iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 **Issue:** IV **Month of publication:** April 2022

DOI: https://doi.org/10.22214/ijraset.2022.42085

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue IV Apr 2022- Available at www.ijraset.com

Multi-class Classification of Twitter Sentiments using Frequency based, LSTM and BERT Methods

Pankaj M Thakur[1], Tejas M[2]
[1]*Department of Electronics and Communication Engineering, Siddaganga Institute of Technology, Tumkur 572102*
[2]*Department of Information Science and Engineering, Siddaganga Institute of Technology, Tumkur 572102*

Abstract: ***Twitter is one of the most popular social media applications used by people of diverse age groups. Tweet has become an important mode of communication. Tweets are unstructured and often consist of different kinds of data tokens like mentions, hash tags, emoticons etc. Analysing the sentiment of this kind of unstructured data is challenging. In this paper, the experimentation of multiclass twitter sentiment analysis is presented, using three different approaches. Starting with the simple approach of frequency based feature extraction method, that helps capture the features from a text document by counting the frequency of the occurrence of words, to more advanced approaches like LSTM and BERT have been used. Different pre-processing techniques and different classifiers are explored in this paper. The performance of all the three methods is evaluated using metrics (precision, recall and f1 score). At the end the results of all three approaches are discussed and compared against one another.***

Keywords: NLP, BERT, LSTM, MLP Classifier, Multiclass Classification, Embeddings, Tweets.

I. INTRODUCTION

Social Media has become a very indispensable part of our lives. People of all ages are using various social media apps for different purposes. These platforms or apps have done a ground-breaking job in connecting people of similar interests, backgrounds, hobbies etc. People share their thoughts opinions, views online with their community, and the world. While it can be seen as a medium of socializing with the birds of same feather across different cities, states and even borders, it can also be a peril and an opportunity for some, to spread hate, insult others, give fruition to malicious propagandas. Online hate is at an all-time high, despite the technologies, method, practices that are in place to fight it. Twitter is one of the most popular platforms used by millions of users, from teens to senior citizens. Hate on twitter catches fire very easily and it can be very depressing for the people at the receiving end. There must be an accurate mechanism to analyse the sentiment of a tweet before it is posted on the platform.

Since we live in a world that is essentially driven by data, analysing the sentiment of a tweet can also be beneficial to the organizations, content creators etc. to gauge the response of the audience to their products, services or content. Organizations can also use this to gather critical feedback about problems or issues in newly released products [1]. People often tweet to express their pleasure or displeasure for products or services they use. Tweets are also used by brands to market their products or services. In this landscape analysing sentiment(s) in tweets becomes a very important task and can determine key trends and patterns of customer behaviour towards their said product or services. Public opinion data can be used to determine public awareness, to predict outcomes of events, and to infer characteristics of human behaviours (Cody et al, 2016) thus, Twitter is an ideal source for eliciting information about societal interest and general people's opinions [2]. We can find direct opinion and comparisons in tweets [4]. The text fragment with subject and sentiment lexicons can be identified by NLP, to carry out sentiment classification, instead of classifying the sentiment of whole text based on the specific subject [5]. Over the years, many approaches have been proposed. The computational classification of text at early stages was heavily dependent on the knowledge mining approaches [14].On the research by Blenn et al [11], a system that worked through a combination of grammatical analysis with traditional word frequency analysis was proposed. Yang et al. [15] obtained sentiment of a single sentence and examined the sentence level challenges and discussed the ways to deal with these.

The current paper focuses on different methodologies that range from simple to more advanced. The idea is to see how different methods fare at analysing textual data. Each method requires its own set of steps for pre-processing of the data, on top of general cleaning up that is done. Both Long short term memory (LSTM) and Bidirectional encoder representation of transformers (BERT) make use of convolutional neural network where LSTM and BERT are one of the layers.

This paper consists of 6 sections. Section II consists of a brief description of the dataset and provides an insight into how the dataset is divided into training and the test set. Section III describes the pre-processing steps implemented on the dataset. Section IV of the paper discusses the three approaches or methodologies that have been used for the task of analysing sentiments that is at hand. These methodologies involve steps involving pre-processing, model building and classification. Section V focuses on the parameters on which the results of all the three approaches and gives a brief comparison of these parameters for different approach. Finally section VI draws inference with respect to the methodologies used.

II. DATASET

The dataset was obtained from kaggle; it consists of a total of 44955 tweets. Around 92% of the total tweets, which is equivalent to 41157 tweets, are used as the training set. The remaining 3798(approximately 8.4%) tweets are used as the test set. The tweets had been pulled from twitter and manually tagged by the kaggle user. Go et al. [6] found that average length of tweets is 14 words, and average length of sentence is 78 characters. The dataset has 6 columns:

1) *Username:* Name of the user represented as coded number to avoid privacy concerns.
2) *Screen Name:* Name of the user on twitter or twitter user name also represented as a coded number to avoid privacy concerns.
3) *Location:* Location of the user from where the tweet was posted.
4) *Tweeted At:* The date when the user had tweeted the concerned tweet.
5) *Original Tweet:* The tweet that the user had posted on twitter. This is the tweet/column for which the sentiment has to be analysed.
6) *Label:* The manual tag assigned to each tweet. One of the five labels has been assigned to each tweets: a) Extremely Negative b) Negative c) Neutral d) Positive e) Extremely Positive.

