International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 04 Issue: 08 | August -2020 ISSN: 2582-3930

Hate Speech Detection in Twitter using Machine Learning and Deep Learning : A TFIDF and LSTM based Approach

Kushagra Srivastava[1], Indresh Verma[2], Bhawna Pareta[3], Abhay Vikram Singh[4]

Department of Information Technology, Babu Banarasi Das National Institute of Technology and Management

Lucknow, India

Abstract- In the modern world, most of our interactions take place digitally through various social media platforms like Twitter and Facebook. People connect and share their thoughts on social media. Despite such benefits, social media is plagued with hate speech and toxic content. In this paper, we discuss how machine learning can help to curb this menace. We propose a machine learning binary classification approach, which classifies tweets into two classes Hate Speech and Neutral Speech. We experimented on various Machine Learning algorithms like Logistic Regression, Support Vector Machine (SVM), Naïve Bayes, and Random Forest, and Deep Learning techniques like Long-Short-Term-Memory (LSTM), a form of Recurrent Neural Network. Models were trained on datasets of tweets available in the public domain. SVM model performed best among the Machine Learning algorithms on this dataset with an accuracy of 92.94% and an F1-Score (Harmonic Mean of Precision and Recall) of 0.83. LSTM model performed like the SVM model, with an accuracy of 92.50% and F1-score of 0.81. Since the dataset is imbalanced, a higher F1-score is preferred in the classifier. In the last module, we built a web interface for the user, connected to the model by a REST API.

Keywords- Hate Speech Detection, Machine Learning, TFIDF, Deep Learning, LSTM, Twitter

I. INTRODUCTION

With the dawn of the 21st century decade and the spread of computational and mobile devices in the hand of masses, has spurred extensive online communication among people. This led to the genesis of various social media platforms like Twitter and Facebook. Through these platforms, people can connect, share, and express their free speech on the internet. Social media websites like Twitter and Facebook generate billions of impressions each day, in the form of comments, likes, shares, and tweets. This unchecked exercise of free speech is abused by anti-social elements, they harass fellow users by cyber-bullying, trolling them, abusing, and sometimes even threats [25]. This leaves a life-long trauma in the minds of the victims; they feel insulted and they are always fearful. All these activities fall into the category of Hate Speech and Toxic content. At worst hate speech can incite violence among different communities, which may harm law and order situations. Cyberbullies can do this because of the non-existence of a robust system, which can detect hate speech, and report it immediately.

Some examples of hate Speech are as followed:

"you are racist bast*rd"

"faggots like you deserve to die"

In this paper, we propose a machine learning-based method to curb this menace. This machine learning model will be to classify a piece of text say a comment or a tweet broadly into two classes of Hate Speech and Neutral Speech. We will make use of binary classification, where 1 signifies hate speech and 0 signifies neutral speech. We have made use of datasets of tweets available in public domain, as well as tweets from twitter employing scraping, through twitter scraper library available in python language. In the dataset, tweets are labeled 0 and 1 for binary classification. After obtaining the data, we preprocess it. The data is dissected into two parts, one as training data, and the other as testing data. We experiment with various machine learning algorithms like Logistic Regression, Support Vector Machine (SVM), Naïve Bayes, Random

Forest. They are trained on the training data and evaluated on testing data. We find that the Support Vector Machine performs the best among the algorithms used. It performs the best, both with and without hyper-parameter tuning. It has an accuracy of 92.94% and F1-score (Harmonic Mean of Precision and Recall) of 0.83. We then experiment with deep learning techniques, using Long-Short-Term-Memory (LSTM), a type of Recurrent Neural Network. With the dataset used in the experiment, the performance of the LSTM model is like the SVM model, it has an accuracy of 92.50% and F1-Score of 0.85. Since the dataset is imbalanced, precision and recall are to be considered, hence LSTM gets an edge over the SVM in the given dataset. It must be noted that as the size of the dataset increases and touches million-mark, the LSTM model starts to outperform the SVM model, owing to its neural network architecture. We create a web interface for the users to check if a tweet is toxic or not. The web interface is connected to the machine learning model via a Restful API.

II. RELATED WORK

Detection of hate speech prevalent in social media websites has been gaining thrust in the recent past. Hate speech and toxic content detection fall in the domain of natural language processing (NLP), which is a very broad field. Professionals belonging to varying fields are interested in it, for instance, psychologists, data science experts, etc. Many notable works have been done in this field, and one such work [10] clearly illustrates the different forms and definitions of hate speech.

