iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 9 **Issue:** VII **Month of publication:** July 2021

DOI: https://doi.org/10.22214/ijraset.2021.36309

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: 7.429
Volume 9 Issue VII July 2021- Available at www.ijraset.com

Twitter Sentiment Analysis Model

Pushkar Dubey[1], Zulfikar Ali Ansari[2], Priyanka Verma[3], Ritu Verma[4], Shivani Avasthi[5], Km Karina Singh[6]
[1, 2, 3, 4, 5]*Computer Science and Engineering, Babu Banarasi Das Institute of Technology and Management Lucknow*

Abstract: Social networks are the main resources to gather information about people's opinion towards different topics as they spend hours daily on social media and share their opinion. Twitter is one of the social media that is gaining popularity. Twitter offers organizations a fast and effective way to analyze customers' perspectives toward the critical to success in the market place. Developing a program for sentiment analysis is an approach to be used to computationally measure customers' perceptions. .We use natural language processing and machine learning concepts to create a model for analysis . In this paper we are discussing how we can create a model for analysis of twittes which is trained by various nlp , machine learning and Deep learning Approach.

Keyword: Machine Learning, Anaconda, python, positive, negative, Social media,Natural Language Processing.

I. INTRODUCTION

What do we do when we want to express ourselves or reach out to a large audience? We log on to one of our favourite social media sites. Social Media has taken over in today's world and Twitter is one of the biggest platform where we express our views, emotions, opinion about a specific point. However, people write anything such as social activities or any comment on products. Twitter serves as a mean for individuals to express their thoughts or feelings about different subjects.

[1] These emotions are used in various analytics for better understanding of humans. . In contrast, consumers have all the power when it comes to what consumers want to see and how consumers respond. With this, the company's success & failure is publicly shared and end up with word of mouth. However, the social network can change the behaviour and decision making of consumers

[2] In this paper, we have attempted to create and trained a model for analysis on "tweets" using machine learning and natural language processing . We attempt to classify the polarity of the tweet where it is either positive or negative. If the tweet has both positive and negative elements, the more dominant sentiment should be picked as the final label.

A. Natural Language Processing

NLP is a branch of Artificial Intelligence which deal with bridging the machines understanding humans in their Natural Language. Natural Language can be in form of text or sound, which are used for humans to communicate each other. NLP can enable humans to communicate to machines in a natural way.Modern NLP algorithms are based on machine learning, especially statistical machine learning. The paradigm of machine learning different from that of most prior attempts at language processing. Prior implementations of language-processing tasks typically involved the direct hand coding of large sets of rules. The machine-learning paradigm calls instead for using general learning algorithms — often, although not always, grounded in statistical inference — to automatically learn such rules through the analysis of large corpora of typical real-world examples. A corpus (plural, "corpora") is a set of documents (or sometimes, individual sentences) that have been hand-annotated with the correct values to be learned. Many different classes of machine learning algorithms have been applied to NLP tasks.

II. LITERATURE SURVEY

1) *Analysis for Social Media [3]:* Analysis is a problem of text based analysis, but there are some challenges that make it difficult as compared to traditional text based analysis This clearly states that there is need of an attempt to work towards these problems and it has opened up several opportunities for future research for handling negations, hidden sentiments identification, slangs , polysemy.
2) *Analysis Prototype System for Social Network Data[4]:* This paper discusses a prototype system to enable automated opinion based analysis of social network data for evaluation of the public social conscience of user provided topics and events.
3) *Analysis of Tweets Using Machine Learning Approach [5]:* The research of sentiment analysis of Twitter data can be performed in different aspects. This paper shows analysis types and techniques used to perform extraction of sentiment from tweets. In this survey paper, we trained a model using machine learning and nlp concepts.

