International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-10 Issue-5, June 2021

Machine Learning Based Detection of Deceptive Tweets on Covid-19

Amisha Sinha, Mohnish Raval, S Sindhu

Abstract: ***Social media plays a vital role in connecting people around world and developing relationships. Social Media has a huge potential audience and the circulation of any information does impact a huge population. With the surge of Covid-19, we can see a lot of fake news and tweets circulating about remedies, medicine, and general information related to pandemics. In this paper, we set out machine learning-based detection of deceptive information around Covid-19. With this paper, we have described our project which could detect whether a tweet is fake or real automatically. The labeled dataset is used in the process which is extracted from the arXiv repository. Dataset has tweets, upon which various methods are applied for cleaning, training, and testing. Pre-processing, Classification, tokenization, and stemming/removal of stop words are performed to extract the most relevant information from the dataset and to achieve better accuracy in comparison with the existing system. For classification, we have used two classification techniques- Tf-Idf and Bags of words. To achieve better accuracy, we have used two other methodology-SVM and Random Forest and have achieved an F1-score of 0.94 using SVM.***

Keywords: Artificial Intelligence, Fake News, Social Media, SVM

I. INTRODUCTION

With the commencement of pandemic due to the novel coronavirus, social media has acted as the foremost way of passing the information from one source to the other. With the large and epiphanic flow of data, the certitude of the latter is indiscernible. With the pandemic extent, it is the need for an hour to stratify the large data into bumf and gospel. Our project uses various machine learning techniques to analyze the tweets into “Real” and “Fake”. The focus of the project is to help in analyzing the data and forbid instances of spreading false information. The model created can be also used in real-time fortracking the source of false information. This will help the government of India to take appropriate actions against the culprit and draconianically deplete wrong tweets circulating on the internet.

Manuscript received on June 14, 2021.
Revised Manuscript received on June 19, 2021.
Manuscript published on June 30, 2021.
* Correspondence Author
Amisha Sinha*, Department of Information Technology, SRM Institute of Science and Technology, Chennai (Tamil Nadu), India. Email: amishasinha84@gmail.com
Mohnish Raval, Department of Information Technology, SRM Institute of Science and Technology, Chennai (Tamil Nadu), India. Email: mohnishraval40@gmail.com
S Sindhu, Assistant Professor, Department of Information Technology SRM Institute of Science and Technology, Chennai (Tamil Nadu), India. Email: sindhus2@srmist.edu.in

The project might help to curb the panic and rumor-mongering of the vaccines, symptoms and oxygen cylinders, andVarious medicines like Remdesivir and Dexamethasone.

II. LITERATURE REVIEW

The previous researcher's machine learning techniques and approaches in fake news detection have been quite successful. Sourya Dipta Das at IIT Madras [1] has used a Heuristic Ensemble framework for detecting fake news about Covid-19. They conducted a prolific study, earmarking assorted levels of prerogative to every level of features obtaining an F-Score of 0.98. [2] Mohamad K. Elhadad, Kin Fun Li and Fayez Gebali, constructed a voting ensemble machine learning classifier for detecting inaccurate information for Covid-19 using ten machine learning algorithms and seven feature extraction methods in their survey. They have used Tf-idf and bow classifier techniques which we have also implemented in our project. [3] Cody Buntain and Jennifer Golbeck at the University of Maryland research work on automatic detection of fake news in popular news on Twitter include the usage of three datasets to identify fake tweets around CREDBANK. [4] IRJET paper on “Fake News Detection Using ML” uses Tf-idf, Naive Bayes Support Vector Classifier (SVC) for their fake news detection model. Various researchers have used their own technique and algorithm, for fake news detection using Artificial Intelligence. Fake news around Covid-19 is mostly negative (63.2%) can lead to threats to public health as stated in Article by Md. Sayeed Al- Zaman on “COVID-19-Related Social Media Fake News in India” [5].

In summary, there are many existing works around fake news detection and for detection, the majority of them rely on manually labelled data. Some existing works are around the detection of fake news around the present Covid-19 situation. As the second wave of Covid-19 have hit India, and we are daily encountering a lot of tweets around this with home remedies and other information with respect to oxygen availability and medicines and there is a lot of fake and misleading news which could be a threat to public and the identification of disinformation is need of the hour.

