 2. balanced, 11k: 0.42 3. imbalanced test, 8k: 0.66 4. balanced, 8k: 0.27	This was the base model except training and validation set was significantly reduced.
Base Bi-LSTM.03	Same as above	1. Imbalanced training data (34164 records) 2. Imbalanced validation data (10250 records) 3. Imbalanced test data (37961 records) 4. balanced test data (6020 records)	5	11k: 91s 8k: 90s	11k: 0.8847 8k: 0.8805	1. imbalanced test, 11k: 0.8871 2. balanced, 11k: 0.7399 3. imbalanced test, 8k: 0.8771 4. balanced, 8k: 0.7159	1. imbalanced test, 11k: 0.88 2. balanced, 11k: 0.69 3. imbalanced test, 8k: 0.86 4. balanced, 8k: 0.67	Significant improvement in accuracy of the imbalanced set, proportionally significant improvement in balanced accuracy.
Base Bi-LSTM.04	Same as above	1. Imbalanced training data (68328 records) 2. Imbalanced validation data (20499 records) 3. Imbalanced test data (37961 records) 4. balanced test data (6020 records)	5	11k: 156s 8k: 149s	11k: 0.9579 8k: 0.9487	1. imbalanced test, 11k: 0.9560 2. balanced, 11k: 0.9073 3. imbalanced test, 8k: 0.9500 4. balanced, 8k: 0.9349	1. imbalanced test, 11k: 0.96 2. balanced, 11k: 0.91 3. imbalanced test, 8k: 0.95 4. balanced, 8k: 0.93	Continued improvement though not as much as previous experiment.

Base Bi-LSTM.05	Same as above	1. Imbalanced training data (136657 records) 2. Imbalanced validation data (40998 records) 3. Imbalanced test data (37961 records) 4. balanced test data (6020 records)	5	11k: 256s 8k: 260s	11k: 0.9730 8k: 0.9491	1. imbalanced test, 11k: 0.9721 2. balanced, 11k: 0.9620 3. imbalanced test, 8k: 0.9507 4. balanced, 8k: 0.9380	1. imbalanced test, 11k: 0.97 2. balanced, 11k: 0.96 3. imbalanced test, 8k: 0.95 4. balanced, 8k: 0.94	Larger improvement comparatively than the last experiment. This is 40% of the base training records and 12% of the validation records.
Base Bi-LSTM.05	Same as above	1. Imbalanced training data, clean + Kaggle (20535 records) 2. Imbalanced validation data, clean + Kaggle (12322 records) 3. Imbalanced test data (37961 records) 4. balanced test data (6020 records)	5	11k: 83s 8k: 68s	11k: 0.5684 8k: 0.5512	1. imbalanced test, 11k: 0.7785 2. balanced, 11k: 0.4998 3. imbalanced test, 8k: 0.7517 4. balanced, 8k: 0.4982	1. imbalanced test, 11k: 0.74 2. balanced, 11k: 0.41 3. imbalanced test, 8k: 0.72 4. balanced, 8k: 0.42	It's interesting to note that during training the highest validation accuracy achieved was 57%; however, when running the model against the test set, the imbalanced test set achieved 74% accuracy.
Phase 1: NB lexiconned	Train and test set filtered out to only include 450 words from lexicon, create a TDM using count vectorizer and passed into a Multinomial NB model with no adjustments to the base configuration	Imbalanced stratified train and test data 136k train set					0.90254	Neutral performed very poorly (~0) because not included in lexicon
Phase 1: NB not lexiconned	Train and test set not filtered by lexicon words. Create TDM using count vectorizer and passed into Multinomial NB with no adjustments to base config	Imbalanced stratified train and test data 136k train set					0.89205	Neutral is much better. relatively high precision but low recall. emotion classification are more or less the same, though surprise tanked in f1-score leading to my surprise
Phase 1: NB not lexiconned	Train and test set not filtered by lexicon words. Create TDM using count vectorizer and passed into Multinomial NB with no adjustments to base config	Imbalanced stratified train and test data 17k train set					0.81782	
Phase 1: NB reverse lexiconned	Train and test set filtered out to NOT include 450 words from lexicon, create a TDM using count vectorizer and passed into a Multinomial NB model with no adjustments to the base configuration	Imbalanced stratified train and test data 136k train set					0.41201	
Phase 2: Bert not lexiconned	Multiclass bert with sparse cross entropy. Dense layer of size 201, classification using cls token, a dropout and then dense layer with softmax	1. imbalanced stratified 17k training 5k val	2		0.9742		0.97	
Phase 2: Bert lexiconned training, val, and text set	Multiclass bert with sparse cross entropy. Dense layer of size 201, classification using cls token, a dropout and then dense layer with softmax	1. imbalanced stratified 17k training 5k val	2		0.913		0.90	

Phase 2: Bert reverse lexiconned training, val, and text set	Multiclass bert with sparse cross entropy. Dense layer of size 201, classification using cls token, a dropout and then dense layer with softmax	1. imbalanced stratified 17k training 5k val	2		0.485		046	
Phase 3: Bert not lexiconned	Multiclass bert with sparse cross entropy. Dense layer of size 201, classification using cls token, a dropout and then dense layer with softmax	1. imbalanced stratified mix of clean and kaggle20k training 12k val	2					
Phase 2: bert not lexiconned	Multiclass bert with sparse cross entropy. Dense layer of size 201, classification using cls token, a dropout and then dense layer with softmax	1. imbalanced stratified 17k training 5k val	2				0.98	
Phase 2: bert not lexiconned	Multiclass bert with sparse cross entropy. Dense layer of size 201, classification using cls token, a dropout and then dense layer with softmax	1. imbalanced stratified 34k training 10k val	2				0.98	
Phase 2: bert not lexiconned	Multiclass bert with sparse cross entropy. Dense layer of size 201, classification using cls token, a dropout and then dense layer with softmax	1. imbalanced stratified 68k training 20k val	2				0.98	
Phase 3: NB on carer, kaggle mix	Train and test set not filtered by lexicon words. Create TDM using count vectorizer and passed into Multinomial NB with no adjustments to base config	1. imbalanced stratified mix of clean and kaggle20k training 12k val					0.78189	
Phase3: Bert on carer, kaggle mix	Multiclass bert with sparse cross entropy. Dense layer of size 201, classification using cls token, a dropout and then dense layer with softmax	1. imbalanced stratified mix of clean and kaggle20k training 12k val	2				0.95	
Phase 2: Bert on balanced set	Multiclass bert with sparse cross entropy. Dense layer of size 201, classification using cls token, a dropout and then dense layer with softmax	1. imbalanced stratified 17k training 5k val	2				0.98	