Only the columns Original Tweet and Labels were used and the rest of the columns were stripped as they do not contribute towards the learning of the model.

III.PRE-PROCESSING

The raw tweets that have been pulled from the twitter cannot be used as such for sentiment analysis. Since there can be some unnecessary data in each tweet that is not of any value to the model in any way, but is only a hindrance, the tweets were cleaned.
The following pre-processing operations were performed on the dataset:

1) Removal of all the urls from the tweet using regular expression.
2) Removal of hash tags using regular expression.
3) Removal of user mentions example @JohnDoe as it does not contribute to the sentiment.
4) Removal of all the digits, extra white spaces, emoticons, leading and trailing spaces.
5) Removal of stop words (words like ‘and’, ’for’ etc. are considered stop words).

Table I
Comparison Of Tweets With And Without Pre-Processing

Without Pre-processing	With Pre-processing
1. @JohnDoe@JenDoe We should watch this movie!! https://www.netflix.com/ca/title/81217330 #thriller#thecourier	1. we should watch this movie
2. @twitteruser12367 I send you my heartfelt wishes, all the best ☺ #bestwishes	2. i send you my heartfelt wishes all the best

A. *Label Encoding*

Each of the 5 labels is encoded from 0 to 4, the following encoding scheme is followed:

1) 'Extremely Negative':0
2) 'Negative':1
3) 'Neutral':2
4) 'Positive':3
5) 'Extremely Positive':4

B. Train Validation Split

The training dataset of 41157 tweets is split into train and validation dataset. The validation set is essentially used to tune out the hyper-parameters so that when the model comes across unseen data in the test set, it can perform better. 10% of training set is split to get the validation set, which left the train count to 36280 tweets and validation count to 4032 tweets.

IV. METHODOLOGIES

A. Method1 (Frequency Based)

The first method focuses on using the frequency (count) of the word as a feature to determine if the sentiment of the entire sample (tweet) is one of the 5 mentioned above. In the training set the model learns which words are negative based on the labels assigned and the collection of words that make the sample negative.

1) *Porter Stemmer:* Stemming is a technique of reducing words to their root/base word. For example 'retrieves', 'retrieved', 'retrieving' is reduced to its root word 'retrieve'. Stemming lowers the inflection of words to their root words and hence it reduces the number of words for text normalization as it aides in pre-processing of the samples. Porter Stemmer (or Porter Stemming) is one of the most popular methods of stemming that was proposed in 1980, which banks on the idea that the suffixes in English language are made up of smaller, simpler suffixes. Example : In the words 'agreed', 'agreeing' and 'agree' the suffixes 'eed', 'eeing' are made up of the simple suffix 'ee' and the words are reduced to the root word 'agree' which uses the simple suffix. For this method, Porter Stemmer is used from the sklearn.feature_extraction module.

2) *Count Vectorizer:* Count Vectorizer is a method used to convert words into numbers or textual data into numerical data based on the frequency of the word in a sample. Since machines cannot understand characters and words this is one of the simplest methods to extract features from textual samples.

Example:
Text = *"This is sentiment analysis project, it is filled with sentiment."*

Word	This	is	sentiment	analysis	project	it	filled	with
Count	1	2	2	1	1	1	1	1

As seen in the example the words 'sentiment' and 'is' appear twice and hence their count is 2, while all the other words have a count of 1.
Let's take another example of 2 inputs and see how count vectorizer is applied to them.
Text1 = "The sun rises in the east."

Text2 = "I prefer the moon over the sun."

In the sentences above there are 10 unique words. The 2 sentences can be represented as follows:

Word	The	sun	rises	in	east	I	prefer	moon	over
Text 1	2	1	1	1	1	0	0	0	0
Text 2	2	1	0	0	0	1	1	1	1

This way from a corpus of samples, unique words are extracted and each sample is converted into a matrix as shown above which represents the count of the unique words present in the sample. Each word is taken as a feature and for the experimentation, maximum features was set to 10000 features. This means that each sample has 10000 features and for the word(s) present in the sample the value of that feature would increase by 1 for each occurrence and will be 0 for all the features (words) not present in the sample (tweet).

3) *Multi-Layer Perceptron (MLP) Classifier:* Multi-Layer Perceptron (MLP) is a supervised machine learning algorithm that relies on the underlying neural network to perform classification. Multi-Layer Perceptron is a feed forward artificial neural network model, that takes in multiple inputs and maps them to a set of classes(labels) with one or more non-linear hidden layers in between the input layer and the output layer. The multilayer perception neural network is built up of simple components [3].Each layer of the multiple-layers in an MLP is fully connected to the following layer. The hidden layers are called non-linear because the nodes (neurons) of the MLP (except the input layer) are activated by the non-linear activation functions. MLP can classify data that is not linearly separable with a high accuracy because it can learn complex non-linear functions.