4) *Opinion Mining on Social Media Data[6]:* Po-Wei Liang et.al used Twitter API to collect data from twitter. Tweets which contain opinions were filtered out. Naive Bayes model was developed for polarity identification. They also workessd for elimination of unwanted features by using the Mutual Information and Chi square feature extraction method. Finally, the approach for predicting the tweets as positive or negative did not give better accuracy by this method.
5) *Analysis for Using Twitter Data Set[7]:* The research of sentiment analysis of Twitter data can be performed in different aspects. This paper shows how we cann create a model and trained it for sentiment analysis t used to perform extraction of sentiment from tweets.

III. PROPOSED METHODOLOGY

The proposed model of twitter data analysis will be implemented using Anaconda python. Anaconda is a free and open source distribution of the Python and R programming languages for data science and machine learning related applications. It can be installed on Windows, Linux, and Mac OS. Anaconda is an open source, cross-platform, package management system. The tweets can be analyzed and characterized based on the emotions used by the social users. We attempt to classify the polarity of the tweet where it is either positive or negative. If the tweet has both positive and negative elements, the more dominant sentiment should be picked as the final label. We use the dataset from twitter which was labelled positive/negative. The data provided comes with emoticons, usernames and hash-tags which are required to be processed and converted into a standard form. Finally, we report our experimental results and findings at the end.

A. Data Description

We using three data set for train our model first one is spam sms data set data set and second is disasters tweets data set by which we create functions for data prepossessing and at the final step we train our model on twitter data set having 1600000 dataset. The data given from the dataset is in the form of comma separated values. The training dataset is a CSV (comma separated value) file of type tweet-id, sentiment, tweet where the tweet-id is a unique integer identifying the tweet, sentiment is either 1 (positive) or 0 (negative), and tweet is the tweet enclosed in "". Similarly, the test dataset is a CSV file of type tweet-id, tweet respectively. The dataset is a mixture of words, emoticons, symbols, URLs and references to people as seen usually on twitter. Words and emoticons contribute to predicting the sentiment, but URLs and references to people don't. Therefore, URLs and references are being ignored. The words are also a mixture of misspelled words / incorrect, extra punctuations, and words with many repeated letters. The "tweets", therefore, must be pre-processed to standardize the dataset. The provided training and test dataset have 800000 and 200000 tweets respectively. Preliminary statistical analysis of the contents of datasets, after pre- processing .

B. Pre-Processing

Raw tweets scraped from twitter generally result in a noisy and obscure dataset. This is due to the casual and ingenious nature of people's usage of social media. Tweets have certain special characteristics such as retweets, emoticons, user mentions, etc. which should be suitably extracted. Therefore, raw twitter data must be normalized to create a dataset which can be easily learned by various classifiers. We have applied an extensive number of pre-processing steps to standardize the dataset and reduce its size. We first do some general pre- processing on tweets which is as follows:

1) *Cleaning the Corpus:* This data needs to be cleaned before analysing it or fitting a model to it. Cleaning up the text data is necessary to highlight the attributes that you're going to want your machine learning system to pick up on. Cleaning (or pre-processing) the data typically consists of a number of steps.

	Target	message	message_len	message_clean
0	ham	Go until Jurong point, crazy..Available only…	20	Go until jurong pointcrazy available only
1	ham	Ok lar…Joking wif u oni…	6	Ok lar joking wif u oni
2	spam	Free entry in 2 a wkly comp to win FA Cup fina…	28	Free entry in a wkly comp to win fa cup final

2) *Stopwords:* Stopwords include: I, he, she, and, but, was were, being, have, etc, which do not add meaning to the data. So these words must be removed which helps to reduce the features from our data. These are removed after tokenizing the text.

	target	message	message_len	message_clean
0	ham	Go until Jurong point, crazy..Available only…	20	Go jurong point crazy available bugis n great…
1	ham	Ok lar…Joking wif u oni…	6	Ok lar joking wif u oni
2	spam	Free entry in 2 a wkly comp to win FA Cup fina…	28	Free entry wkly comp win fa cup finaltkts m..

3) *Stemming/ Lemmatization:* For grammatical reasons, documents are going to use different forms of a word, such as *write, writing and writes*. Additionally, there are families of derivationally related words with similar meanings. The goal of both stemming and lemmatization is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form. **Stemming** usually refers to a process that chops off the ends of words in the hope of achieving goal correctly most of the time and often includes the removal of derivational affixes. **Lemmatization** usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base and dictionary form of a word

As far as the meaning of the words is not important for this study, we will focus on stemming rather than lemmatization.

a) Stemming Algorithms

There are several stemming algorithms implemented in NLTK Python library:

- PorterStemmer uses *Suffix Stripping* to produce stems. PorterStemmer is known for its simplicity and speed. Notice how the PorterStemmer is giving the root (stem) of the word "cats" by simply removing the 's' after cat. This is a suffix added to cat to make it plural. But if you look at 'trouble', 'troubling' and 'troubled' they are stemmed to 'trouble' because PorterStemmer algorithm does not follow linguistics rather a set of 05 rules for different cases that are applied in phases (step by step) to generate stems. This is the reason why PorterStemmer does not often generate stems that are actual English words. It does not keep a lookup table for actual stems of the word but applies algorithmic rules to generate stems. It uses the rules to decide whether it is wise to strip a suffix.
- One can generate its own set of rules for any language that is why Python NLTK introduced SnowballStemmers that are used to create non-English Stemmers!
- LancasterStemmer (Paice-Husk stemmer) is an iterative algorithm with rules saved externally. One table containing about 120 rules indexed by the last letter of a suffix. On each iteration, it tries to find an applicable rule by the last character of the word. Each rule specifies either a deletion or replacement of an ending. If there is no such rule, it terminates. It also terminates if a word starts with a vowel and there are only two letters left or if a word starts with a consonant and there are only three characters left. Otherwise, the rule is applied, and the process repeats.

	target	message	message_len	message_clean
0	ham	Go until Jurong point, crazy..Available only…	20	go jurong point crazi avail bugi n great world...
1	ham	Ok lar…Joking wif u oni…	6	ok lar joke wifoni
2	spam	Free entry in 2 a wkly comp to win FA Cup fina…	28	free entriwkli comp win fa cup final tkts m...

4) *Target encoding:* Target encoding is the process of replacing a categorical value with the mean of the target variable. Any non-categorical columns are automatically dropped by the target encoder model. ... This can help improve machine learning accuracy since algorithms tend to have a hard time dealing with high cardinality columns.

C. Vectorization

We have the messages as lists of tokens (also known as lemmas) and now we need to convert each of those messages into a vector the SciKit Learn's algorithm models can work with.

We'll do that in three steps using the bag-of-words model:

1) Count how many times does a word occur in each message (Known as term frequency)
2) Weigh the counts, so that frequent tokens get lower weight (inverse document frequency)
3) Normalize the vectors to unit length, to abstract from the original text length (L2 norm)

Let's begin the first step:

Each vector will have as many dimensions as there are unique words in the SMS corpus. We will first use SciKit Learn's CountVectorizer**.** This model will convert a collection of text documents to a matrix of token counts.

We can imagine this as a 2-Dimensional matrix. Where the 1-dimension is the entire vocabulary (1 row per word) and the other dimension are the actual documents, in this case a column per text message.

D. Tunning CountVectorizer

CountVectorizer has a few parameters you should know.

1) *stop_words:* Since CountVectorizer just counts the occurrences of each word in its vocabulary, extremely common words like ‘the’, ‘and’, etc. will become very important features while they add little meaning to the text. Your model can often be improved if you don’t take those words into account. Stop words are just a list of words you don’t want to use as features. You can set the parameter stop_words=’english’ to use a built-in list. Alternatively you can set stop_words equal to some custom list.
2) *ngram_range:* An n-gram is just a string of n words in a row. E.g. the sentence ‘I am Groot’ contains the 2-grams ‘I am’ and ‘am Groot’. The sentence is itself a 3-gram. Set the parameter ngram_range=(a,b) where a is the minimum and b is the maximum size of ngrams you want to include in your features. The default ngram_range is (1,1). In a recent project where I modeled job postings online, I found that including 2-grams as features boosted my model’s predictive power significantly.

E. TF-IDF

In information retrieval, tf–idf, TF-IDF, or TFIDF, short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus It is often used as a weighting factor in searches of information retrieval, text mining, and user modeling. The tf–idf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general.

tf–idf is one of the most popular term-weighting schemes today. A survey conducted in 2015 showed that 83% of text-based recommender systems in digital libraries use tf–idf.

F. Word Embeddings: GloVe

To obtain a vector representation for words we can use an unsupervised learning algorithm called GloVe (Global Vectors for Word Representation**)**, which focuses on words co-occurrences over the whole corpus. Its embeddings relate to the probabilities that two words appear together.

Word embeddings are basically a form of word representation that bridges the human understanding of language to that of a machine. They have learned representations of text in an n-dimensional space where words that have the same meaning have a similar representation. Meaning that two similar words are represented by almost similar vectors that are very closely placed in a vector space.Thus when using word embeddings, all individual words are represented as real-valued vectors in a predefined vector space. Each word is mapped to one vector and the vector values are learned in a way that resembles a neural network.

Pad_sequences- This function transforms a list (of length num_samples) of sequences (lists of integers) into a 2D Numpy array of shape (num_samples, num_timesteps). num_timesteps is either the maxlen argument if provided, or the length of the longest sequence in the list.

```
>>> sequence = [[1], [2, 3], [4, 5, 6]]
>>>tf.keras.preprocessing.sequence.pad_sequences(sequence, padding='post')
array([[1, 0, 0],
       [2, 3, 0],
       [4, 5, 6]], dtype=int32)
```

G. Modeling

We will be using a model consisting of both test and training dataset model using various algorithms as due to the modular nature of the program we can add and remove the algorithms with ease. Let's understand the workflow of our system with the help of above diagram. First, we have split the data into training and test set. We also keep separate positive and negative pre-labelled datasets for training the model and checking their generalization classification in test set. After this the training data is fed to some machine learning algorithm like Naive Bayes, Maximum Entropy,SVM[9] which learns to make predictions. To evaluate our system, we use Baseline Classification which is our evaluation metric in which test data is fed to the learned algorithm which in return generates recommended prediction ratings of words. With help of pre-classified golden set and evaluation metric we check the accuracy of our model.

1) *Naïve Bayes:* Naive Bayes classifiers are a family of simple "probabilistic classifiers "based on applying Bayes' theorem with strong (naive) independence assumptions between the features." Naive Bayes is a simple model which can be used for text classification. In this model, the class *c* is assigned to a tweet t, where

$$\hat{class} = argmax_{class}\, P(Class = class \mid Feature\,set = feature\,set)$$

$$P(class|feature\,set) = \frac{P(feature\,set|class) \times P(class)}{P(feature\,set)}$$

In the formula above, fi represents the *i*th feature of total features. Pct and $Pfic$ can beobtained through maximum likelihood estim ates. We used Multinomial NB from scikit-learn naïve bayes package of scikit-learn for Naive Bayes classification. We used Laplace smoothed version of Naive Bayes with the smoothing parameter α set to its default value of 1. We used sparse vector representation for classification and ran experiments using both presence and frequency feature types.

2) *XGBoost:* XGBoost is an algorithm that has recently been dominating applied machine learning and Kaggle competitions for structured or tabular data. XGBoost is an implementation of gradient boosted decision trees designed for speed and performance. XGBoost dominates structured or tabular datasets on classification and regression predictive modelling problems. The evidence is that it is the go-to algorithm for competition winners on the Kaggle competitive data science platform.
3) *LSTM:* LSTM networks are well-suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. LSTMs were developed to deal with the vanishing gradient problem that can be encountered when training traditional RNNs. Relative insensitivity to gap length is an advantage of LSTM over RNNs, hidden Markov models and other sequence learning methods in numerous applications.

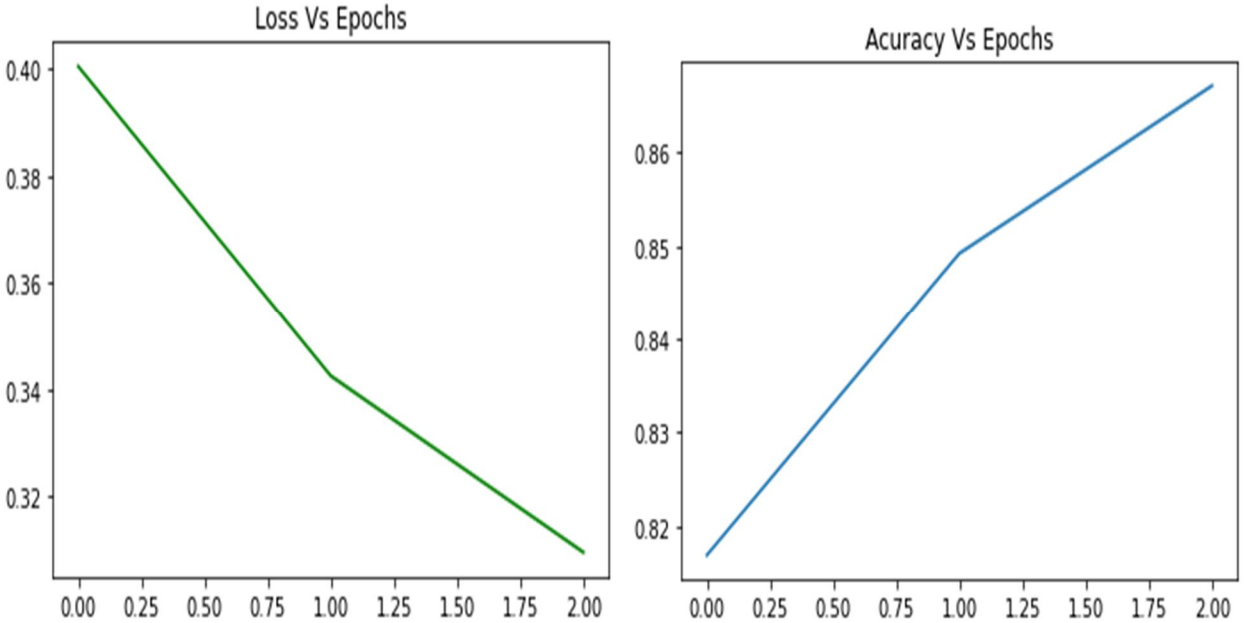
In this project LSTM make small modifications to the information by multiplications and additions. With LSTMs, the information flows through a mechanism known as cell states. This way, LSTMs can selectively remember or forget things. The information at a particular cell state has three different dependencies.

Top words for Fake messages

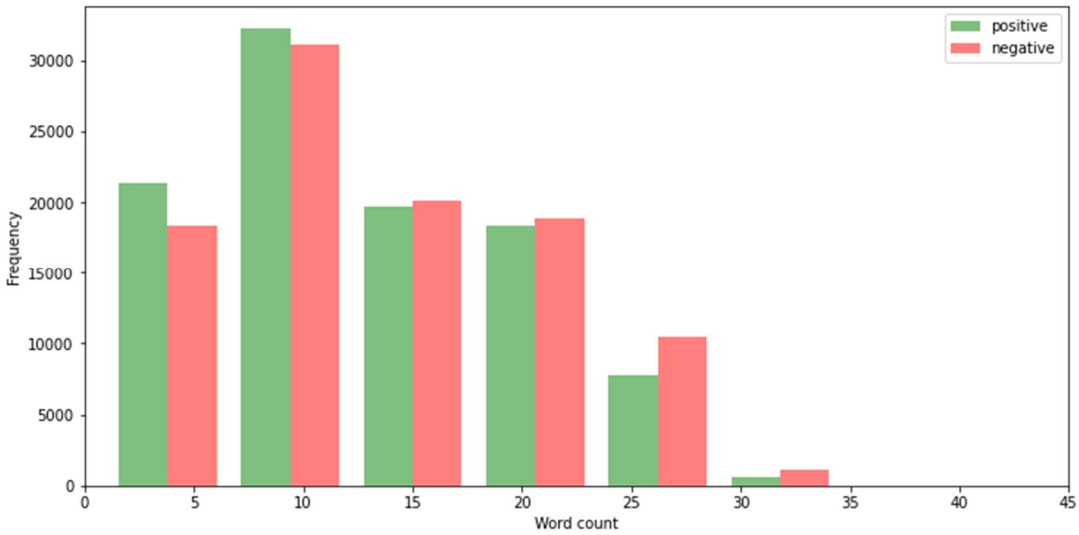
Top words for Real Disaster tweets

IV. RESULT

A. *This Graph Shows losses and Acuuracy of our Model*



B. Positive And Negative Polarity Of Twitts Based On Tweets Length



V. CONCLUSION

In this technical paper, we discussed the methods of creating model for tweets analysis . We showed the result of accuuracy of our model .. The model we've built can easy be scaled for new algorithms be it in Machine Learning, Deep learning or Natural Language Processing. Sentiment analysis system is an active field of research and we can still further improve our system by working more on the algorithms, trying out different things in pre-processing and checking which ones get the best precision metrics.

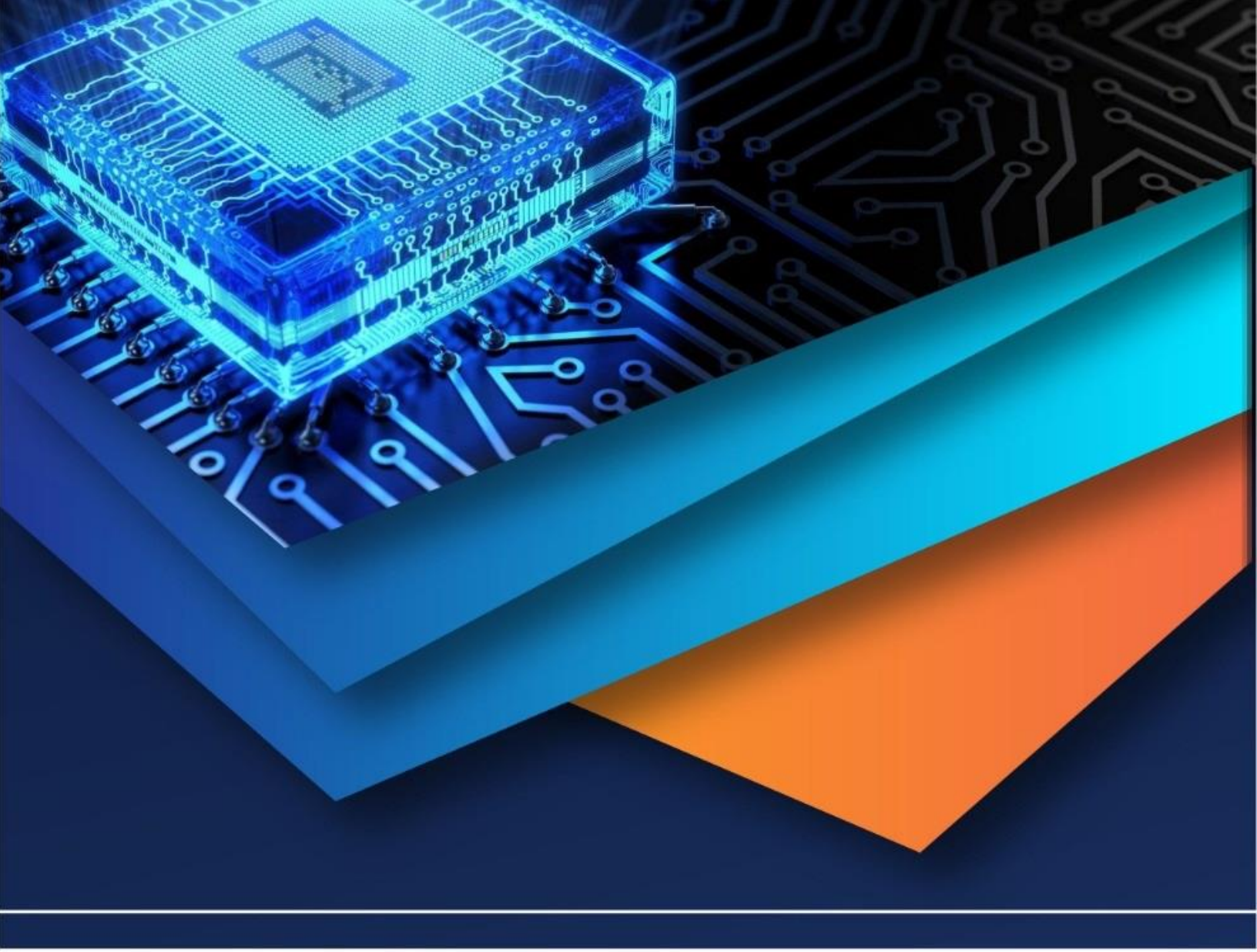
VI. FUTURE WORK

We have presented an effective and robust model for sentiment detection for Twitter messages which uses biased and noisy labels as input tobuild its models. This performance is due to thefact that: (1) our approach creates a more abstract representation of these messages, instead of usinga raw word representation of them as some previous approaches; and (2) although noisy and biased, the data sources provide labels of reasonablequality and, since they have different bias, combining them also brought some benefits.

A. Use of Emoticons
B. Handling Use of Symbol in each and every data set of tweets
C. Model is slow we can make it fast
D. We can remove system Dependency , this model can not run fast if system have less memory.
E. Finding Related hashtags positive and negative
F. Forecast for trends based on above details

REFERENCES

[1] Younis, Eman MG. (2015) "Sentiment analysis and text mining for social media microblogs using open source tools: an empirical study." International Journal of Computer Applications, 112(5) :0975 – 8887
[2] Garg, Neha, and Rinkle Rani (2017) "Analysis and visualization of Twitter data using k-means clustering." International Conference on Intelligent Computing and Control Systems (ICICCS), IEEE, 670-675.
[3] A.Brahmananda Reddy, D.N.Vasundhara, P. Subhash "Sentiment Research on Twitter Data" Sept(2019) International Journal of Recent Technology & Engineering (IJRTE) ISSN: 2277- 3878
[4] Devika, R., and S. Revathy. (2018) "Survey on clustering techniques in Twitter data." Second International Conference on Computing Methodologies and Communication (ICCMC). IEEE, 1073-1077.
[5] Ankita Sharma , Udayan Ghose "sentimental Analysis of twitter Data with respect to General Elections in India" International Conference on smart Sustainable Intelligent Computing and application under (ICITETM2020)325-334
[6] Ankit Pradeep Patel, Ankit Vithalbhai Patel, Sanjay Kumar Ghanshyam bhai Butani, Prashant B. Sawant "Literature Survey on Sentiment Analysis of twitter Data using Machine Learning Approaches" 2017 International Journal Innovative Research in Science & Technology (IJIRST). ISSN, 2349-6010



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY