Sentiment Analysis on Movie Reviews

Xinyi Pang, Xiling Zou, Jingyi Lu, Hui Sui

Abstract

Sentiment Analysis has been a popular topic as people leave more text information online to express feelings or share experiences. To better understand people's attitudes from their comments more quickly, we tried to predict the sentiment scores on movies reviews from Rotten Tomatoes. We have tried several models including logistic regression, BERT, BERT with LSTM, BERT with Linear, and BiLSTM with CNN. BiLSTM trained with BERT embedding achieves the best results, with an accuracy of 0.68 and a weighted average of F1-score of 0.69.

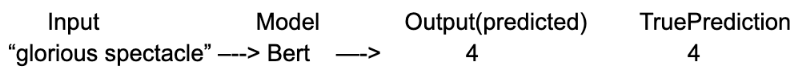
1 Introduction

In the current society, people express their ideas online as reviews and comments everywhere about products and specific topics. For example, people write reviews about products they bought on Amazon, and movie reviews on Rotten Tomato. For these reviews, people may write some long reviews expressing their experience. On the other side, people may also leave some very short comments on Twitter and Tiktok just to express their current feelings.

To better understand people's attitude quickly from the text information, we do research on sentiment analysis on movies reviews from Rotten Tomatoes. We choose this topic because we think our research can be helpful in several application environments, such as improving the recommendation system for some social platforms, and quickly gathering peoples' attitudes towards a certain product.

In other words, we use the movie review text as input, and output a sentiment score from 0 to 4, after training through natural language processing techniques and models we have learned from class. The sentiment score is in order, which means 0 represents the most negative feeling, and 4 represents the most positive feeling. We have built a simple baseline with logistic regression, a strong baseline with BERT, and extensions with BERT with LSTM, BERT with Linear, and BiLSTM with CNN.

Here is an example of using a movie review as input into model BERT, and then get an output of sentiment score 4, and we compare it with the true score with is also 4.



Input Model Output(predicted) TruePrediction
“glorious spectacle” —-> Bert —-> 4 4

Figure 1: example of an input and output

2 Literature Review

In ”Sentiment Analysis on Movie Review Data Using Machine Learning Approach”, Atiqur Rahman and Md. Sharif Hossen (2019) [2] studied the performance of various machine learning techniques on sentiment analysis using movie review data. The machine learning classifiers they tried are Bernoulli Naïve Bayes (BNB), Decision Tree (DE), Support Vector Machine (SVM), Maximum Entropy (ME), and Multinomial Naïve Bayes (MNB). The data was preprocessed by removing URL, punctuations, brackets, numbers, and stop-words, tokenization, case conversion, and finally stemming. They included parts of speech (PoS) tags as part of the features. They chose confusion matrix as their evaluation method and compared the performance on accuracy, precision, recall, f-score individually. The result shown Multinomial NB has the highest accuracy, precision, and f-score, and SVM has the highest recall.

In ”Transfer Learning from LDA to BiLSTM-CNN for Offensive Language Detection in Twitter”[4], the authors emploied a sequentially combined BiLSTM-CNN neural network on offensive language classification. The task is complex due to three reasons: first, the language data is ”atipycal” because of incorrect spellings, false grammar and non-standard language variations; second, the assessment on offensiveness highly dependent on sentence and discourse level semantics, and subjective criteria; thrid, there lacks a common definition of the actual phenomenon to tackle. They use a neural network model combining bidirectional Long-Short term memory(LSTM) and convolutional (CNN) layers to address the problems. They pretrained the BiLSTM-CNN model with Supervised category transfer, Weakly-supervised category transfer, and Unsupervised category transfer. Compared to their SVM baseline, the BiLSTM-CNN model architecture with topic transfer delivers significantly improved results for a multi-class classification with more fine-grained labels sub-categorizing the same tweets into either ‘insult’, ‘profanity’, ‘abuse’, or ‘other’.