Fig. 1 demonstrates the layers of the model built for method 1.

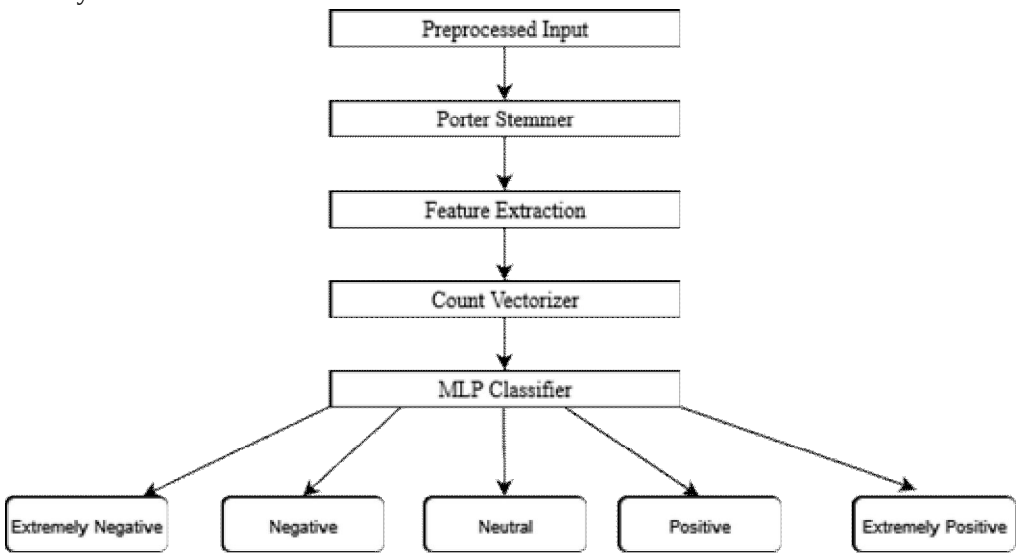


Fig. 1 Method 1(Frequency Based)

B. Method 2(LSTM Based)

1) *Long Short-term Memory (LSTM):* Long short-term memory is an artificial recurring neural network (RNN) architecture, which has become very popular these days in Natural Language Processing (NLP) applications. LSTM is used in the field of deep learning. The LSTM architecture does not make use of conventional feed-forward neural networks but it has its own dedicated feedback connections. LSTM can process a sequence of data apart from processing single data input points. In the context of the approach that is being discussed, LSTM can process the whole tweet (sample) apart from processing each word in a tweet. As the name suggests, LSTMs are good in remembering information for long time [7]. The traditional recurring neural networks (RNN) which is a precursor to the LSTMs had a drawback of vanishing gradients. LSTM can also solve artificial long time lag tasks [8] that have never been solved by recurring neural networks.

When in an iteration of the model, the weight receives a very small update which is close to zero over multiple recent iterations. This is because the gradient is vanishingly small in those iterations. In such cases due to vanishing gradient problem the neural network might stop training.

The most basic LSTM unit is composed of a cell; forget gate, an input gate and an output gate.

- A forget gate controls what information is to be forgotten when new information enters the network.
- An input gate controls what new input is to be encoded in the cell state when new information enters the network.
- An output gate controls the encoded output information to be sent out into the next time step.
- Long term information and dependencies are encoded into a cell.

LSTMs have direct access to forget gate's activation function and they use the unique property of additive gradient structure which encourages gate updates on each time step, no matter how small and hence prevent the vanishing gradient problem. Fig. 2 depicts an LSTM cell.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue IV Apr 2022- Available at www.ijraset.com

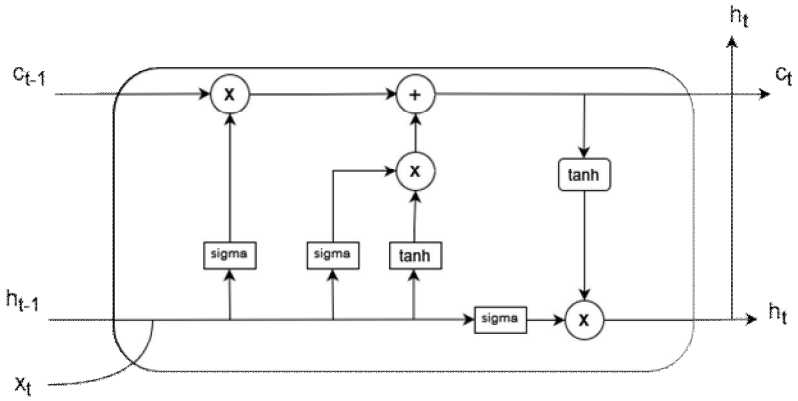


Fig. 2 LSTM Cell

2) *Tokenization and Padding of sequences:* Tokenization and padding of sequences are the pre-processing techniques used for an LSTM based model. From the sample of tweets presented as the training set a vocabulary of words is created as a key-value pair, where to each word a numerical value is assigned. Then each word of the sample in the training set is encoded with the corresponding numerical value. Any word that is not a part of the vocabulary is assigned an out of vocabulary <oov> token. Since each sequence in a sample will be of different length and each sample is of varying length, a maximum length of the sequence is set. Any word that exceeds the maximum length is truncated. There is an option of truncating from either the beginning or the end of the sequence. The sequences that are shorter than the maximum length are padded with zeros. In the current approach the zeros are padded from the left, i.e. from the beginning of the sequence, for sequences that are shorter than the maximum length. Fig. 3 demonstrates the layers in the model for method 2.