NLP techniques play a vital role in classification problems as work is done by [1] shows how sentiment polarity, part-of-speech tagging, unigrams can help us
in the detection of hate speech. Character n-gram, bag-of-words approach and TF-IDF provide an effective way for binary as well multiclass classification, the works of [2],[9],[12] show how above statistical techniques can be coupled with classical machine learning algorithms like Random Forest, Naïve Bayes, Logistic regression and Support Vector Machine to develop a robust classifier.

Deep learning models have also been employed by researchers to detect toxic content on social networking websites, especially Twitter, Facebook, YouTube, and Reddit. An author [4] in his paper has shown the remarkable performance of recurrent neural networks (RNNs) at the detection of hate speech in tweets. In reference [6] the author describes the performance of convolutional neural networks (CNNs), RNNs, and BERT on a “multi-lingual dataset”. In reference [8] the author effectively compares the performance of CNNs, RNNs, fastText, and machine learning models on an anointed dataset of tweets.

III. PROPOSED APPROACH

In this paper, we present a comparative approach to build a classifier, which can classify tweets broadly into two categories, hate-speech, and neutral-speech. We are considering tweets here because our dataset is made up of tweets. We will compare the performances of classical machine learning models and deep neural network models on the labeled dataset. Accuracy will not be our only parameter for judging the performance, since the dataset is imbalanced, precision, recall, F1-score will also be considered. We will make use of TF-IDF for feature extraction in machine learning models and embedding layers in LSTM deep neural network. After obtaining the best performing algorithm, we will build a web interface for the user to check tweets/texts for hate speeches. The UI will be connected to the model via a Flask API.

IV. DATA COLLECTION

Our dataset is a combined dataset of tweets available in the public domain, and tweets directly scraped from Twitter. The tweets are scraped in two ways, one using Twitter API, for this you need to have a Twitter API account and unique key. Personal Key is generated at the time of the creation of the account. We can fetch tweets by providing hashtags as a parameter. But there is one issue, only a limited number of tweets can be fetched in one day. The other way is to use a python library called twitterscraper [13], it allows us to fetch tweets based on hashtags without any limitation. We can specify the timeline in the form of start date and end dates, as well as the number of tweets. The dataset available in the public domain had multi-class labels like threat, obscene, etc. [14]. A part of dataset is available at Kaggle [14]. We have converted the data into binary classification (the label column created only has binary (0/1) values) manually. We have 65,000 tweets in our dataset, and tweets with hate-speech and neutral tweets are present in the ratio of 1:4. Hate Speech tweets are labeled as 1 and Neutral Speech tweets as 0.

TABLE I

Sample of Labeled Dataset

S.NO	Text	Label
1	You and your community are a curse to this nation!!!	1
2	You are a very gentle and a kind person #peace	0
3	We all must unite and fight against common evil	0
4	@Karl I'll kill and hunt you down you dog	1

V. PREPROCESSING

The data obtained in the above step is raw and is filled with noise. Noise can be defined as the presence of any unwanted element in the text. Noise exists in the form of special symbols, HTML tags, emojis, slangs, stop words, hyperlinks, and extra spaces. In this step, we make extensive use of python's NLTK library [9], [17], which has strong capabilities to deal with NLP related tasks. First, tweets are transformed to lowercase, and then the following steps take place to remove noise:

Removing Extra Space
Removing special symbols like '@', '#', etc. Removing HTML tags
Substituting Contractions with their real forms Removing Emojis
Removing Hyperlinks

Further, the tweets go through the processes of Stemming and Lemmatization. Stemming is a process that removes suffix and prefix from the word, and Lemmatization renders the word to its present tense.

After preprocessing, the dataset is divided into two parts, training data, and testing data. Training data is comprised of 70% of the data, it is used for training the models. Testing data is comprised of the rest 30% of tweets, it used to evaluate the performance of the models. The original dataset is shuffled multiple times before splitting, to remove any kind of bias. Shuffling is done to ensure uniform distribution of minority class between training dataset and testing dataset.

VI. FEATURE EXTRASCTION

In this process, we make ample use of statistical techniques like n-grams and TF-IDF. Firstly, tweets are tokenized into words. Tokenized words are used to extract features in form of n-gram [12],[7], where 'n' can vary from 1 to 3. TF-IDF is utilized to weight the features following their importance in the given document. TF-IDF [2], [12] is a statistical technique that helps us to determine the importance of a word in a document. TF-IDF generates a sparse vector or sparse matrix. Sparse vectors are normalized through L2 normalization.

TF-IDF works by calculating the Term Frequency of a word, Inverse Document Frequency of a word, and multiplying them to determine the weight. The formulas are as followed:

tfidf (t, d, D) = tf (t, d) * idf (t, D) [2]

tf (t, d) = (frequency of term t in d document) / (total number of words in document d)

df(t)= number of documents where t is present.

idf(t) = log (total number of documents/ (df +1))

Where t signifies the terms; d signifies each document; D signifies corpus collection.

After obtaining the sparse matrix, the L2 norm is used to normalize the matrix. This method scales the cell values of the row such that, the sum of all the cells of the row will come equal to 1.

Fig.1. Word cloud of unigram

VII. CLASSIFICATION

In this step we have done experiments on both classical Machine Learning algorithms and Deep Neural Network model. In machine learning approach we have utilized major algorithms like Naïve Bayes, Logistic Regression and Support Vector machine (SVM), Random Forest. In deep learning approach we will make use of LSTM Recurrent Neural Network, and we will compare the performances of both the approaches

A. CLASSIFICATION USING MACHINE LEARNING ALGORITHMS

In this step we have done experiments on classical Machine Learning algorithms. In machine learning approach we have utilized major algorithms like Naïve Bayes, Logistic Regression and Support Vector machine (SVM), Random Forest. Input data after preprocessing and feature extraction is passed to the algorithms. The algorithms are trained on training dataset and their and their performance is evaluated on testing dataset. Since the dataset is imbalanced, we will not consider accuracy as the only parameter for performance judgement, but will also consider Precision, Recall and f1-score. At first, we check the performances of the algorithms without any hyper-parameter tuning [2]. We find that SVM performs the best an accuracy of 86% and f1-score of 0.78 and Logistic Regression came second with an accuracy of 80% and f1-score of 0.72. After this we performed grid search [12] to optimize the performance of the algorithms. After tuning values of the parameters, for which respective algorithms were noted, for SVM, the value of parameter c changed to 1.0 and class_weight parameter changed to 'balanced'. After tuning it was found that SVM still performs the best with an accuracy of 92.94% and f1-score of 0.83.

TABLE II.

Performance of Algorithms after Parameter Optimization

S.NO	Algorithm	Accuracy	F1-Score
1	Support Vector Machine	92.94%	0.83
2	Logistic Regression	92.92%	0.80
3	Naïve Bayes	91.43%	0.74
4	Random Forest	58.30%	0.44

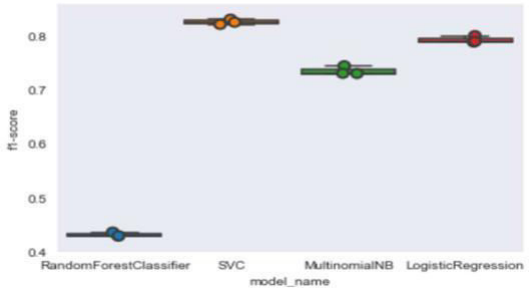

Fig. 2. F1-Score of ML Algorithms

B. CLASSIFICATION USING DEEP NEURAL NETWORK MODEL

In this approach first we tokenize incoming tweets into separate words or tokens, we preprocess them and after preprocessing, they are fed to LSTM architecture, a special form of Recurrent Neural Network [4]. RNNs have been actively used in process pertaining to the field of NLP for example sentiment analysis. RNNs can capture the semantic context of the word, hence are very popular. RNNs suffer from certain issues like vanishing gradient and short-term-memory, i.e. RNNs are unable to remember information. LSTM has removed these problems. LSTM can do it because of Gates and its cells [18],[19], which are able to hold back information for a longer time and take decisions as per the context. Output of the previous cell is passed along with current input, in this way LSTM remembers context while training is going on.

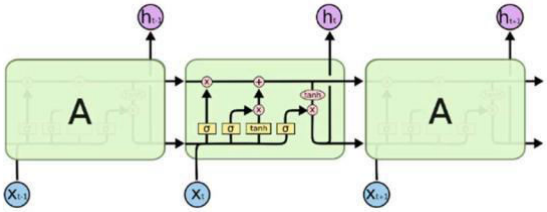

Fig. 3. Working of LSTM architecture [33]

Classification in deep neural network is done through different layers, these layers are provided by Keras library [20]. From Keras we import layers like LSTM, Dense and a very important layer called Embedding layer. Embedding layer creates word embedding after learning from the corpus. It represents word through dense vector in vector space. When an unknown word is encountered by the model, it is mapped in vector space with words with cosine similarity.

We further make use of LSTM, Dense, Dropout and Activation layers. The best results were obtained with 60 units of LSTM layer, with 120 cells in the hidden layer, and 15 epochs with 1024 batch-size. The model obtained an accuracy of 92.50% and an f1-score of 0.85.

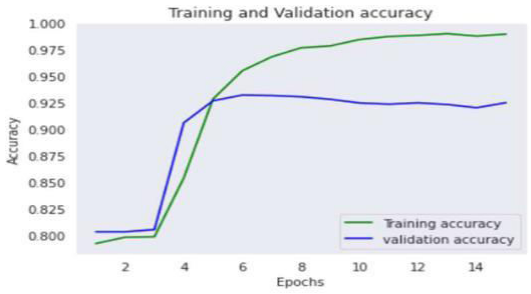

Fig. 4. Validation accuracy of LSTM model

VIII. RESULT

The comparative analysis of classical machine learning algorithms from Table 2 shows that SVM performs the best among them with 92.94% accuracy and 0.83 F1-score. The performance of SVM is then compared with performance of LSTM model. It is observed that performance of both the models is similar, where SVM has slightly higher validation accuracy as well as slightly higher F1-Score. Since the dataset is imbalanced and to reduce bias towards majority class, our classifier must predict minority class accurately. Prediction of minority class is important because we are dealing with a very sensitive subject of hate speech in social media, if our classifier wrongly predicts majority class 0 as 1, it will not cause much damage, but if it predicts minority class 1 as 0, it can cause damage to the society. Hence, it is important that our model has high F1-Score (harmonic mean of Precision and Recall). Therefore, SVM model turns out to be the better classifier among the two. As clear from Table III, SVM approach performs better than LSTM Model on this prepared dataset of 65 thousand tweets.

Table III

Performance Analysis of LSTM and SVM Model

S.NO	Algorithm	Accuracy	F1-Score
1	SVM	92.94%	0.83
2	LSTM	92.50%	0.81

IX. USER INTERFACE

Our final module is to build an interface for the user, which will be utilized by the user to check tweets and texts for hate speech. We have built a web interface using HTML and CSS. This interface is connected to the classifier model by mean of a REST API. We have built a REST API using python's micro-framework Flask. This API takes tweets from the user, pass it to the model as input, and displays the output of the classifier on the UI.

X. CONCLUSION:

SVM provides an efficient and effective approach for Hate Speech and Toxic Content detection in Social Media. SVM performs well on all tasks pertaining to NLP, be it sentiment-analysis or text generation when the size of dataset is under one million . Though both SVM model and LSTM deep neural network perform similar on a smaller dataset, but as observed through experiments that as the size of the dataset increases and it reaches into millions, LSTM classifier starts to outperform SVM classifier by a big margin.

The project leaves a room for research, where latest tools and techniques like pre-trained word embedding algorithms like BERT [21], GloVe [22], word2vec [23], and fastText [24],[8] can be employed to further enhance the efficiency of the classifier.

ACKNOWLEDGEMENT

This research paper has come out as part of the work carried out for the Final Year B-Tech project. We would like to extend our gratitude to Prof. (Dr.) Bhavesh Kumar Chauhan, Director BBDNITM, for allowing us to work on this project. We would also like to thank Prof. (Dr.) Manuj Darbari, HOD IT Department, for his constant motivations, and Prof. Asit Kumar Gahalaut, our project guide, for his guidance and vital suggestions, which were essential in the successful completion of the project.

REFERENCES

[1] Watanabe, H., Bouazizi, M., & Ohtsuki, T. (2018). Hate speech on twitter: A pragmatic approach to collect hateful and offensive expressions and perform hate speech detection. IEEE Access, 6, 13825-13835.

[2] Oriola, O., & Kotzé, E. (2020). Evaluating Machine Learning Techniques for Detecting Offensive and Hate Speech in South African Tweets. IEEE Access, 8, 21496-21509.

[3] L. H. Son, A. Kumar, S. R. Sangwan, A. Arora, A. Nayyar and M. Abdel-Basset. (2019). Sarcasm Detection Using Soft Attention-Based Bidirectional Long Short-Term Memory Model with Convolution Network, in IEEE Access, vol. 7, pp. 23319-23328, 2019, doi: 10.1109/ACCESS.2019.2899260.

[4] Pitsilis, G. K., Ramampiaro, H., & Langseth, H. (2018). Detecting offensive language in tweets using deep learning. arXiv preprint arXiv:1801.04433.

[5] Salminen J., Luotolahti J., Almerekhi H., Jansen B.J., Jung S. (2018). Neural Network Hate Deletion: Developing a Machine Learning Model to Eliminate Hate from Online Comments. In: Bodrunova S. (eds) Internet Science. INSCI 2018. Lecture Notes in Computer Science, vol 11193. Springer, Cham

[6] Modha, S., Mandl, T., Majumder, P. et al. (2020). Tracking Hate in Social Media: Evaluation, Challenges and Approaches. SN COMPUT. SCI. 1, 105. https://doi.org/10.1007/s42979-020-0082-0

[7] Shervin Malmasi & Marcos Zampieri (2018) Challenges in discriminating profanity from hate speech, Journal of Experimental & Theoretical Artificial Intelligence, 30:2,187-202, DOI: 10.1080/0952813X.2017.1409284

[8] Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. (2017). Deep Learning for Hate Speech Detection in Tweets. In Proceedings of the 26th International Conference on World Wide Web Companion (WWW '17 Companion). International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 759–760. DOI: https://doi.org/10.1145/3041021.3054223

[9] Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017). In International AAAI Conference on Web and Social Media. Retrieved from https://www.aaai.org/ocs/index.php/ICWSM/ICWSM17/paper/view/15665/14843

[10] Schmidt, Anna & Wiegand, Michael. (2017). A Survey on Hate Speech Detection using Natural Language Processing. 1-10. 10.18653/v1/W17-1101.

[11Warner, W., & Hirschberg, J. (2012, June). Detecting hate speech on the world wide web. In Proceedings of the second workshop on language in social media (pp. 19-26). Association for Computational Linguistics.

[12] Gaydhani, A., Doma, V., Kendre, S., & Bhagwat, L. (2018). Detecting hate speech and offensive language on twitter using machine learning: An n-gram and tfidf based approach. arXiv preprint arXiv:1809.08651.

[13] pypi.org, 2019 [online]. Available: https://pypi.org/project/twitterscraper/0.2.7/. [Accessed:15-Nov-2019]

[14] Kaggle.com, 2019 [online]. Available: https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data/. [Accessed: 20-Nov-2019]

[15] Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. IEEE transactions on Signal Processing, 45(11), 2673-2681.

[16] Del Vigna12, F., Cimino23, A., Dell'Orletta, F., Petrocchi, M., & Tesconi, M. (2017). Hate me, hate me not: Hate speech detection on facebook. In Proceedings of the First Italian Conference on Cybersecurity (ITASEC17) (pp. 86-95).

[17] Bird, S., Klein, E., & Loper, E. (2009). Natural language processing with Python: analyzing text with the natural language toolkit. " O'Reilly Media, Inc.".

[18] Sak, H., Senior, A. W., & Beaufays, F. (2014). Long short-term memory recurrent neural network architectures for large scale acoustic modeling.

[19] Gers, F. A., Schmidhuber, J., & Cummins, F. (1999). Learning to forget: Continual prediction with LSTM.

[20] Ketkar N. (2017) Introduction to Keras. In: Deep Learning with Python. Apress, Berkeley, CA

[21] Sun, C., Qiu, X., Xu, Y., & Huang, X. (2019, October). How to fine-tune BERT for text classification? In China National Conference on Chinese Computational Linguistics (pp. 194-206). Springer, Cham.

[22] Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).

[23] Goldberg, Y., & Levy, O. (2014). word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method. arXiv preprint arXiv:1402.3722.

[24] Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759

[25] Jhaver, S., Ghoshal, S., Bruckman, A., & Gilbert, E. (2018). Online harassment and content moderation: The case of blocklists. ACM Transactions on Computer-Human Interaction (TOCHI), 25(2), 1-33.

[26] Conover, M. D., Ratkiewicz, J., Francisco, M., Gonçalves, B., Menczer, F., & Flammini, A. (2011, July). Political polarization on twitter. In Fifth international AAAI conference on weblogs and social media.

[27] Habermas, J. (1984). Überlegungen zur Kommunikationspathologie. Jiirgen Habermas (Hg.), Vorstudien und Ergänzungen zur Theorie des kommunikativen Handelns. Frankfurt aM.: Suhrkamp, 226-270.

[28] Nagle, A. (2017). Kill all normies: Online culture wars from 4chan and Tumblr to Trump and the alt-right. John Hunt Publishing.

[29] Fortuna, P., & Nunes, S. (2018). A survey on automatic detection of hate speech in text. ACM Computing Surveys (CSUR), 51(4), 1-30.

[30] Nobata, C., Tetreault, J., Thomas, A., Mehdad, Y., & Chang, Y. (2016, April). Abusive language detection in online user content. In Proceedings of the 25th international conference on world wide web (pp. 145-153).

[31] Warner, W., & Hirschberg, J. (2012, June). Detecting hate speech on the world wide web. In Proceedings of the second workshop on language in social media (pp. 19-26). Association for Computational Linguistics

[32] Zimmerman, S., Kruschwitz, U., & Fox, C. (2018, May). Improving hate speech detection with deep learning ensembles. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018).

[33] colah.github.io, 2020 [online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/