”Classifying Tweet Sentiment Using the Hidden State and Attention Matrix of a Fine-tuned BERTweet Model” [1] investigates various sentiment analysis models for tweets. They used a strategy that involves first computing some embeddings to represent the information in the tweet and then applying classifiers to the embedding to determine whether the tweet has a positive or negative sentiment. In addition to several baseline models using classical embedding, like Bag-of-words and Word2vec, and classification algorithms, including Random Forest and AdaBoost, they employed the BERTweet model, which is a variation of BERT pre-trained on tweets data, as the basis of their suggested model. They fine-tuned the pre-trained BERTweet model on the given dataset and investigated the effect of its hidden states and attention matrices. They generated a total of 1024 feature embeddings for each tweet, which they then used to train a multi-layer perceptron with a high dropout rate. The final solution achieves 91.11% validation accuracy.

3 Experimental Design

3.1 Data

This dataset is comprised of tab-separated files with phrases from Rotten Tomatoes. Each Sentence has been parsed into many phrases by the Stanford parser. Each phrase has a PhraseId. Each sentence has a SentenceId. Repeated phrases (such as short/common words) are only included once in the data. There are 8529 sentences parsed into 156060 phrases in the train set and 3310 sentences parsed into 666292 phrases in the test set. The average word length of phrases in both sets is 7. There are five sentiments: 0 - negative, 1 - somewhat negative, 2 - neutral, 3 - somewhat positive, 4 - positive. Below is an example of our data: Phrase is input and Sentiment is output

Phrase	**Sentiment**
you have a case of masochism and an hour and a half to blow	0
by half-baked setups and sluggish pacing	0
the failure of the third Revenge of the Nerds sequel	1
nearly incoherent	1
what you'd expect	2
its charms and its funny moments but not quite enough of them	2
that excites the imagination and tickles the funny bone	3
strives to be intimate and socially encompassing but	3
Hoffman 's performance is great	4
crisp and purposeful without overdoing it	4

After further split of into train set and development set, the size of our dataset is shown below:

dataset	**number of phrases**
train	124848
development	31212
test	66292

There is a skewed distribution over the labels, as shown in the bar plot below. The majority of the class labels were 2, strong positives and strong negatives made up only a small portion of the train set.

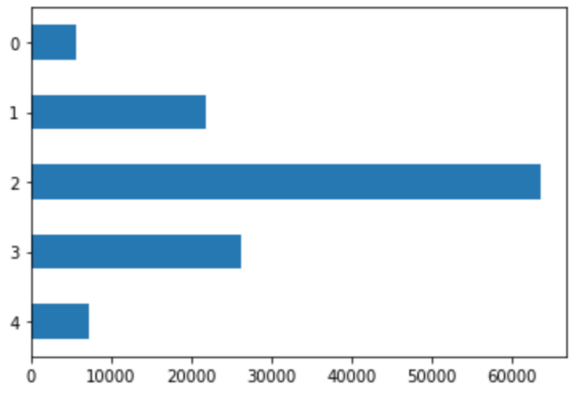


Figure 2: class labels distribution

3.2 Evaluation Measure

We use F1 score as our evaluation metric. F1 score is a measure of accuracy, calculated from the precision and recall. In our case, our output metric is an array that contains the F1 score for each label, and a macro F1 score to measure the overall performance of our prediction. In macro F1 score, each class has the same weight in the average, so that there is no distinction between highly and poorly populated classes. We think each class in our case should have the same weight, so we use macro F1 score to measure the overall performance. In our project, we are predicting the sentiment score from movie review, so we are doing multi-class classification. F1 score is a good evaluation metric for it, as proposed in research paper. It gives us information from both the recall and precision, and can give us insights about whether a model works better than another model.
The following equations were used:

$$f1\ score = 2 * \frac{Precision * Recall}{Precision + Recall} = \frac{TP}{TP + 0.5 * (FP + FN)}$$

$$macro\ f1\ score = \frac{sum(f1\ scores)}{number\ of\ classes}$$

In ”Sentiment Analysis on Movie Review Data Using Machine Learning Approach” by Atiqur Rahman and Md. Sharif Hossen (2019) [2], F1-score was also used as one of the performance metrics.

3.3 Simple Baseline

We use bag-of-words as features and implemented a logistic regression as the simple baseline model. We chose to perform the multiclass classification using one-versus-rest (OvR) scheme, regularized with L2 penalty and solved with 'liblinear' algorithm. On the development data set, we tuned the hyper parameter C, which represents the inverse of regularization strength. We tried values ranging from 0.1 to 3.0, and C = 2.5 gives the best prediction on development data. The best model has an accuracy of 0.7107 on training data and 0.6420 on development data.

4 Experimental Results

4.1 Published Baseline

We refer to this study[3] to extract features using vanilla BERT in our strong baseline model. In the published paper, BERT is utilized in the published work to extract embedding vectors from twitter text, and the feature vectors are fed into a logistic model to assess whether a tween belongs to a medical or non-medical domain. Data in this work are tweets from certain manually picked official profiles of authoritative news or websites, such as CNN, BBC News, etc. Only tweets with a full statement / article and a URL linked to the original domain that falls under the "health/medical" category are labeled as medical. They achieved an accuracy of 0.886 and F1-score of 0.879. BERT is an algorithm that computes the embeddings of a text by taking into account the words in relation to all the other words in a context, which can be helpful in assessing the sentiment of a sentence. Similarly, we use BERT as an encoder and feed the output into logistic regression. Sentences are tokenized by BertTokenizer then send to the BertModel to generate a vector of length 768 for each sentence. Then, those feature vectors are feed to logistic regression to fit and predict. We achieved a training accuracy of 0.634 and a development accuracy of 0.623. Our results are not directly comparable since we are using different datasets. Possible reasons for having a lower level of F1-score are first, we are performing a multiclass classification with 5 labels while the published work are only perform binary classification. Furthermore, the domain of a tweet in the original work can be highly related to certain keywords or phrases, making it easier to identify than sentiment classification.

4.2 Model Extension1: Fine Tuning BERT

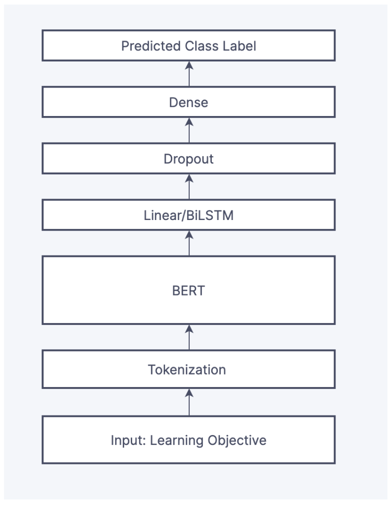
Our first model extension is to further improve the BERT model. Comparing with the published baseline that uses logistic regression to fit the pre-trained BERT embedding, we decided to use neural network to process the representation vectors encoded by BERT. As mentioned in previous section, the BERT we used is trained on sources like Wikipedia, which is not similar from the movie review data we used. Therefore, the motivation of this extension is that to make the embedding fit the

data better and further interpret the embedding using more complex model.
We chose two type of neural network to add on top of the BERT model, multi-layer perceptron (MLP) and Bidirectional Long short-term memory (BiLSTM). We also tried two loss functions, the default CrossEntropyLoss and the weighted CrossEntropyLoss to alleviate the class imbalance issue. During the model training process, all the weights in BERT and the added neural network on top of the BERT were trained for one epoch. Then all the weights in BERT were frozen, while the added neural network was trained for another 30 epochs. We used Adam as optimization algorithm and CosineAnnealingWarmRestarts as scheduler to adjust the learning rate as epoch increasing. The accuracy and F1-score are listed in the table below. We can see that the BERT+BiLSTM+Weighted Loss achieve the highest macro F1-score and weighted F1-score. The architecture of the model is shown in Figure 3.

4.3 Model Extension2: BiLSTM-CNN

Inspired by ”Transfer Learning from LDA to BiLSTM-CNN for Offensive Language Detection in Twitter”[4], we came up with our second model extension, BiLSTM+CNN. After examining the performance of the Extended BERT model, we wondered that whether using other embedding method instead of BERT will achieve similar performance. Thus, in this model extension, we chose GloVe.6B.300d as our embedding. Then, the embedding are trained with the following BiLSTM-CNN structure. CNN captures feature space’s structural locality and is shift invariant with pooling but are not robust to noisy data which can be solved by using BiLSTM. Thus, we hypothesized that BiLSTM-CNN model is also suitable for this research. The model was trained for 5 epoch and use the weighted CrossEntropyLoss. The accuracy and F1-score are listed in the table below. We can see that BiLSTM-CNN can achieve a similar but a little worse result than BERT+BiLSTM+Weighted Loss with less computation needed. The architecture of the model is shown in Figure 3.

Section	Model	Accuracy	Macro F1	Weighted F1
Simple Baseline	Bag-of-words+Logistic	0.64	0.51	0.62
Published Baseline	BERT+Logistic	0.62	0.46	0.60
Extension1	BERT+Linear+Unweighted Loss	**0.69**	0.61	0.69
Extension1	BERT+Linear+Weighted Loss	0.68	0.61	0.68
Extension1	BERT+BiLSTM+Unweighted Loss	0.68	0.61	0.68
Extension1	BERT+BiLSTM+Weighted Loss	0.68	**0.62**	**0.69**
Extension2	GloVe+BiLSTM+CNN	0.67	0.59	0.67



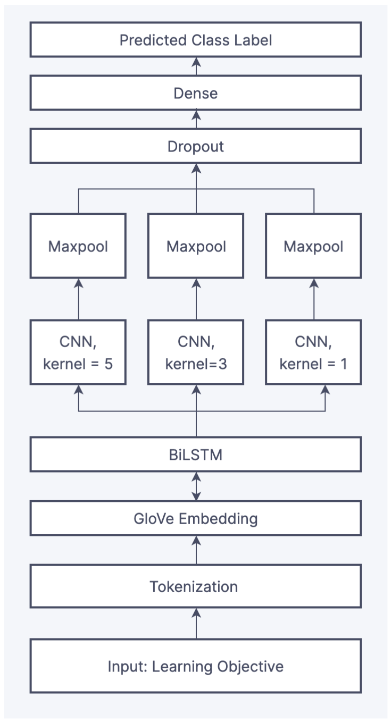


Figure 3: Architecture of Fine-tuning Bert (left) and BiLSTM-CNN (right)

4.4 Error Analysis

From our best results, we check how each class is predicted incorrectly in the following table. In the table, we can see that our model is doing well in predicting extremely negative feeling (class 0) and extremely positive feeling (class 4). For these two classes, our model gets the least incorrect predictions. However, as feeling becomes more neutral, we see more incorrect predictions, and the neutral class (class 2) gets the most incorrect predictions. We think this is reasonable, as when people may express whether they like or dislike some part of the movie a little bit, they may still tend to be objective and then give a middle score, which is score 2 (class 2). When they show really clear extreme feelings in the reviews, their score may be highly influenced by their emotion, and they will just give out the extreme score (class 0 and class 4). Also, we can see that for each prediction class, the most incorrect predictions show up in the nearby prediction class(es). For example, we can see that it is most easily for the model to incorrectly predict class 2 as class 1. This is reasonable because our prediction class is in order, so the the classes near each other share closer feelings. And people may just write down the part they like or hate the most in reviews, but also think of other parts when they give a score.

True	Predict 0	Predict 1	Predict 2	Predict 3	Predict 4
0	30541	632	30	9	0
1	611	29466	994	141	3
2	125	2318	26687	2018	64
3	7	201	1366	28717	921
4	0	10	18	625	30559

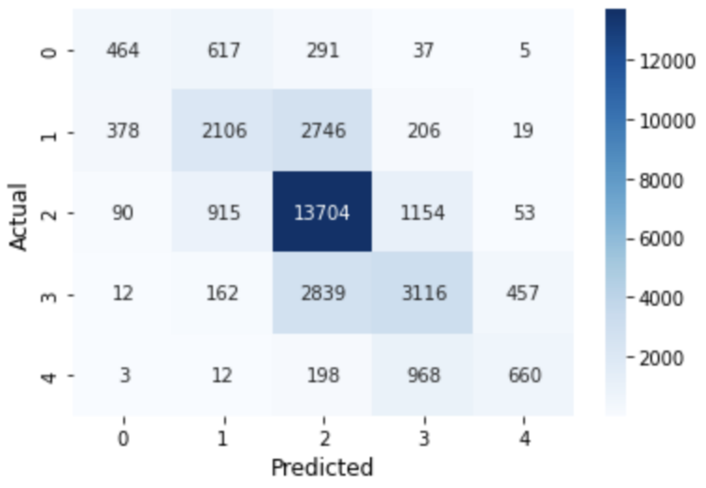
5 Conclusions

In this work, we explored different approach to do sentiment analysis on movie reviews. We started with a simple baseline by passing bag of words into a Logistic Regression. Then, we tried a strong baseline using vanilla BERT and Logistic Regression. Next, we extended the strong baseline using MLP and BiLSTM on top of the BERT model. It has the option to do the default CrossEntropyLoss and the weighted CrossEntropyLoss. Next, we tried another way to extend the strong baseline model by using GloVe.6B.300d embeddings and feed into BiLSTM-CNN model. We found that BiLSTM trained with BERT embedding and weighted cross entropy loss achieves the highest results, with an accuracy of 0.68 and a weighted average of F1-score of 0.69. The highest level of accuracy achieved when performing the same task on Kaggle is 0.71, which is 4.4% higher than our model. Further research can be focusing on preprocessing and alternative feature extraction techniques to more efficiently preserve the contextual and ordering information. We also paln to evaluate our proposed architecture on different tasks such as news categorization and detecting offensive language on social media.

6 Appendicies

Model: Simple baseline

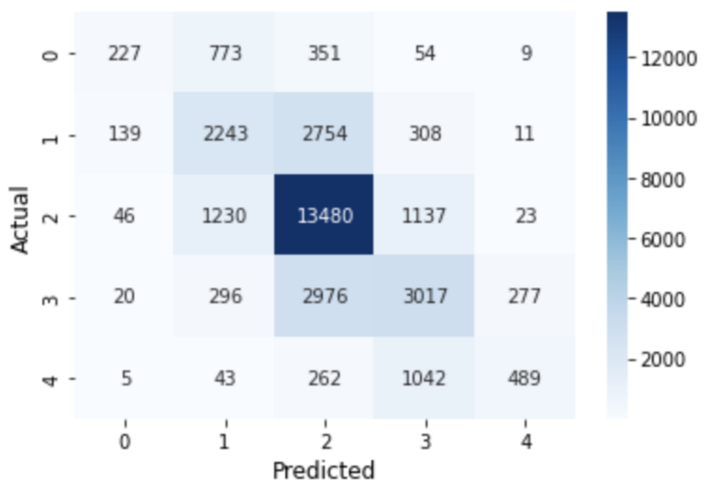
```
  Confusion Matrix
```



```
  Classification Metrics

                precision    recall  f1-score   support

             0       0.49      0.33      0.39      1414
             1       0.55      0.39      0.45      5455
             2       0.69      0.86      0.77     15916
             3       0.57      0.47      0.52      6586
             4       0.55      0.36      0.43      1841

      accuracy                           0.64     31212
     macro avg       0.57      0.48      0.51     31212
  weighted avg       0.62      0.64      0.62     31212
```

Model: Published Baseline

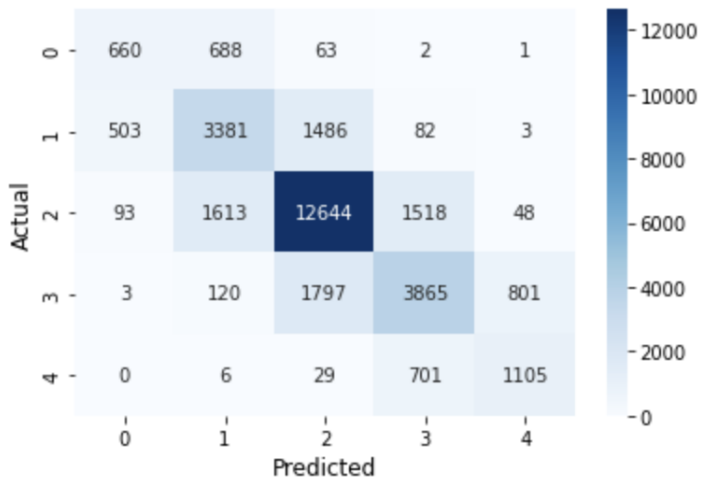
```
Confusion Matrix
```



```
Classification Metrics

              precision    recall  f1-score   support

           0       0.52      0.16      0.25      1414
           1       0.49      0.41      0.45      5455
           2       0.68      0.85      0.75     15916
           3       0.54      0.46      0.50      6586
           4       0.60      0.27      0.37      1841

    accuracy                           0.62     31212
   macro avg       0.57      0.43      0.46     31212
weighted avg       0.61      0.62      0.60     31212
```

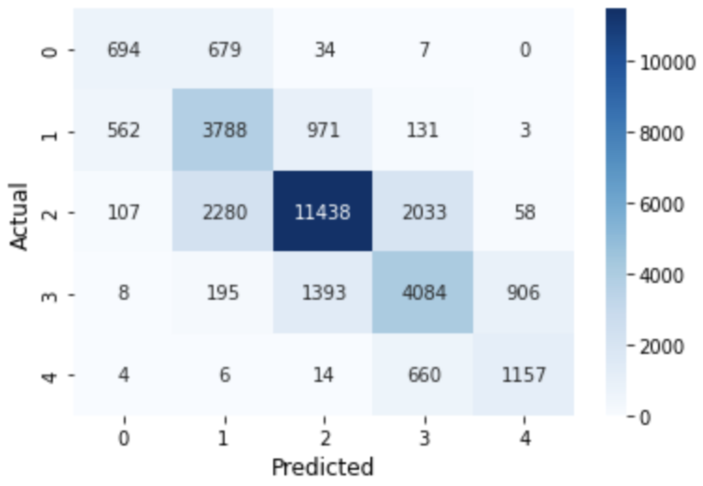
Model: BERT+Linear+Unweighted.

```
Confusion Matrix
```



```
Classification Metrics

              precision    recall  f1-score   support

           0       0.52      0.47      0.49      1414
           1       0.58      0.62      0.60      5455
           2       0.79      0.79      0.79     15916
           3       0.63      0.59      0.61      6586
           4       0.56      0.60      0.58      1841

    accuracy                           0.69     31212
   macro avg       0.62      0.61      0.61     31212
weighted avg       0.69      0.69      0.69     31212
```

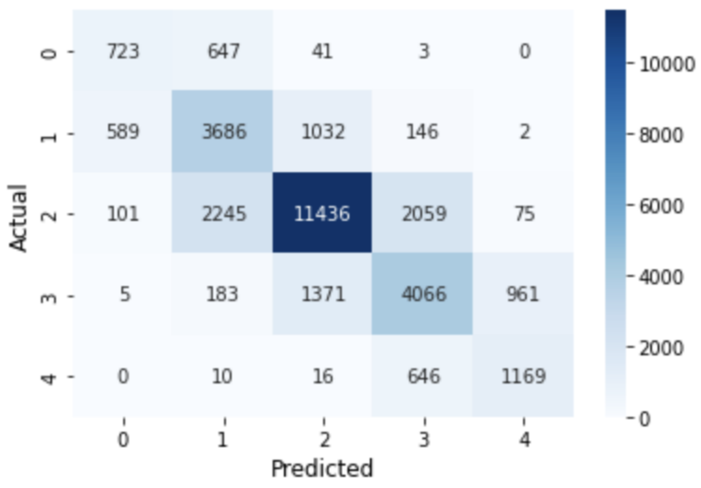
Model: BERT+Linear+Weighted.



```
Classification Metrics

              precision    recall  f1-score   support

           0       0.50      0.49      0.50      1414
           1       0.55      0.69      0.61      5455
           2       0.83      0.72      0.77     15916
           3       0.59      0.62      0.60      6586
           4       0.54      0.63      0.58      1841

    accuracy                           0.68     31212
   macro avg       0.60      0.63      0.61     31212
weighted avg       0.70      0.68      0.68     31212
```

Model: BERT+LSTM+Unweighted.

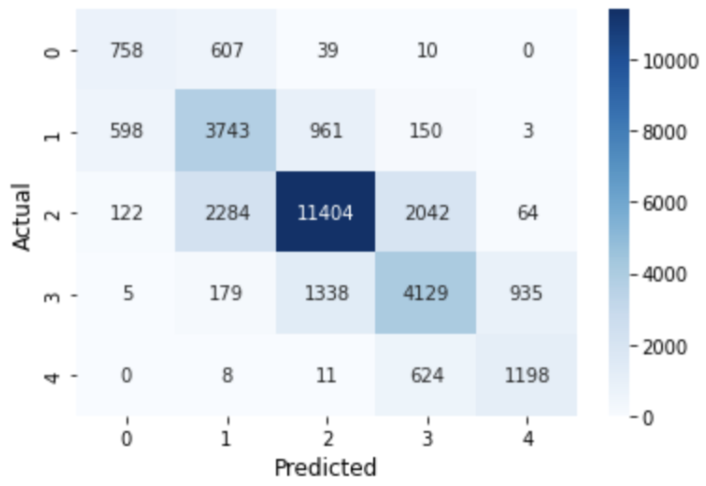
```
Confusion Matrix
```



```
Classification Metrics

              precision    recall  f1-score   support

           0       0.51      0.51      0.51      1414
           1       0.54      0.68      0.60      5455
           2       0.82      0.72      0.77     15916
           3       0.59      0.62      0.60      6586
           4       0.53      0.63      0.58      1841

    accuracy                           0.68     31212
   macro avg       0.60      0.63      0.61     31212
weighted avg       0.69      0.68      0.68     31212
```

Model: BERT+LSTM+Weighted.

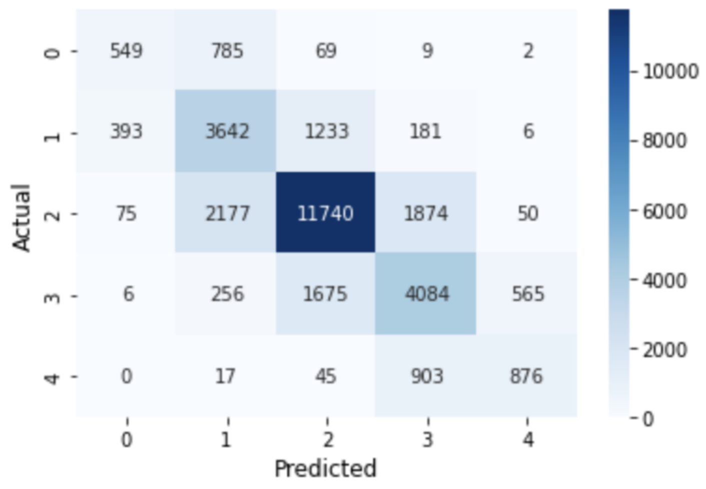
```
Confusion Matrix
```



```
Classification Metrics

              precision    recall  f1-score   support

           0       0.51      0.54      0.52      1414
           1       0.55      0.69      0.61      5455
           2       0.83      0.72      0.77     15916
           3       0.59      0.63      0.61      6586
           4       0.54      0.65      0.59      1841

    accuracy                           0.68     31212
   macro avg       0.61      0.64      0.62     31212
weighted avg       0.70      0.68      0.69     31212
```

Model: BiLSTM+CNN.

```
Confusion Matrix
```



```
Classification Metrics

              precision    recall  f1-score   support

           0       0.54      0.39      0.45      1414
           1       0.53      0.67      0.59      5455
           2       0.80      0.74      0.77     15916
           3       0.58      0.62      0.60      6586
           4       0.58      0.48      0.52      1841

    accuracy                           0.67     31212
   macro avg       0.61      0.58      0.59     31212
weighted avg       0.68      0.67      0.67     31212
```

References

[1] Tommaso Macrì, Freya Murphy, Yunfan Zou, and Yves Zumbach. Classifying tweet sentiment using the hidden state and attention matrix of a fine-tuned bertweet model, 2021.

[2] Atiqur Rahman and Md Sharif Hossen. Sentiment analysis on movie review data using machine learning approach. In *2019 International Conference on Bangla Speech and Language Processing (ICBSLP)*, pages 1–4. IEEE, 2019.

[3] Kevin Roitero, Cristian Bozzato, Vincenzo Della Mea, Stefano Mizzaro, and Giuseppe Serra. Twitter goes to the doctor: Detecting medical tweets using machine learning and bert. In *SIIRH@ECIR*, 2020.

[4] Gregor Wiedemann, Eugen Ruppert, Raghav Jindal, and Chris Biemann. Transfer learning from lda to bilstm-cnn for offensive language detection in twitter. *arXiv preprint arXiv:1811.02906*, 2018.