Comparative Evaluation on Sentiment Analysis Algorithms

Aman Kumar(✉), Manish Khare, and Saurabh Tiwari

Dhirubhai Ambani Institute of Information and Communication Technology (DA-IICT), Gandhinagar, Gujarat, India
{201911018,manish_khare,saurabh_t}@daiict.ac.in

Abstract. Classifying texts based on sentiments present in the text is called Sentiment Analysis. There are many Sentiment Analysis Techniques available. In this paper, we have addressed the problem of sentiment analysis and compared many different machine learning and deep learning algorithms to perform sentiment analysis based on their accuracy. We extracted useful features to feed them in our classifier to generate results. Also, we have used the majority vote ensemble method to achieve more accurate results.

Keywords: Sentiment analysis · Natural language processing · Machine learning · Deep learning · Comparison

1 Introduction

Sentiment Analysis is one of the functional problems used for quantifying the sentiments and emotions from neutral texts. It refers to obtaining sentiment data from the source, processing it using an algorithm, and generating positive, negative, and neutral sentiments [1]. Sentiment Analysis helps industries to analyze thousands of product reviews in mere seconds. Sentiment Analysis involves four steps: Step 1 is Gathering data, which involves gathering data to analyze and use, for Example, GitHub comment logs. Step 2 is Preparing data for processing, and here the data undergoes a few steps to make it for processing. These steps involve Tokenisation, Lower Casing, Stemming, Removing punctuation, and stop words. Step 3 is the significant step for Sentiment Analysis which is Applying Classifier. We select a classifier that will classify the text into sentiments. Step 4 is the Visualisation of results. In this step, we visualize the results to understand the outcome of the analyzed data.

This paper has compared many machine learning algorithms used for Sentiment Analysis. These algorithms are NLTK, Decision Trees, Random Forest, Support Vector Machine (its kernels, i.e., Linear Kernel, Polynomial Kernel, Gaussian Radial Basis (RBF) Kernel, Sigmoid Kernel), Convolutional Neural Network (CNN), Long Short Term Memory (LSTM) and Ensemble Techniques.

The organization of the paper is as follows. Section 2 contains the discussion about sentiment analysis and the various literature survey related to sentiment analysis.

A. D. Mambo et al. (Eds.): InterSol 2022, LNICST 449, pp. 119–131, 2022.
https://doi.org/10.1007/978-3-031-23116-2_9

Section 3 presents the methodology, dataset creation, pre-processing, and tool selection. Section 4 presented the result analysis and discussion. Section 5 presents the conclusions and future work of the study.

2 Sentiment Analysis

Sentiment Analysis is the management of sentiments, subjective texts, and opinions. The sentiments can be categorized into negative, positive, and neutral. Sentiment analysis is used to determine the emotion of the user automatically. Sentiment analysis is an important research area due to the massive number of daily posts on social media; extracting people's opinions is challenging. Sentiment Analysis allows us to sort large data sets and automatically detect each text's polarity, which saves time and resources [2]. Some Examples of Sentiment Analysis are given in Table 1.

Table 1. Examples of sentiment analysis

Text	Sentiments
I am very happy today	Positive
I have to complete this	Neutral
The weather is terrible	Negative
Dataset is a mixture of words	Neutral

2.1 Advantage of Sentiment Analysis

The advantage of Sentiment Analysis is that it can analyze vast amounts of data quickly, which can be a hassle to do manually. Real-Time Analysis: Industries use it to monitor real-time data and make changes or improvements wherever needed. Consistent Criteria: Analysing sentiment is a subjective task that can be perceived differently when done by two-member and results will probably be biased.

2.2 Related Work

Several authors have conducted several studies on sentiment analysis. An essential aspect in approaching Sentiment analysis is presented in Feldman [3]. The author discussed the techniques to perform sentiment analysis and the various sentiment analysis applications that help us solve real-world problems. The author also discussed how sentiment analysis techniques could solve complex problems that some industries face and simplify their sentiment analysis problems.

Beigi et al. [4] discussed the application of sentiment analysis on disaster relief and the overview of sentiment analysis in social media. The author talks about how sentiment analysis is not limited to politics, business intelligence, and other issues. The author studied the reaction of the local crowd and used such information to improve disaster management. Dolianiti et al. [5] presented the applications of sentiment analysis in education. The author discussed how recognition and emotions are involved in every

learning process and how student profiles can enhance information regarding the effective state. The author explored many different ways in which sentiment analysis can apply in the educational domain.

Das et al. [6] discussed real-time stock prediction by analysing the sentiment of Twitter streaming data in their study. The author attempts to make decisions related to finance, such as stock market prediction, to predict a company's stock prices. To perform this task, Twitter streaming data has been considered for scoring the impression carried for a particular firm.

3 Methodology

This study performs a comparative study on different sentiment analysis algorithms. The overall methodology adopted for sentiment analysis is divided into three different parts. First, the data is collected from the dataset used. Second, the sentiment analysis is conducted on the artefacts. Finally, the tool is selected for performing sentiment analysis.

3.1 Dataset

We have used sentiment140[1] dataset. Graduate students of Stanford University create Sentiment140. It contains 1,600,000 pre-classified tweets extracted using Twitter API. The tweets have predefined sentiments (0 = negative, 4 = positive) that have been used for sentiment analysis. It contains six fields target and that is: it is the predefined polarity of the tweet, ids: it is an id assigned to a tweet, date: the date on which tweet is written, flag: query of the tweet, user: the user who has written the tweet, text: Textual content of the tweet. Many other datasets can be used for Sentiment Analysis, but this is the most common dataset.

3.2 Sentiment Analysis

Sentiment analysis contains six different steps: Tokenisation, Lower casing, removing punctuation, removing stop words, stemming, and word embedding. These steps are explained in Fig. 1, and details about these steps are given below.

1) **Tokenisation.**

Tokenisation is converting a piece of text into tokens. It is the task of converting sentiments into words, called tokens. Tokens are the building blocks of the Natural Language. The common ways of processing raw text happen at the token level. We have used functions of the list to tokenise the text [7].

2) **Lower Casing.**

The lower casing is converting the tokens into lower case letters. We converted all texts into lowercase so that the processing algorithm does not recognise capital letters and lowercase letters separately. We have used the lower method of string in python for lower casing the text [8].

[1] http://help.sentiment140.com.

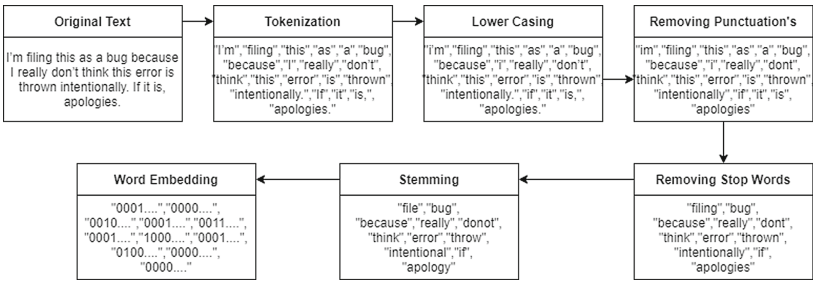

Fig. 1. Steps of data pre-processing for sentiment analysis

3) **Removing punctuation.**

It helps us optimise text so that the processing algorithm works more efficiently because punctuation does not add more value to sentiment data. We have removed punctuation from the text using a punctuation variable from the string in python [8].

4) **Removing Stop Words.**

Stop words are words that do not contribute to sentiment analysis. These stop words are a set of commonly used words that carry very little information. Stop words such as: I, am, the, etc. We have used nltk.corpus[2] library to remove stop words from text [8].

5) **Stemming.**

It is a process of gaining the root word of a word by removing affixes and suffixes. Some examples of stemming are like for word 'confirmed' the stemmed word is 'confirm'. We have used WordNetLemmatizer from nltk.stem[3] library to perform word-stemming [8].

6) **Word Embedding.**

It represents words in the form of a vector that encodes the value of the word. Word2Vec [9] for Word Embedding allows us to represent each word in unique vectors and represent the relationship between them. We have used Word Embedding for Machine Learning Algorithms such as SVM, CNN and LSTM.

3.3 Tool Selection

After pre-processing data and making it fit our sentiment analysis algorithm, we use different classifiers and algorithms to classify our text into sentiments [10–12]. We used different Sentiment Analysis Algorithms.

1) **Natural Language Toolkit (NLTK).**

The central concept behind NLP is that we must analyse the dataset by accepting the dataset and reading the data. We read the data and compare it with the dictionary of

[2] https://www.nltk.org/api/nltk.corpus.html.
[3] https://www.nltk.org/api/nltk.stem.html.

words to check whether the word is already defined in the dictionary or not. NLP has two main approaches [13].

- Rule-Based Approach—We use a supervised learning algorithm based on a dictionary of words and rules.
- Machine Learning-Based Approach—We use unsupervised data. In this, we use training data as trained data to generate our model.

We are using a Rule-based approach for sentiment analysis. We are using Sentiment Intensity Analyser of nltk.sentiment.vader[4] library. The sentiment IntensityAnalyzer assigns a sentiment score to each token which determines the polarity of the word. Then we take the overall sum of each tweet; if the sum is positive, it is assigned as a positive tweet, but if the sum is zero, then the tweet is assigned as a neutral tweet, or else it is assigned as a negative tweet. The process of sentiment analysis is given in Fig. 2.

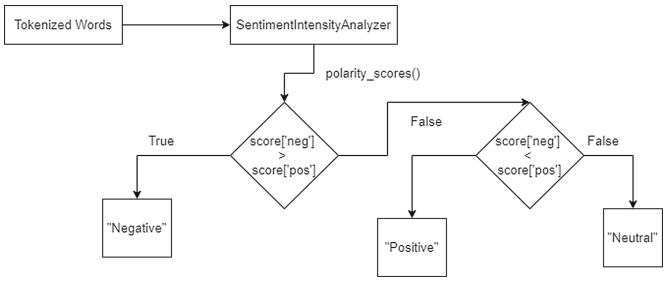

Fig. 2. Process of Natural Language ToolKit (NLTK)

2) **Decision Tree.**

It is a classifier model in a tree where each node represents a feature, and every child node represents an outcome. It uses labels to form a decision tree. The data division is done until the leaf node contains specific minimum numbers of recursion records used for classification [14]. We use categorical variable decision trees as sentiment is divided into three classes positive, neutral and negative. First features are extracted from the text, which acts as nodes in a decision tree. Then the data is fed to a DecisionTreeClassifier from sklearn[5] library, which gives us the polarity of the text. The process is given in Fig. 3.

3) **Random Forest.**

[4] https://www.nltk.org/api/nltk.sentiment.html.

[5] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html.

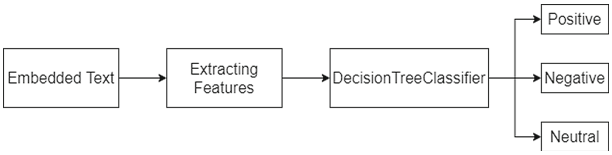

Fig. 3. Process of decision tree

It is an ensemble technique that can use for classification problems. It is a Bagging Problem. It includes Bootstrapping and Aggregation. Bootstrapping is selecting random samples and creating Decision Trees.

Aggregation is the voting of the classifier by checking each Decision Tree and noting the maximum frequency result [14]. We are creating many different decision trees and applying a voting classifier to the predictions of all the trees. The process of applying random forest is given in Fig. 4.

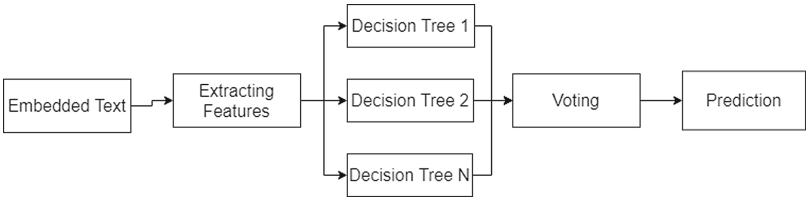

Fig. 4. Process of random forest

4) **Support Vector Machine.**

SVM is a linear classifier that gives non-probabilistic binary values. It is sparse, so it is ideally suited for text classification. In SVM, for a training set of points, we want to find a maximum-margin hyper-plane. The equation of hyper-plane is

$$w^T.x + b = 0 \tag{1}$$

where $x, w \in R^n, b \in R$

we need to maximise the margin [14]

$$\min_{w,b,\xi_i} \frac{1}{2} w^T w + \sum_{i=1}^{l} \xi_i \tag{2}$$

Subject to,

$$y_i(w^T x_i + b) \geq 1 = \xi_i \tag{3}$$

$$\xi_i \geq 0, i = 1, 2, 3, ..., l$$

Here l is the number of training point

Our approach has experimented with four different kernels used in SVM [14, 15]. The formula for all these kernels are given below:

- Linear Kernal

$$K(x_i, x_j) = x_i^T x_j \tag{4}$$

- Polynomial Kernal

$$K(x_i, x_j) = (x_i^T * x_j + 1)^d \tag{5}$$

- Gaussian Radial Basis Kernal (RBF)

$$K(x_i, x_j) = \exp(-\gamma ||x_i - x_j||^2) \tag{6}$$

- Sigmoid Kernal

$$K(x, y) = \tanh(\alpha x^T y + c) \tag{7}$$

We have first extracted all the embedded text features and then applied an svm classifier from sklearn library. In the classifier, we have used different kernels of SVM to get more accurate results. The process of applying SVM is given in Fig. 5.

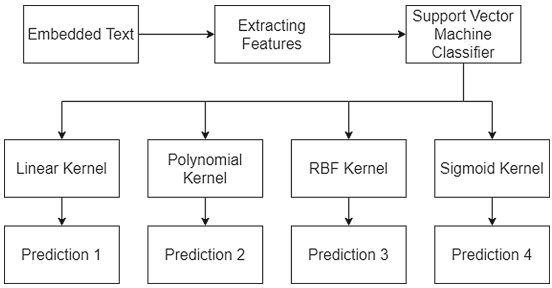

Fig. 5. Process of Support Vector Machine

5) **Convolutional Neural Network.**

CNN is a Deep Learning algorithm that can take input in the form of matrices and can differentiate one from another by assigning importance to various factors. To extract specific features from the data convolutional layers have many filters known as kernels. As we want to analyse sequential data, we are using temporal convolution. We have used Keras with Tensorflow to implement the CNN model. From Word Embedding, we received the vocabulary of the top 90,000 tweets represented by a 200-dimensional vector. We have trained 2 CNN based models [16–18].

- 3 Layer Convolutional Neural Network—First Layer is a Dropout Layer with parameter 0.4, which drops out 40% of the data from the Word Embedding Layer. Then we have used a CNN Layer with 600 filters of kernel size 3 with zero paddings with a relu activation function. After the Convolutional layer, we have applied a max pool layer. Similarly, two more CNN Layers with 300 and 150 filters, respectively. Then we have used Flatten Layer to flatten the vectors. Then we have used a dense layer with parameter 600, which have a dropout layer with parameter 0.5, which drops out 50% of the output. Then we have used a dense layer with a sigmoid activation function to get the model's output. The process of the 3-Layer Convolutional model is shown in Fig. 6.
- 4 Layer Convolutional Neural Network—First Layer is a Dropout Layer with parameter 0.4, which drops out 40% of the data from the Word Embedding Layer. Then we have used a CNN Layer with 600 filters of kernel size 3 with zero paddings with a relu activation function. After the Convolutional layer, we have applied a max pool layer. Similarly, three more CNN Layers with 300, 150 and 75 filters, respectively. Then we have used Flatten Layer to flatten the vectors. Then we have used a dense layer with parameter 600, which have a dropout layer with parameter 0.5, which drops out 50% of the output. Then we have used a dense layer with a sigmoid activation function to get the model's output. The process of the 4-Layer Convolution model is shown in Fig. 7.

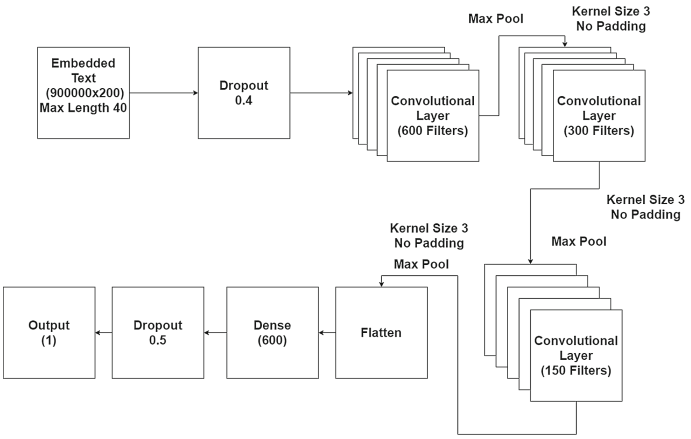

Fig. 6. Process of Convolution 3 Layer Model

6) **Long Short Term Memory (LSTM).**

LSTM is based on RNN architecture used in the deep learning field, and it has feedback connections. It can also process an entire sequence of data points. The core concept of LSTM's is the cell states and their various gates. Information is passed through various gates, which are composed of point-wise multiplication operation and sigmoid layer. The gates can learn what information to keep and forget during training [19–21].

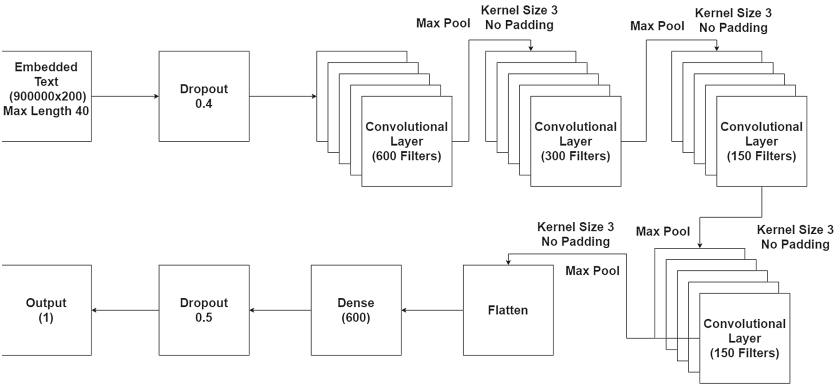

Fig. 7. Process of Convolution 4 Layer Model

We have used Keras with Tensorflow to implement the LSTM model. From Word Embedding, we received the vocabulary of the top 90,000 tweets represented to be a 200-dimensional vector. First Layer is a Dropout Layer with parameter 0.4, which drops out 40% of the data from the Word Embedding Layer. Then we have used an LSTM layer with parameter 128 with relu activation function. Then we have used Flatten Layer to flatten the vectors. Then we have used a dense layer with parameter 600, which have a dropout layer with parameter 0.5, which drops out 50% of the output. Then we have used a dense layer with a sigmoid activation function to get the model's output. The process of the LSTM model is shown in Fig. 8. At the same time, essential components of the LSTM network model are given in Fig. 9.

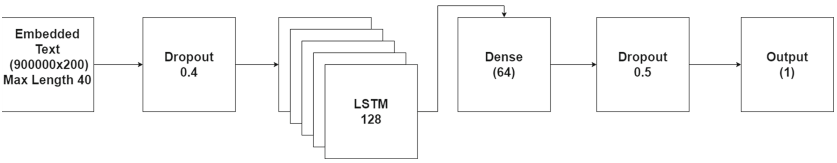

Fig. 8. Process of LSTM Model

- Sigmoid—It is the activation function that gives values ranging from 0 to
- Forget gate—It decides whether to keep the information or forget the formation based on the output of a sigmoid function.
- Input gate—It is used to update cell state. It generates values between 0 and 1 and provides the importance of data.
- Cell State—we can calculate cell state by multiplying cell state by forgetting vector.

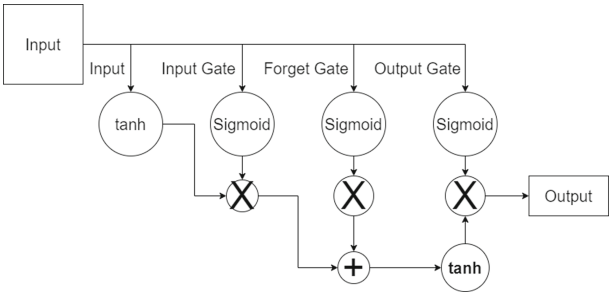

Fig. 9. An essential component of the LSTM Model

- Output gate—It determines the next hidden state for prediction.

7) **Ensemble Technique.**

In the quest to improve accuracy, we have developed a simple ensemble model. We have developed several models and used those models with a Majority vote Ensemble to get the result favoured by most models. We have developed three models for ensemble techniques. They are as follows [22].

- LSTM
- 3-Layer Convolutional Neural Network
- 4-Layer Convolutional Neural Network

we have used a majority vote ensemble to get the final sentimental score. In the Majority vote, we can choose the option coming from a maximum number of models. For example, if we have 3 model ensemble techniques in which two models are giving True, and one model is giving False. So we choose True as a majority of the models are giving True as an answer. The ensemble model is given in Fig. 10.

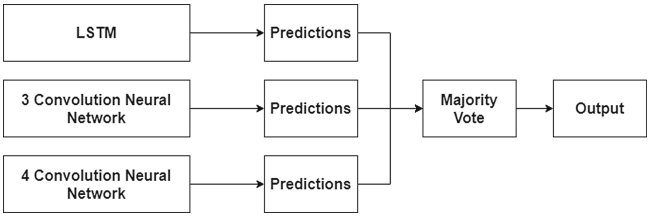

Fig. 10. Process of ensemble model

4 Result Analysis and Discussion

The dataset's data contains some emoticons, URLs, user mentions, symbols, and hashtags, so we need to pre-process the data before using sentiment analysis algorithms. We

have implemented several machine learning and deep learning algorithms. Our dataset contains 800,000 negative and 800,000 positive tweets in text format. So for machine learning algorithms, we have converted them in suitable format by using Word2Vec as a Word Embedding Technique [5]. Figure 11, shows a comparison of different machine learning algorithms for sentiment analysis. From Fig. 11, one can be observed that NLTK has a minimum accuracy of 61.2%, whereas the ensemble model has the highest accuracy of 83.88%. Also, in SVM, RBF Kernel has the most accuracy among all the Kernels, as shown in Fig. 11.

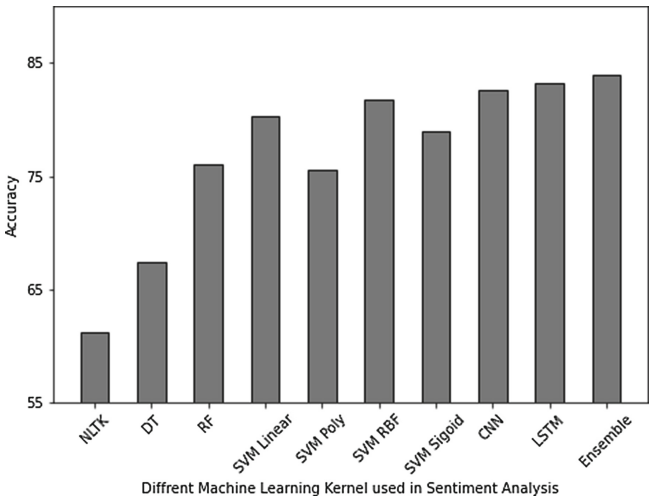

Fig. 11. Comparison of sentiment analysis algorithms

5 Conclusions and Future Work

The Research deals with the different sentiment analysis algorithms we can use and compare their results to get the most accurate results. From our results, we can say that Recurrent Neural Networks like LSTM works best for text classification. We have also created an ensemble model that gives the best results using the majority vote ensemble on three models. For our future work, we can extend the application of our sentiment analysis algorithms. For example, we can use sentiment analysis algorithms to gain knowledge about reviews from Twitter.

For future work, we can extend the application of our sentiment analysis algorithms, for example, sentiment analysis algorithms, to gain knowledge about reviews from Twitter. We can also use it for research purposes like developing a project that can get any topic from twitter and review the topic. For example, in COVID-19, we can gain knowledge about the world's sentiment by analysing tweets. We can also implement Sentiment Analysis algorithms so that they can work with different languages.

References

1. Vinodhini, G., Chandrasekaran, D.: Sentiment analysis and opinion mining: a survey. Int. J. Adv. Res. Comput. Sci. Softw. Eng. **2**(6), 282–292 (2012)
2. Ain, Q.T., Riaz, A., Noureen, A., Kamran, M., Hayat, B., Rehman, A.: Sentiment analysis using deep learning techniques: a review. Int. J. Adv. Comput. Sci. Appl. **8**(6), 424–433 (2017)
3. Feldman, R.: Techniques and applications for sentiment analysis. Commun. ACM **56**(4), 82–89 (2013)
4. Beigi, G., Hu, X., Maciejewski, R., Liu, H.: An overview of sentiment analysis in social media and its applications in disaster relief. In. Pedrycz, W., Chen, S.M. (eds.) Sentiment Analysis and Ontology Engineering. Studies in Computational Intelligence, vol. 629, pp. 313–340 (2016)
5. Dolianiti, F.S., Iakovakis, D., Dias, S.B., Hadjileontiadou, S., Diniz, J.A., Hadjileontiadis, L.: Sentiment analysis techniques and applications in education: a survey. In: Tsitouridou, M., Diniz, J. A., Mikropoulos T. (eds.) Technology and Innovation in Learning, Teaching and Education. TECH-EDU 2018. Communications in Computer and Information Science, vol. 993, pp. 412–427 (2019)
6. Das, S., Behera, R.K., Kumar, M., Rath, S.K.: Real-time sentiment analysis of Twitter streaming data for stock prediction. Procedia Comput. Sci. **132**, 956–964 (2018)
7. Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., McClosky, D.: The Stanford CoreNLP natural language processing toolkit. In: Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 55–60 (2014).
8. Kawade, D.: Sentiment analysis: machine learning approach. Int. J. Eng. Technol. **09**, 2183–2186 (2017)
9. Wang, J. H., Liu, T.W., Luo, X., Wang, L.: An LSTM approach to short text sentiment classification with word embeddings. In: Proceedings of the 30th Conference on Computational Linguistics and Speech Processing (ROCLING 2018), pp. 214–223 (2018)
10. Jongeling, R., Datta, S., Serebrenik, A.: Choosing your weapons: on sentiment analysis tools for software engineering research. In: proceeding of IEEE International Conference on Software Maintenance and Evolution (ICSME), pp. 531–535 (2015).
11. Ramakrishnan, U., Shankar, R., Krishna, G.: Sentiment analysis of Twitter data: based on user-behaviour. Int. J. Appl. Eng. Res. **10**(7), 16291–16301 (2015)
12. Varsha, S., Vijaya, S., Apashabi, P.: Sentiment analysis on Twitter data. Int. J. Innov. Res. Adv. Eng. **1**(2), 2349–2163 (2015)
13. Pletea, D., Vasilescu, B., Serebrenik, A.: Security and emotion: sentiment analysis of security discussions on Github. In: Proceedings of the 11th Working Conference on Mining Software Repositories, pp. 348–351 (2014)
14. Rahman, M.A., Seddiqui, M.H.: Comparison of classical machine learning approaches on bangla textual emotion analysis. https://arxiv.org/abs/1907.07826 (2019)
15. Goyal, M., Gupta, N., Jain, A., Kumari, D.: Smart government e-services for Indian railways using Twitter. In: Sharma, D.K., Balas, V.E., Son, L.H., Sharma, R., Cengiz, K. (eds.) Micro-Electronics and Telecommunication Engineering, pp. 721–731 (2020)
16. Santos, C.D., Gatti, M.: Deep convolutional neural networks for sentiment analysis of short texts. In: Proceeding of 25th International Conference on Computational Linguistics, pp. 69–78 (2014)
17. Zhang, L., Wang, S., Liu, B.: Deep learning for sentiment analysis: a survey. WIREs Data Min. Knowl. Discov. **8**(4), e1253 (2018)
18. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the International Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1746–1751 (2014)

19. Tang, D., Qin, B., Liu, T.: Document modeling with gated recurrent neural network for sentiment classification. In: Proceedings of the International Conference on Empirical Methods in Natural Language Processing, pp. 1422–1432 (2015)
20. Tholusuri, A., Anumala, M., Malapolu, B., Lakshmi, J.: Sentiment analysis using LSTM. Int. J. Eng. Adv. Technol. (IJEAT) **8**(6S3), 2249–8958 (2019)
21. Kurniasari, L., Setyanto, A.: Sentiment analysis using recurrent neural network. In: Journal of Physics: Conference Series, vol. 1471, p. 012018 (2020)
22. Wang, X., Jiang, W., Luo, Z.: Combination of convolutional and recurrent neural network for sentiment analysis of short texts. In: Proceedings of 26th International Conference on Computational Linguistics, pp. 2428–2437 (2016)