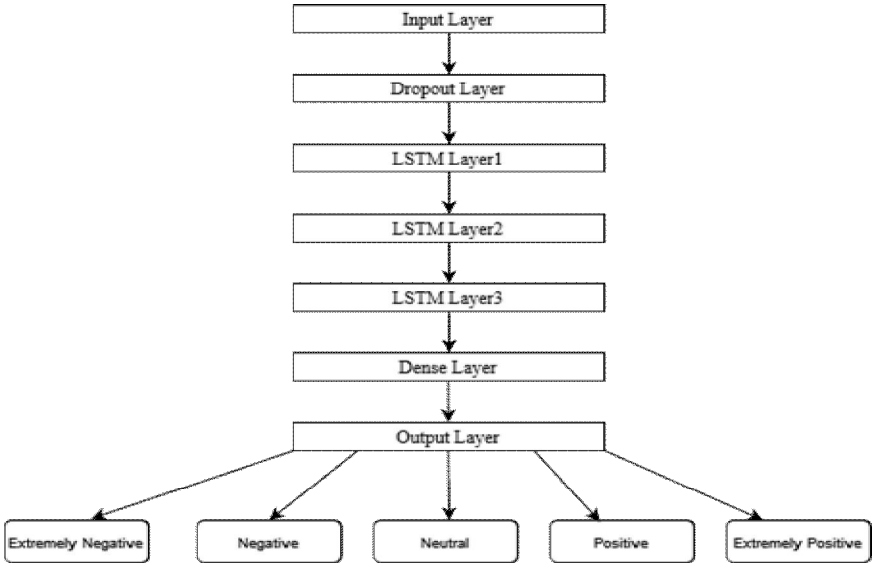


Fig. 3 Flowchart of Model2 (LSTM Based)

3) *The Model:* Firstly embeddings of the tokenized, padded sequences are generated. The maximum matrix size is set to 300. Embeddings are the feature vector of each word in the sequence. These embeddings are learnt from the training set and help determine if the word is aligned towards positive or negative sentiment.

The final model comprises of the following in chronological order:

- An input layer which receives the tokenized, padded sequences.
- A Dropout layer.
- Three LSTM layers with activation function ‘relu’.
- A Dense layer with activation function ‘softmax’.
- An Output Layer.

C. Method 3 (BERT Based)

1) *One Hot Encoding:* After the data set is pre-processed as described in the pre-processing steps mentioned above, an additional pre-processing step is performed on the cleaned dataset in this approach. The labels of training, validation and test set are applied with one hot encoding. One hot encoding creates additional binary column for each label (encoded or not encoded) and then it assigns 1 or 0 to the respective columns with respect to the label.

Example:
If a sentence is labeled as Neutral;

Extremely Negative	Negative	Neutral	Positive	Extremely Positive
0	0	1	0	1

The column labelled 'Neutral' will have a 1 and all others will be zero.

One hot encoding is preferred over integer encoding with some models, because labels are categorical data and categorical data does not have a natural order. Assuming an order for such data can lead to lower performance in some cases, hence one hot encoding binary technique is used.

2) *BERT:* BERT stands for Bidirectional Encoder Representations from Transformers, which is a language representation model. BERT is designed to pre-train language representations. Unlike recent language representation models (Peters et al., 2018a; Radford et al., 2018), BERT is designed to pre-train deep bidirectional representations from unlabelled text by jointly conditioning on both left and right context in all layers[9]. Pre-trained BERT model can be leveraged to create state-of-the-art models for wide range of deep learning applications. BERT is based on attention mechanism and transformer coding structure [10].

3) *BERT Tokenizer:* In the current approach BERT tokenizer was used.

Each sequence is tokenized, i.e. each sentence is broken down into words. In BERT, the sequence is passed to a BERT tokenizer.

a) The tokenization is performed by an algorithm called wordpiece tokenizer. It is different from other tokenizers because it splits each word into a sub-word. In regular tokenizers, if an unknown word is encountered, it is assigned an unknown [UNK] token. When the model comes across many unknowns when compared to the vocabulary of the training set, much information is lost and the model performs poorly. But wordpiece tokenizer splits the word, for example the word 'eating', will be split into 'eat' and '##ing', which are more commonly used words.
b) [CLS] is added to the beginning of each sentence to denote the start of the string.
c) [SEP] is added at the end of each sentence to denote the end of the string.
d) [PAD] is added when the length of the sequence is smaller than the maximum set length.
e) All the tokens are converted into corresponding ids.

Pre-trained BERT tokenizer is used to tokenize training, validation and test data into input ids and attention masks. The BERT model is defined and maximum input length is set. Other hyper-parameters that are set are:

- Epoch
- Batch size
- Learning rate

Adam optimizer is used to accelerate the gradient descent. It makes use of 2 gradient descent methodologies. The loss function used is categorical cross-entropy. Categorical cross-entropy is widely used for multi-class classification problems. Categorical accuracy is the metric measured.

The model consists of an input layer. The input ids and the attention masks are generated for the input and then the input ids and the attention masks are supplied to the BERT model. The next layer is the dropout layer followed by a dense layer and an output layer.

- Input Layer
- BERT Model
- Dropout Layer
- Dense Layer with activation function 'relu'
- Output Layer

Fig. 4 depicts the different layers in the model for method 3.

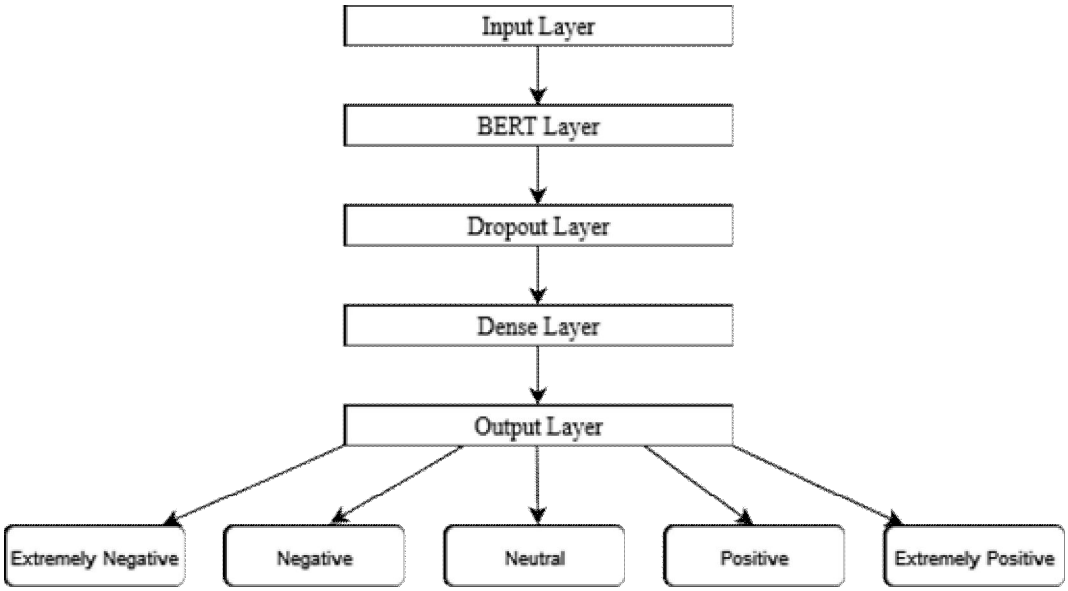


Fig. 4 Flowchart for Method 3(BERT Based)

V. RESULTS AND DISCUSSION

As discussed, three different methodologies have been used and the same data set is made use of, by each approach. The following are the metrics that were measured for all the three methodologies.

Accuracy: Accuracy is the ratio of total number of predictions that were correct to the total number of predictions.

$$Accuracy = (TP + TN)/(TP + TN + FP + FN)$$

Precision: Precision is the ratio of True positives to total number of positives (sum of true and false positives).

$$Precision = TP/(TP + FP)$$

Recall: Recall is the measure of total number of true positives against the sum of true positives and false negatives.

$$Recall = TP/(TP + FN)$$

F1 score: F1 score is a harmonic mean of Precision and Recall.

$$F1\ Score = 2 \times (Precision \times Recall)/(Preciscion + Recall)$$

These metrics for each of the three methods were measured as follows:

A. Method1 (Frequency Based)

For the method, count vectorizer was used to extract feature and classifier used was MLP classifier. Though simple the method gave very promising results with an accuracy of 64%. The precision, recall and the f1 score of each label can be seen from table II. Confusion matrix is of size $\ell \times \ell$, where ℓ is the number of different label values [12].

Fig. 5 represents the confusion matrix for each label. Labels 2 and 4, i.e. 'Neutral' and 'Extremely Positive' have the best F1 score of 0.70, while label 2 has the best recall of 0.72 and label 4 has the best precision of 0.74. Table II captures the values of the 3 metrics: F1 score, precision and recall for all the labels for method 1.

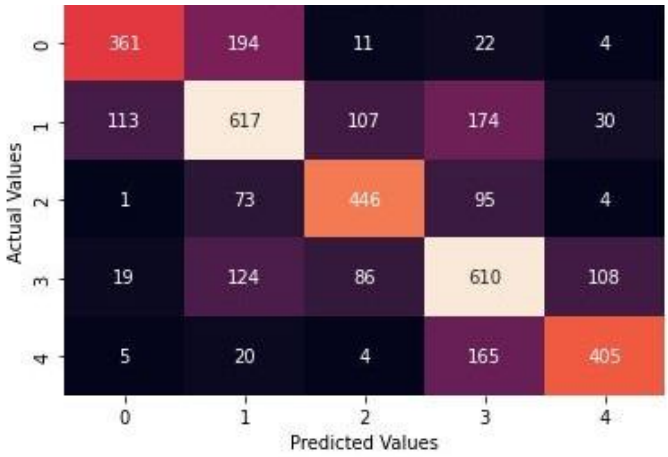


Fig. 5 Confusion Matrix for Method 1(Frequency based)

Table II
Evaluation Metrics for Method 1

	Precision	Recall	F1 score
Extremely Negative	0.72	0.61	0.66
Negative	0.60	0.59	0.60
Neutral	0.68	0.72	0.70
Positive	0.57	0.64	0.61
Extremely Positive	0.74	0.68	0.70

B. Method 2

The second method, the LSTM method fared better than the first method in terms of the results. The overall accuracy on the test set was 73%, which is 9% higher than the first method. The labels 'Neutral' and 'Extremely Positive' gave the best results in terms of F1 score with the value of 0.79 each. Again, in terms of recall the same parameters had the best value of 0.75 each. Following the same trend 'Neutral' and 'Extremely Positive' demonstrated the highest precision of 0.83 each. The label 'Negative' had the lowest values of 0.67, 0.65 and 0.69 respectively for each of F1 score, recall and precision. All labels performed the best on precision, while they had the lowest values for recall. Fig. 6 portrays the confusion matrix for method 2. Table III has all the values of the 3 metrics for tall the labels.

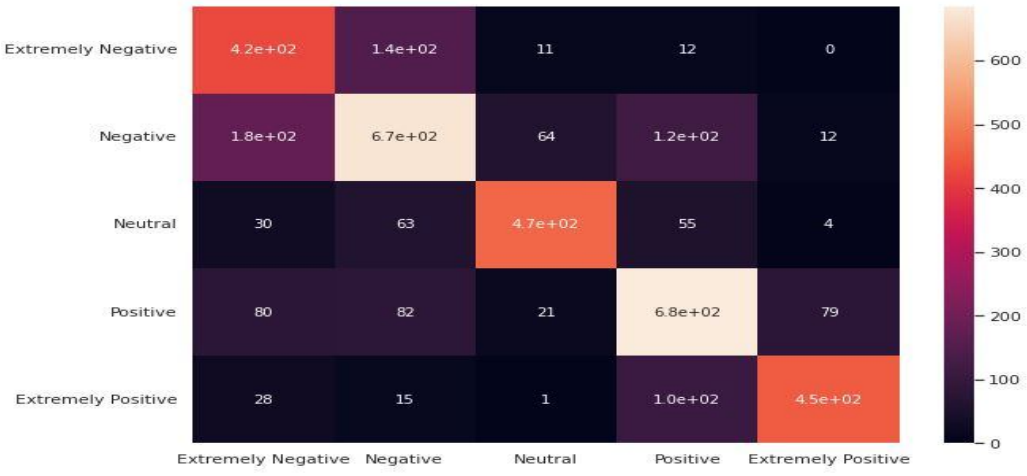


Fig. 6 Confusion matrix for Method 2(LSTM Based)

Table III
Evaluation Metrics for Method 2

	Precision	Recall	F1 score
Extremely Negative	0.76	0.69	0.72
Negative	0.69	0.65	0.67
Neutral	0.83	0.75	0.79
Positive	0.70	0.72	0.71
Extremely Positive	0.83	0.75	0.79

C. Method 3

This method made use of the language model BERT. The preliminary results were very exemplary with an accuracy of 85.75%. The labels 'Extremely Negative' and 'Extremely fared the best in terms of F1 score with a value of 0.88. The label 'Neutral' had the best recall of 0.88 while the label 'Extremely Positive' had the best precision of 0.92. All the labels had comparable precision, recall and F1 scores, with the maximum difference being 0.1 between the maximum and minimum value for precision, i.e. between 'Neutral' and 'Extremely Positive' for precision.

This method that made use of the language model BERT the preliminary results was very exemplary with a validation accuracy of 85.75%. There wasn't a need to encode the labels into numerical values. The labels 'Extremely Negative' and 'Extremely Positive' fared the best in terms of F1 score with a value of 0.88. The label 'Neutral' had the best recall of 0.88 while the label 'Extremely Positive' had the best precision of 0.92. All the labels had comparable precision, recall and F1 scores, with the maximum difference being 0.1 between the maximum and minimum value for precision, i.e. between 'Neutral' and 'Extremely Positive' for precision. Fig. 7 demonstrates the confusion matrix for method 3, while table IV captures the values of F1 score, precision and recall for method 3, for all the labels.

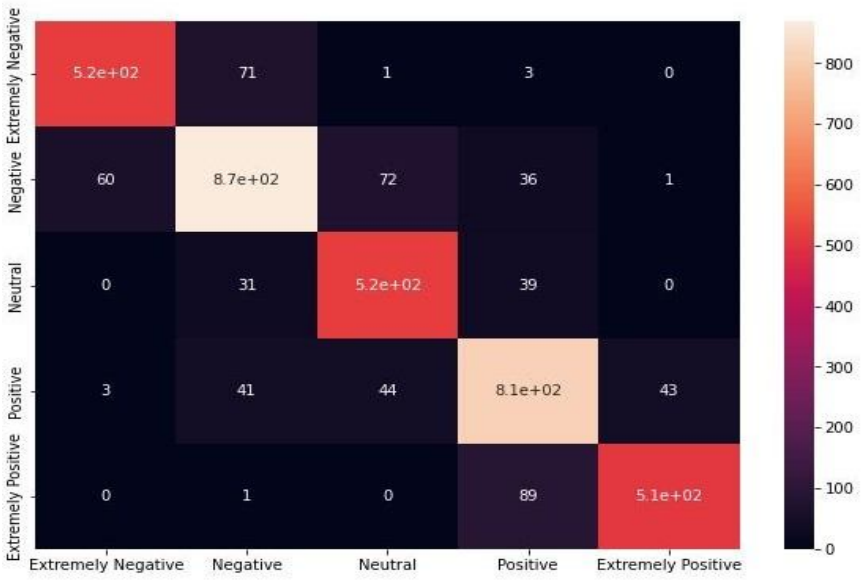


Fig. 7 Confusion matrix for Method 3(BERT Based)

TABLE IV
Evaluation Metrics For Method 3

	Precision	Recall	F1 score
Extremely Negative	0.89	0.87	0.88
Negative	0.86	0.84	0.85
Neutral	0.82	0.88	0.85
Positive	0.83	0.86	0.84
Extremely Positive	0.92	0.85	0.88