III. DATASET DESCRIPTION

The dataset for identifying fake tweets for Covid-19 was extracted from arXiv which is an open repository for data and scientific papers related to diverse fields and can be accessed online.

This dataset consists of labeled data in two columns with one column having the tweets and the other column marked as Label with data as either '0' or '1'. Here, '0' implies a real tweet and '1' implies a potential fake tweet. The dataset consists of 2140 tweets with 1120 real and 1020 fake tweets. It has a vocabulary size of 63032 with a number of URLs and unique usernames.

IV. METHODOLOGY

The Methodology used in our project is based on basic natural language processing. The methodology consists of esoteric components like - collection of labeled data, processing of dataset, applying classification techniques such as tf-IDF and bow, visualization of the dataset, tokenizing the tweets in the dataset, finding the stems in the tokenized words, and removal of stop words.

A. Pre-processing

In this work, we mainly focus on the polarity of the tweet and the tweet itself. So, we have pre-processed the tweets by removing unwanted details, stemming, tokenization, and removal of stop words. This involves the removal of mentions, retweets, URLs, and non-alphanumeric characters. The main purpose of data cleaning is to refine the quality of the dataset which is critical for modeling processes. The quality of the dataset determines how the model will perform in a specific circumstance. Mentions consist of the tagged tweets with the person's username and retweets are a kind of reply to a tweet that plays no role in determining the nature of the tweet. Similarly, the removal of URLs is crucial as they are links to other websites and is a type of unstructured data with no meaning. Non-alphanumeric characters provide no Meaning to the context and so can be removed. These steps of data cleaningensure structured data which is a salient step before proceedingwith the dataset.

B. Tf-idf technique

TF-IDF is a method used to fetch information supporting the frequency of a term (TF) and its inverse document frequency. It acknowledges a keyword in any given content and based on thenumber of times it appears in a document, it earmarks the importance to it. Taking a hypothetical situation like we have agroup of words that says "Crocin is the answer to COVID-19" and we are keeping that as the reference want to rank the order of priority of other words in the document, can be started simply by eliminating the sentences without this keyword and often, the maximum frequency definition of the word in the document is used to measure the importance, and the count of the word in a given document is referred to as term frequency, and the process is known as term frequency. Talking about Inverse document frequency, within a document, the usage of the words in the document are not equally important. The words like "the", 'a' can be seen ample times in any document, irrespective of its relevance to a particular document. Prepositions, supporting verbs are common to all documents. The relevance of a keywordwithin a document is expressed as a corpus and can be checked through the tf-idf algorithm rule.

$$\text{tf-idf} = \text{tf} \times \text{idf}$$
$$\text{idf(t)} = \log \{n+1/df(d, f) + 1\} + 1$$

Fig.1. TF-IDF Formula

TF-IDF score is the degree of importance of a word in a sentence. The tf score and Idf score are used to obtain information from the text provided by the user.

In our project, we have made sure that the collection of corpora is checked and for that purpose, we have used this approach.

```
print("Precision: ", precision)
print("Recall: ", recall)
print("F-score: ", Fscore)
print("Accuracy: ", accuracy)
```

```
[196]: sc_tf_idf = TweetClassifier(trainData, 'tf-idf')
sc_tf_idf.train()
preds_tf_idf = sc_tf_idf.predict(testData['Tweet'])
metrics(testData['Label'], preds_tf_idf)

Precision:  0.8421052631578947
Recall:  0.9411764705882353
F-score:  0.8888888888888888
Accuracy:  0.8888888888888888
```

Fig.2. Snapshot of TF-IDF result

"Fig.2." shows the snapshot of code, where we have achieved F-Score of 0.88 using this technique.

C. Bag of words

Bag of words is used for text which converts a particular text into a bag of words i.e. it keeps track of the frequency in which such terms appear in a specific piece of content. It is easy to comprehend and implement the algorithm. The bag of words models creates a frequency matrix of the words in the given content. It converts the text into vectors of fixed size. Major steps involved here are Vocabulary determination (1) where all the words found in the document are stored, counting(2), here all the words along with their frequency is determined in vector form. Tokenizer() and tokenizer.texts_to_matrix() are the python functions that are used to carry out these tasks. The bag of words is useful and widely used owing to its simplicity and also for the fact that its computation is inexpensive.

```
[197]: sc_bow = TweetClassifier(trainData, 'bow')
sc_bow.train()
preds_bow = sc_bow.predict(testData['Tweet'])
metrics(testData['Label'], preds_bow)

Precision:  0.68
Recall:  1.0
F-score:  0.8095238095238095
Accuracy:  0.7777777777777778
```

Fig.3. Snapshot of BOW result

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication

"Fig.3." shows the snapshot of the code, where we have achieved F-Score of 0.80 using this technique.

D. Tokenization

Tokenization involves splitting sequences of words (Sentences) into keywords, symbols, and other components referred to as tokens. These tokens can be used as input for the processes. Tokens are mostly separated by whitespace or any line breaks. Only words can be included as tokens(any character with a continuous stream of words). To implement this in any dataset, a Natural language toolkit is used which has both word and sentence tokenization functions. Word_tokenizer() is used to split a sentence into tokens. Python offers various tokenization functions which include (1) Sentence tokenization which is imported from the NLTK library. (2) Punkt Sentence Tokenizer which is used when there are large chunks of data and tokenization is not easy using the regular sentence tokenizer. (3)Tokenize sentences of different languages. (4) Word_tokenizer() (5) Treebank Word Tokenizer which separates the sentence using punctuation and white spaces. (6) Punkt Word tokenizer (7) Using Regular expression which uses regexp_tokenize to tokenize based on the expression. Tokenization leads to the creation of tokens from a series of words which is useful in finding patterns that are used in further steps of stemming and lemmatization.

E. Stemming and Removal of Stop Words

Stemming reduces morphological variants of a word. For instance, it reduces all the words like [likes, liked, likely] to a root word 'like'. It is beneficial in reducing akin words. Some algorithms to implement stemming are - Potter's stemmer, Lovins, Dawson, Krovetz, Xerox, and N-gram. The stemming method is useful to reduce the number of tokens and assign them to a domain vocabulary library. It is also very useful to fetch results in any search engine like google.

```
if gram > 1:
    w = []
    for i in range(len(words) - gram + 1):
        w += [' '.join(words[i:i + gram])]
    return w
if stop_words:
    sw = stopwords.words('english')
    words = [word for word in words if word not in sw]
if stem:
    stemmer = PorterStemmer()
    words = [stemmer.stem(word) for word in words]
return words
```

Fig.4. Snapshot- Stemming & Removal of stop-words

"Fig.4." shows the snapshot of our code where we have used Porter Stemmer for Stemming.

Stop Words syntactically and grammatically supports a document but do not have any significance with the Content of the document. Such stops are removed Before indexing to concise the index itself. NLPTK is used in python to implement it. Here NLPTK supports a corpus module that contains a list of stop words, which needs to be imported, then we define all the English stop words to a variable, then for those words in a sentence and if it does contain such stop words, it's removed from the data frame formed.

Stop Words are basically just common words that are not so significant in the indexing and can be ignored. These stop words can be usual words like "the", "and", "an", 'a' , etc. "Fig.4." depicts how we have removed stop words from our dataset.

V. MODULE DESCRIPTION

We have used the Support Vector Machine(SVM) algorithm andRandom Forest in our project.

A. Support Vector Machine (SVM)

SVM is an algorithm that has been facile to embrace but extremely efficient and versatile in its application. It can be used for classification as well as regression. The framework of this algorithm is illustrated in "Fig.5". The space is depicted as a multi-dimensional space of various groups. In "Fig.5" , the hyperplane must be chosen such that the distance between the support vectors is maximized, referred to as the maximum margin, where support vectors are the points closest to the hyperplane and points of two classes are shown in green and blue.

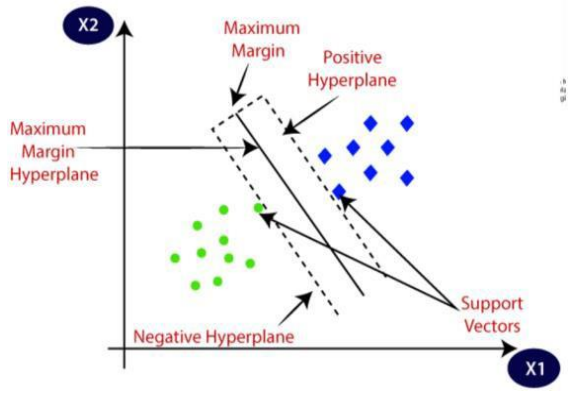

Fig.5. Depiction of SVM model with Maximum marginal between two classes

Using the Hinge loss function, we strive to enhance the marginin the SVM algorithm as shown in "Fig.6".

SVM Hypothesis:

$$h_\theta(x) = \begin{cases} 1 & \text{if } \theta^T x >= 0 \\ 0 & \text{otherwise} \end{cases}$$

Fig.6. Hinge Loss Function

On the condition that the value obtained out of prediction and that out of calculation comes to no difference at all, then the losscalculated is zero, else we calculate the loss using the hinge loss function. In order to balance the maximization of margin and loss, we add cost function, as in "Fig.7"

$$Cost(h_\theta(x), y) = \begin{cases} max(0, 1 - \theta^T x) & \text{if } y = 1 \\ max(0, 1 + \theta^T x) & \text{if } y = 0 \end{cases}$$

Fig.7. Cost Function

For Non-Linear SVM algorithm, the function used is shown in "Fig.8"

$$Hypothesis: \quad h_\theta(x) = \begin{cases} 1 & \text{if } \theta^T f >= 0 \\ 0 & \text{otherwise} \end{cases}$$

$$Cost\ Function: \quad J(\theta) = C[\sum_{i=1}^{m} y^{(i)} Cost_1(\theta^T(f^{(i)}) + (1 - y^{(i)}) Cost_0(\theta^T(f^{(i)})]$$

Fig.8. SVM Function

The only difference between the function used for linear and non-linear SVM algorithms is that 'f' is used instead of 'x', where f is a function of x. This is called the Kernel function. There are various Kernel functions among which "Gaussian function" is the most popular one as it can be used on the condition where there is no prior understanding of the data.

In our project we have used SVM algorithm, a snapshot of the code and the accuracy is shown in " Fig.9".

```
import time
from sklearn import svm
from sklearn.metrics import classification_report
linearclassification = svm.SVC(kernel='linear')
zerotime = time.time()
linearclassification.fit(vectortraining, b_training)
firsttime = time.time()
predictlinear = linearclassification.predict(vectortesting)
secondtime = time.time()
trainingtime = firsttime-zerotime
predictiontime = secondtime-firsttime

print("Training time: %fs; Prediction time: %fs" % (trainingtime, predictiontime))
evaluationrep = classification_report(b_testing, predictlinear)
print(evaluationrep)
```

```
              precision    recall  f1-score   support

         0.0       0.91      0.96      0.94       110
         1.0       0.96      0.90      0.93       104

    accuracy                           0.93       214
   macro avg       0.94      0.93      0.93       214
weighted avg       0.94      0.93      0.93       214
```

Fig.9. Snapshot- SVM algorithm

In a nutshell, it comes to the conclusion that the SVM algorithm works well on the clear margin of separation between classes and is effective where the dimensional frequency is greater than that of samples.

B. Random Forest

This supervised algorithm due to its simplicity and its dual purpose to serve classification and regression because of which it has been the most prevailing technique.

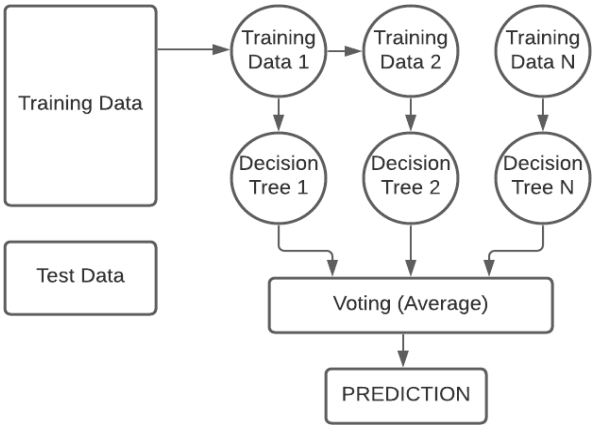

Fig.10. Random Forest Working

The concept of the decision tree is used in the latter which has three cardinal parts: Root node, Leaf node, and branches. At every node, the decision is either the left sub-node or the right node depending on the condition fulfilled in terms of "True" or "False".The verdict taken is determined from the features of the dataset. A minute change in data can produce varied consequences due to its tactfulness.The muddle of overfitting is also eloquently abated due to use of multiple decision trees rather than a single decision tree.

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
rf_model = RandomForestClassifier(n_estimators=100,
                                  bootstrap = True,
                                  max_features = 'sqrt')
import time
from sklearn.metrics import classification_report
zerotime = time.time()
rf_model.fit(vectortraining, b_training)
firsttime = time.time()
predictlinear = rf_model.predict(vectortesting)
secondtime = time.time()
trainingtime = firsttime-zerotime
predictiontime = secondtime-firsttime

print("Training time: %fs; Prediction time: %fs" % (trainingtime, predictiontime))
evaluationrep = classification_report(b_testing, predictlinear)
```

```
Training time: 0.502298s; Prediction time: 0.012629s
print(evaluationrep)
              precision    recall  f1-score   support

         0.0       0.92      0.92      0.92       110
         1.0       0.91      0.91      0.91       104

    accuracy                           0.92       214
   macro avg       0.92      0.92      0.92       214
weighted avg       0.92      0.92      0.92       214
```

Fig.11. Snapshot - Random Forest

In our project we have used Random Forest algorithm, snapshotof the code and the accuracy is shown in "Fig.11"

VI. SYSTEM ARCHITECTURE

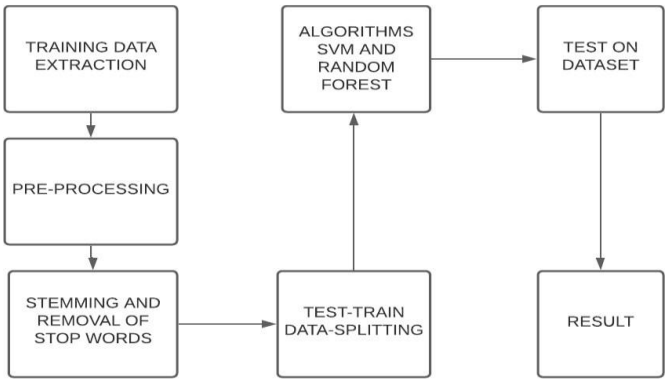

Fig.12. Flowchart- System Architecture

VII. PERFORMANCE OF FINAL METHOD

In our project we have used two classifiers- tf-idf and bag of words and two algorithms - SVM (Support Vector Machine) andRandom Forest.

Table 1: Shows the summary of the accuracies obtained from all the algorithm used:

S.No	Algorithms Implemented	Accuracy (in %)
1	Support Vector Machine(SVM)	94%
2	Random Forest	92%
3	Bayes Theorem(TF-IDF)	88.88%
4	Bayes Theorem(BOW)	77.77%

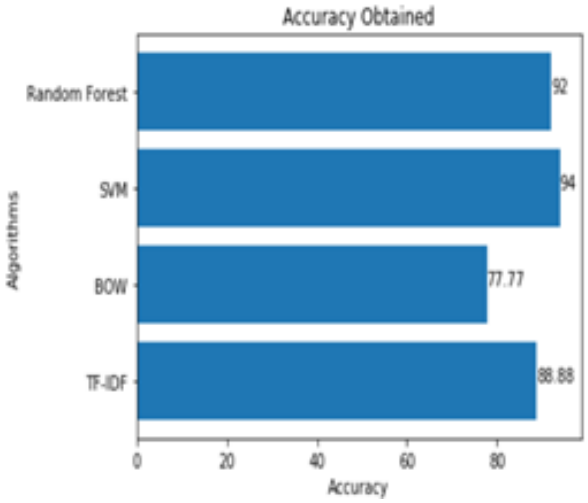

Fig:13 Bar Graph depicting accuracies of various algorithm

VIII. CONCLUSION AND FUTURE WORK

In our model, we achieved an accuracy of 94 percent using the SVM method from the various classification methods as shown in table 1. Perforce, we conclude that the SVM approach has been found to be the most rational. We bring forward this substructure to assist in the identification of misleading info on any probable public health like the 3rd and 4th waves of coronavirus, which are anticipated analogous to writing this survey. This same framework can be used even in the detection of any misleading news, other than global health as well. For our future work, we would like to look at how other pre-trained models work with the same dataset and also our framework performance with other datasets. It'll be riveting to see how our system executes against supplementary assimilated Fake News dataset.

REFERENCES

1. Ayan Basak , Saikat Dutta and Sourya Dipta Das (2020) - "Heuristic-driven ensemble framework for covid-19 fake news detection" (razorthink inc, USA): IIT madras, India
2. Fayez Gebali, (Life Senior Member, IEEE) , Kinfu LI, (Senior Member, IEEE) and Mohamed K. Elhad (2020) - "Detecting Misleading Information on COVID-19" IEEE :Department of Electrical and Computer Engineering, University of Victoria, Victoria, Canada
3. Cody Buntain and Jennifer Golbeck - "Automatically Identifying Fake Newsin Popular Twitter Threads ": University of Maryland
4. Srishti Agrawal, Ruchika Arora, Pronika Chawla et al.-(2020) - "Fake NewsDetection Using ML"- (IRJET) : MRIIRS, Faridabad
5. Md. Sayeed Al-Zaman - Article on "COVID-19-Related Social Media Fake News in India": Department of Journalism and Media Studies, Jahangirnagar University, Savar, Dhaka 1342, Bangladesh
6. Anjali jain, harsh khatter and avinash shakya -(2019) - "A smart system for fake news detection using machine learning" : Dr. Apj abdul kalam university, lucknow, india
7. Aphiwongsophon & Chongstitvatana (2018) - "Detecting Fake News with Machine Learning Method." (2018): Computer, Telecommunications and Information Technology (ECTI-CON) https://doi.org/10.1109/ECTICon.2018.8620051
8. Aswini Thota, Priyanka Tilak, Simrat Ahluwalia et al.- (2019) - "Fake News Detection: A Deep Learning Approach" : Southern Methodist University
9. Uma Sharma, Sidarth Saran, Shankar M. Patil- (2021) - "Fake News Detection using Machine Learning Algorithms" :Bharati Vidyapeeth College of Engineering Navi Mumbai, India
10. Markines.B Cattuto, C & Menczer, F. (2009). —"Social spam detection‖. In Proceedings of the 5th Adversarial Information Retrieval International Workshop on the Web"
11. .H. Gupta, M. S. Jamal, S. Madisetty and M. S. Desarkar (2018) -"A framework for real-time spam detection in Twitter," 10th International Conference on Communication Systems & Networks (COMSNETS), Bengaluru, pp. 380-382
12. Aayush Ranjan — "Fake News Detection Using Machine Learning‖" (2018)-Department Of Computer Science & Engineering, Delhi Technological University
13. C. Buntain and J. Golbeck - (2017) "Automatically Identifying Fake News in Popular Twitter Threads," - IEEE International Conference on Smart Cloud , New York, NY, 2017, pp. 210-215.
14. Cade Metz- (2016)- "The bittersweet sweepstakes to build an AI that destroys fake news. "
15. http://www.fakenewschallenge.org/
16. 1T. Granskogen and J. A. Gulla - (2017)- "Fake news detection: Network data from social media used to predict fakes," CEUR Workshop Proc , pp. 59–64

AUTHORS PROFILE

Amisha Sinha, B. Tech with specialization in Information Technology from SRM Institute of Science & Technology, Kattankulathur, 2020 graduate. Software Engineer. Core Interest: Machine Learning, Data Science, Algorithm development

Mohnish Raval, B. Tech with specialization in Information Technology from SRM Institute of Science & Technology, Kattankulathur, 2020 graduate. Application development Analyst. Core Interest: Machine Learning, Data Science, Algorithm development.

S Sindhu, Assistant Professor (OG) at SRM Institute of Science & Technology, Kattankulathur. M.E with specialization in Computer Science from Anna University, 2011. Research Interest: Data Mining, Big data.