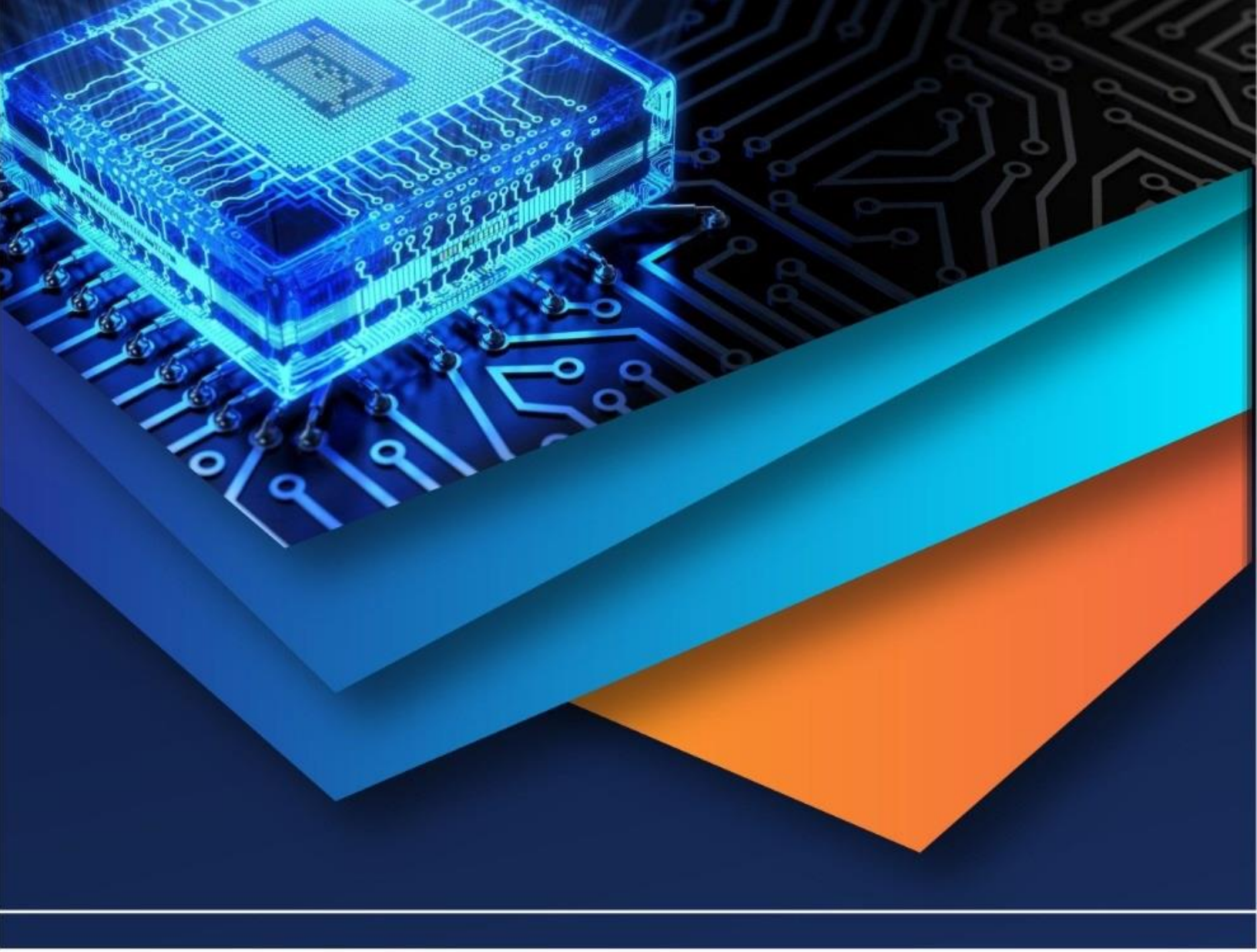
VI. CONCLUSION

In this paper different methods to analyse sentiments in a tweet were explored. The sentiment classification was a multiclass classification problem with 5 labels (classes). It was observed that for NLP applications pre-processing of the text, sequences plays a key role in the performance of the models that follow the pre-processing steps and also have a significant impact on the performance metrics. For this paper different pre-processing steps were implemented, ranging from cleaning of texts of hash tags, mentions, html tags etc. to tokenization of each word in a sentence to generating the word embeddings for the LSTM and BERT models. Each method required its own specific pre-processing step and its own technique of classification. BERT based model i.e. method 3 gave the best results in terms of accuracy, precision, recall and F1 scores. BERT continues to be one of the most important language models currently across many applications. For method 1 that used count vectorizer to extract features from text and MLP classifier for classification, the result was moderately good with an accuracy of 64%. LSTM method performed better than the method 1 with an accuracy of 73% on the test set. The precision, recall and F1 score values of the labels varied over a relatively sizeable range.

Twitter data is highly unstructured and there is a usage 5of lot of slangs, shorthand forms and insider references. In future work we aim to continuously advance our pre-processing techniques to accommodate these nuances and informal conventions, connotations that drift away from structured, formal texts. For emoticons, emoticon-based analysis [13], according to the research on Automatic Sentiment Analysis of Twitter message, shall be used. More number of classifiers paired with frequency based feature extraction methods will be implemented. We also intend to find ways to improve BERT and LSTM based models by exploring different model architectures.

REFERENCES

[1] A. Go, L. Huang, and R. Bhayani, R, “Twitter sentiment analysis”. Entropy 17, 2009.
[2] S. W. Davenport, S. M. Bergman, J. Z. Bergman, J. Z., and M. E.Fearrington, “Twitter versus Facebook: Exploring the role of narcissism in the motives and usage of different social media platforms”, Computers in Human Behavior 32, pp. 212-220, 2014.
[3] Nazzal, Jamal & El-Emary, Ibrahim & Najim, Salam. (2008). Multilayer Perceptron Neural Network (MLPs) For Analyzing the Properties of Jordan Oil Shale. World Applied Sciences Journal. 5.
[4] A. Westerski, “Sentiment Analysis: Introduction and the State of the Art overview”, Universidad Politecnicade Madrid, Spain, pp 211-218, 2007.
[5] T. Nasukawa, and J. Yi, “Sentiment analysis: Capturing favorability using natural language processing”, in Proceedings of the 2nd international conference on Knowledge capture, pp. 70-77, ACM, October 2003.
[6] A. Go, R. Bhayani, and L. Huang, L., “Twitter sentiment classification using distant supervision”, CS224N Project Report, Stanford, 1-12, 2009.
[7] Jitendra Kumar, Rimsha Goomer, Ashutosh Kumar Singh, Long Short Term Memory Recurrent Neural Network (LSTM-RNN) Based Workload Forecasting Model For Cloud Datacenters, Procedia Computer Science, Volume 125, pp. 676-682, 2018.
[8] Hochreiter, Sepp & Schmidhuber, Jürgen. (1997). Long Short-term Memory. Neural computation. 9. 1735-80. 10.1162/neco.1997.9.8.1735.
[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
[10] Yongjun Hu, Jia Ding, Zixin Dou, Huiyou Chang, "Short-Text Classification Detector: A Bert-Based Mental Approach", Computational Intelligence and Neuroscience, vol. 2022, Article ID 8660828, 11 pages, 2022. https://doi.org/10.1155/2022/8660828.
[11] N. Blenn, K. Charalampidou, and C. Doerr, “Context-Sensitive sentiment classification of short colloquial text”, in NETWORKING 2012, pp. 97-108, Springer Berlin Heidelberg, 2012.
[12] R. Kohavi, and F. Provost, “Glossary of terms” Machine Learning 30(2-3), pp 271-274, 1998.
[13] A. C. Lima, and L. N. de Castro, “Automatic sentiment analysis of Twitter messages”, in Computational Aspects of Social Networks (CASoN), 2012 Fourth International Conference, pp. 52-57, IEEE, 2012.
[14] Sebastiani F. “Machine learning in automated text categorization,” ACM computing surveys (CSUR). vol. 34, no. 1, 2002, pp. 1-47.
[15] Yang B, Cardie C. “Context-aware learning for sentence-level sentiment analysis with posterior regularization” Proc. on Annual Meeting of the Association for Computational Linguistics,2014, pp. 325-335.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY