COMPARATIVE STUDY OF VARIOUS MACHINE LEARNING ALGORITHMS FOR TWEET CLASSIFICATION

By

UMAR ABUBAKAR * SULAIMON A. BASHIR ** MUHAMMAD BASHIR ABDULLAHI ***

OLAWALE S. ADEBAYO ****

*,**,*** Department of Computer Science, Federal University of Technology, Minna, Nigeria.
**** Department of Cyber Security Science, Federal University of Technology, Minna, Nigeria.

Date Received: 11/01/2019 Date Revised: 28/01/2019 Date Accepted: 05/04/2019

ABSTRACT

Twitter is a social networking platform that has become popular in recent years. It has become a versatile information dissemination tool used by individuals, businesses, celebrities, and news organizations. It allows users to share messages called tweets with one another. These messages can contain different types of information from personal opinions of users, advertisement of products belonging to all kinds of businesses to the news. Tweets can also contain messages that are racist, bigotry, offensive, and of extremist views as shown by research. Manual identification of such tweets is impossible as hundreds of millions of tweets are posted every day and hence a solution to automate the identification of these types of tweets through classification is required for the Twitter administrators or an intelligence and security analyst. This paper presents a comparative study of traditional machine learning algorithms and deep learning algorithms for the task of tweet classification to detect different categories of abusive languages with the aim to determine which algorithm performs best in detecting abusive language that is prevalent on social media. Two approaches for building feature vectors were explored. Feature vectors based on the bag-of-words method and feature vectors based on word embeddings. These two methods of feature representation were evaluated in this paper using tweet messages representing five abusive language categories. The experiments show that the deep learning algorithms trained with word embeddings outperformed all the other machine learning algorithms that were trained with feature vectors based on the bag-of-words approach.

Keywords: Social Media, Tweets Classification, Feature Extraction, Machine Learning, Artificial Neural Networks, Deep Learning.

INTRODUCTION

Social media platforms such as Twitter and Facebook have become widely accepted as major means of communication. They have become versatile information dissemination tools in this era of digital economy. Twitter allows users (individual, governmental, and non-governmental organizations) to post messages (tweets) for people to see, comment on, and share with other users. Tweets are in the form of text, images, and videos that contain expressions, opinions, and emotions of users, which has led to vast amount of user created content on Twitter. Statistics provided by Twitter as of January 2018 shows that Twitter is being used monthly by 330 million active users who exchange at least 500 million tweets daily (Twitter, n.d). Research has shown that amongst the millions of tweets that are posted every day, are tweets that contain languages and opinions that are racist, offensive, bigotry, and of extremist views and they are occurring now more than ever (Sureka & Agarwal, 2014). Therefore, it is very important to detect these different types of abusive languages before they spread to large number of users and cause societal disturbances indirectly affecting the progress of digital economy. This is also pertinent in understanding the kind of people posting

such abusive content such as in user profiling.

Generally accepted methods of tweet classification usually consists of representing tweets with high-dimensional feature vectors which is then used to train different classifiers. Two of these methods are explored in this paper. The first approach is the bag-of-words method, which represents a text document as a collection of frequently occurring words within the document and the second approach is called word embeddings, which represents a document by grouping similar words together in a vector space. This paper aims to evaluate the performance of various machine learning algorithms with a view to determine which algorithm is better for detecting abusive languages in tweets.

This paper shows the importance of different feature representation methods for training machine learning algorithms for the task of tweet classification. The identification of different types of abusive languages will make it possible to know people's attitude towards different news articles, groups of people, and events in Twitter very quickly and the Twitter administrators can filter out abusive tweets more efficiently. Intelligence and security analysts can identify tweets that incite violence and the seriousness of the degree to which each tweet violates the law.

1. Related Work

Machine learning is a field under Artificial Intelligence that deals with the problem of extracting features from data in order to solve many predictive tasks which is the case with traditional methods. Whereas, deep learning based approaches do not employ feature selection as a separate step, as they are applied directly to the raw data. Deep learning methods have a capability of extracting dependencies among training data. Deep Learning is a new area of Machine Learning research, which has been introduced with the objective of moving Machine Learning closer to one of its original goals: Artificial Intelligence. Deep Learning is about learning multiple levels of representation and abstraction that helps to make sense of data, such as images, sound, and text (Deep Learning Tutorial, 2015). The idea that machine learning is based on is the notion that systems can learn from data, detect patterns, and make decisions with minimal human involvement. Machine Learning methods make use of a training set and a test set for classification. The training set contains feature vectors and their matching class labels as inputs. A classification model is developed which tries to classify the input feature vectors into corresponding class labels using the training set. The test set is then used to validate the model by predicting the class labels of unobserved feature vectors.

Sureka and Agarwal (2014) studied the classification of hate and extremism promoting tweets. The problem of hate and extremism promoting Tweet detection was expressed as a one-class classification problem by the authors and they also proposed several linguistic features. Result showed that Linear SVM outperformed KNN classifier. They concluded that a strong indication of a Tweet to be hate promoting is if it contains some words, which are religious, war related, offensive, and also contains negative emotions. The result also showed that the presence of internet slangs, emoticons, and question mark plays an important role in linear SVM classifier unlike KNN classifier. Uysal and Murphey (2017) carried out a comparative study between different feature based approaches and deep learning for sentiment classification. The authors conducted an in-depth analysis of two different feature selection methods: bag-of-words approach and word embeddings approach. Experiments were conducted using four datasets with varying characteristics. In order to investigate the effectiveness of using word embeddings, feature sets including combination of selected bag-of-words features and averaged word embedding features were used in sentiment classification. For analyzing deep learning models, they implemented three different deep learning architectures, such as convolutional neural network, long short-term memory network, and long-term recurrent convolutional network. The results they obtained from their experiments showed that deep learning models performed better on three out of the four datasets, a combination of selected bag-of-words features and averaged word embedding features gave the best

performance on one dataset. In addition, they showed that a deep learning model initialized with either one-hot vectors or fine-tuned word embeddings performed better than the model initialized using word embeddings without tuning. Zhang, He, Gao, and Ni (2018) used deep learning to detect traffic accidents from social media data. The authors carried out their investigation for one year with over three million tweet contents in two metropolitan areas: Northern Virginia and New York City. Their results showed that paired tokens can capture the association rules inherent in the accident- related tweets and further increase the accuracy of the traffic accident detection. Second, two deep learning methods: Deep Belief Network (DBN) and Long Short-Term Memory (LSTM) were investigated and implemented on the extracted token. Results show that DBN can obtain an overall accuracy of 85% with about 44 individual token features and 17 paired token features. The classification results from DBN outperform those of Support Vector Machines (SVMs) and supervised Latent Dirichlet Allocation (sLDA). Finally, to validate this study, they compared the accident-related Tweets with both the traffic accident log on freeways and traffic data on local roads from 15,000 loop detectors. It was found that nearly 66% of the accident-related Tweets can be located by the accident log and more than 80% of them can be tied to nearby abnormal traffic data. Lee, Lee, Park, and Han (2018) designed a decision system that successfully detects (obfuscated) abusive text using an unsupervised learning of abusive words based on word2vec's skip-gram and the cosine similarity. The system deployed several efficient gadgets for filtering abusive texts, such as blacklists, n-grams, edit-distance metrics, mixed languages, abbreviations, punctuation, and words with special characters to detect the intentional obfuscation of abusive words. The authors integrated both an unsupervised learning method and efficient gadgets into a single system that enhances abusive and non-abusive word lists. The integrated decision system based on the enhanced word lists obtained a precision of 94.08%, a recall of 80.79%, and an f-score of 86.93% in malicious word detection for news article comments, a precision of 89.97%, a recall of 80.55%, and an f-score 85.00% for online community comments, and a precision of 90.65%, a recall of 93.57%, and an f-score of 92.09% for Tweets. Their approach is expected to improve the current abusive word detection system, which is crucial for several web-based services including social networking services and online games. Xianghui, Yuangang, Xiaoyi, and Zhan (2015) developed a method to detect if a Tweet will become popular from a very early stage. Their proposed method involved analyzing the changes of several features over time and perceived a good set of feature combinations and timing to build a Tweet propagation prediction model. The Tweets were then categorized into two classes: popular or not popular, transforming the prediction problem into that of classification. Feature extraction for Tweet classification was looked into by Tsapatsoulis and Djouvas (2017). In their approach, Tweets were represented as multidimensional points in a vector space model. Specifically, binary vectors indicating whether the corresponding term was present or not in the Tweet was used to represent each Tweet. Colloquial Arabic Tweets were classified in real-time to detect high-risk floods in the work of Alabbas, al-Khateeb, Mansour, Epiphaniou, and Frommholz (2017). They represented words in the dataset as frequencies of weighted terms which they generated using TF-IDF weighting method. The weights were then used to train three traditional classifiers and a neural network. Aphinyanaphongs, Ray, Statnikov, and Krebs (2014) carried out a feasibility study on the automatic detection of alcohol-use related Tweets through the classification of texts. They employed four encodings of Tweets (uni-grams, bi-grams, stemmed uni-grams, and stemmed bi-grams) to train Naïve Bayes, linear SVM, Bayesian logistic regression, and random forest algorithms. Semberecki and Maciejewski (2017) conducted a study on how to build effective classifiers for subject text classification of articles using deep learning methods. Their approach involved representing documents as word embeddings using word2vec algorithm and as Bag-of-Words representation. A deep neural network with Long Short Term Memory (LSTM) Units was then trained with these two feature representation methods.

The lack of a comparative analysis between most of the machine learning algorithms for the task of tweet classification in the existing work was the main motivation behind this paper.

1.1 Traditional Machine Learning Algorithms

The traditional machine learning algorithms used for classification in this paper are described below. In this paper, 'traditional approaches' is used to refer to approaches that are based on the bag-of-words model. In the experiments, sci-kit learn which is a machine learning library for python programming language was used to implement the traditional approaches.

1.1.1 Naïve Bayes (NB)

It is one of many different classifiers that are based on the Bayes Theorem and is particularly useful when the dimensionality of the input feature space is high (Wikarsa, & Thahir (2015). It is based on the theory that every feature being classified is independent of the value of any other feature. Given a set of features $X = x_1, x_2, \ldots, x_n$ obtained from a Tweet and a set of target labels $y_1, y_2, \ldots, y_k$, the NB classifier assigns a class y_i with the maximum posterior probability, i.e.

$$P(Y \mid X) = (P(X \mid Y) * P(Y)) / (P(X))$$

where $P(Y|X)$ is the posterior probability (conditional probability of Y given X), $P(X|Y)$ is the likelihood (conditional probability of X given Y), P(Y) is the prior probability (independent probability of Y), and P (X) is the independent probability of X. This algorithm is referred to as "naïve" because of its shortcoming which is that features are not always independent. In a nutshell, the algorithm works by making predictions using probability given a set of features.

1.1.2 Support Vector Machine (SVM)

It is an established model well-suited for linear classification, and is considered to be among the best "off- the-shelf" supervised learning models (Lundeqvist, & Svensson, 2017). SVMs have a theoretical basis derived from statistical learning theory. SVM was originally designed for binary classification. It can however be extended for multiclass classification by breaking the problem down to several binary classifiers, following either one-against-one or one-against-all strategy. Given a binary classification problem and assuming that the training dataset with input vectors $x = (\{x_i\}i=0)$, n where $x_i \in \hat{E}\{y_i\}i=0)n$ where $y_i \in \hat{E}+1, -1\}$. The SVM has two main problems to solve: Find a hyperplane in RN-1 that divides the input space into two subspaces. One subspace for each class; and maximize the margin from the dividing hyperplane to the border vectors, also called support vectors, of both subspaces. The equation of a hyperplane is given as:

$$w.x + b = 0$$

where w is called the weight vector, defining the orientation of the hyperplane and b is called the bias, defining the offset of the hyperplane from the origin. SVM performs an implicit mapping from the input to high-dimension feature space for identifying a clear margin thus, making it a non-probabilistic linear binary classifier. The equation below represents the definition of SVM:

$$f(x) = sgn(w.x + b)$$

1.1.3 K-Nearest Neighbor (KNN)

KNN should be among one of the first choices for a classification task when there is little or no prior knowledge about the distribution of the data because it is one of the most fundamental and simple classification algorithms (Ahmed, Razzaq, & Qamar, 2013). Here, K represents the number of nearest neighbors to be considered for classifying Tweets. It is based on the assumption that points that are close in the feature space are more likely to belong to the same class. A voting scheme where the class with highest votes is assigned as the predicted class is the most common mechanism used for aggregating the k-points. One of several measures used to determine the distance between two points is the Euclidean distance $D(x, y) = \sqrt{(x-y)}2$.

1.1.4 Logistic Regression (LR)

It is an algorithm that is relatively simple and powerful for deciding between two classes, i.e. it is a binary classifier (O'Dea et al., 2015). The logistic function is the core function behind LR and it is also what LR is named after. The logistic function, also known as the sigmoid function is used to map any real-valued number into a value

between 0 and 1, but never exactly at those limits. The equation of the sigmoid function is given as:

$$1 / (1 + e\text{-}z)$$

where e is the base of the natural logarithms and the actual numerical value that is being transformed is z. It basically gives a function that is a boundary between two different classes. It can be extended to handle a multi-class classification problem by a method referred to as "one-vs-all" (multinomial logistic regression or softmax regression), which is really a collection of binary classifiers that predicts the most likely class by looking at each class individually against everything else and then picks the class that has the highest probability. LR is represented by an equation that is similar to that of linear regression. Linear combination of Input values (x) using weights or coefficient values is used to predict an output value (y). A key difference between LR and linear regression is that with LR, the output value being modeled is a binary value (0 or 1) rather than a numeric value. The LR equation is given as:

$$y = e(b_0 + b_1 * x) / 1 + e(b_0 + b_1 * x)$$

where y, b_0, and b_1 are the predicted output, the bias or intercept term and the coefficient for the single input value (x), respectively. Each column in the input data has an associated b coefficient (a constant real value) that must be learned from the training data.

1.1.5 Random Forest (RF)

It is a supervised classification algorithm. As the name suggests, this algorithm creates the forest with a number of trees (decision trees). The RF is a form of nearest neighbor predictor that can also be thought of as an ensemble approach. Ensembles use a divide-and-conquer strategy to improve performance. The belief that a group of "weak learners" can come together to form a "strong learner" is the main idea behind ensemble methods. The RF begins with a standard machine learning technique called a "decision tree" which, in ensemble terms, corresponds to a weak learner. In a decision tree, the data gets bucketed into smaller and smaller sets as it traverses down the tree when an input is entered at the top (Wan & Gao, 2015). That is, it operates by outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees through the building of a multitude of decision trees at training time and RF also corrects for the decision trees' habit of over fitting to their training set.

1.2 Deeep Learning Algorithms

The deep learning models used for classification in this study are described below. In the experiments, Keras deep learning library with Theano as backend was used to implement these models.

1.2.1 Long Short Term Memory (LSTM)

LSTM is a special kind of Recurrent Neural Network (RNN) that was developed to overcome the vanishing gradient problem experienced by RNNs on long sequences of data (Lundeqvist & Svensson, 2017). LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn. The LSTM network includes a new structure called a memory cell, which is comprised of four main elements: a neuron with a self- recurrent connection (a connection to itself), an input gate, an output gate, and a forget gate. This new structure changes the nature of hidden units from "sigmoid" or "tanh" to memory cells whose inputs and outputs are controlled by gates. These gates control flow of information to hidden neurons and preserve extracted features from previous time steps. The function of the gates is to modulate the interactions between the memory cell (ct) itself and its environment. The input gate (it) allows incoming signals to modify or block the state of the memory cell. The output gate (ot) makes it possible for the state of the memory cell to have an effect on other neurons or prevent it. Lastly, the forget gate (ft) has the ability to modulate the memory cell's self-recurrent connection, making it possible for the cell to remember or forget its previous state, as required. The gating equations for the LSTM network are:

$$it = \sigma(wi \; . \; [ht\text{-}1, xt] + bi)$$

$$ft = \sigma(wf \; . \; [ht\text{-}1, xt] + bf)$$

$$ot = \sigma(wo \; . \; [ht\text{-}1, xt] + b0)$$

$$\hat{C}t = \tanh(wc \; . \; [ht\text{-}1, xt] + bc)$$

$$ct = ft * ct\text{-}1 + it *$$

$$\hat{C}t\ ht = ot *$$

$$\tanh(ct)$$

where w, b ht, xt, ct, tanh, and σ are the weights, biases, output vector of LSTM unit, input vector, cell state vector, hyperbolic tangent, and the sigmoid activation functions, respectively.

1.2.2 Gated Recurrent Unit (GRU)

It is a modified version of LSTM. It preserves the LSTM's resistance to the vanishing gradient problem, but its internal structure is simpler, and therefore is faster to train, since fewer computations are needed to make updates to its hidden state. The GRU cell has two gates, an update gate (z), and a reset gate (r) compared to the input, output and forget gates in the LSTM cell (Dey & Salemt, 2017). The update gate controls which information in the previous memory to keep around and the reset gate determines how to combine the new input with the previous memory. The GRU does not have a persistent cell state that is distinct from the hidden state in the LSTM. The main difference between LSTM and GRU is that LSTMs control the exposure of memory content (cell state) while GRUs expose the entire cell state to other units in the network. The LSTM units have separate input and forget gates, while the GRU performs both of these operations together via its reset gate. The GRU gating equations are:

$$zt = \sigma(wz\ .\ [ht\text{-}1\ ,\ xt]\ )$$

$$rt = \sigma(wr\ .\ [\ ht\text{-}1\ ,\ xt]\ )$$

$$\hat{h}t = \tanh(W\ .\ [rt * ht\text{-}1\ ,$$

$$xt])\ ht = (1\ -zt) * ht\text{-}1\ + zt$$

$$* \hat{h}t$$

where zt, rt, w, xt, ht, tanh, and σ are the update gate, reset gate, weight, input vector, output vector, hyperbolic tangent activation function, and the sigmoid activation function, respectively.

2. Methodology

The method employed in this study is depicted in Figure 1 below. It consists of collecting a large amount of training data from Twitter, feature extraction, and training various machine learning algorithms for the task of Tweet classification.

2.1 Data Collection

The labeled dataset of Tweets belonging to all categories were obtained using the Twitter API, which enables programmatic collection of Tweets. The Tweets were collected using certain keywords, such as "nigger", "kill", "hate Muslims", "hate Jews", "kill", "fuck" that indicate abusive language. The distribution of the collected data across the five classes is shown in Figure 2.

2.2 Data Preprocessing

Preprocessing is one of the key components in text classification. With preprocessing, the dataset is transformed from its raw form into a form the learning algorithms can understand. Preprocessing also provides the opportunity to remove noise from the data, which can give more accurate learning algorithms.

This step includes removing URLs, emoticons, special characters and stop words from the dataset and lastly, the dataset is tokenized (converting a sequence of characters or words into a sequence of tokens/strings with an assigned or identified meaning). The target variable is a categorical variable and denotes which class each Tweet belongs to. After preprocessing, labels were manually assigned to each Tweet. There are five classes: 0 denotes that a Tweet is bigotry; 1 denotes that a Tweet is offensive; 2 denote that a Tweet is racist; 3 denote that a Tweet contains extremist views; 4 denote that a Tweet does not contain any abusive language.

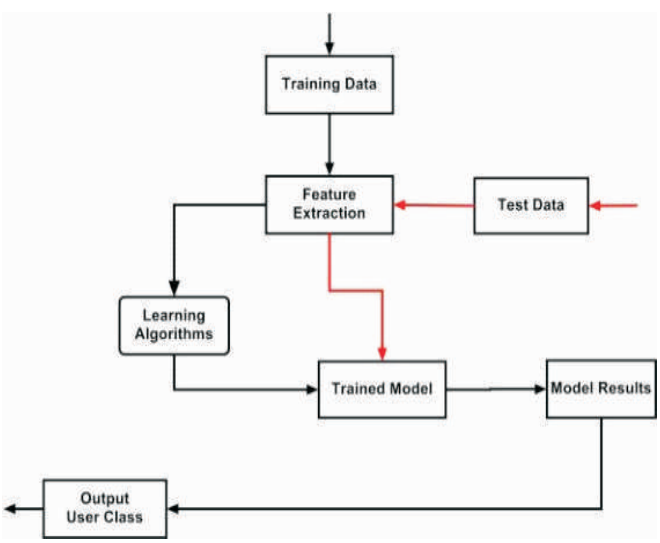

Figure 1. Proposed Tweet Classification Model

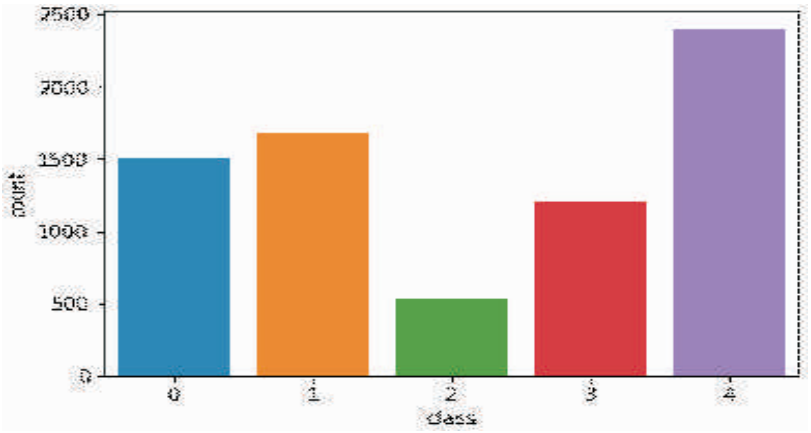

Figure 2. Distribution of Collected Tweets

There is no absolute criterion to judge whether a Tweet is bigotry, racist, offensive, and is of extremist views or neutral and labels depend on certain people's opinion. Ethical problems may exist, for example, if Twitter filters out all the Tweets that they think may include bigotry remarks and extremist views and leaves the rest to the users; freedom of speech can be an issue. Or when an African posts a tweet which contains the word "nigger", it is difficult to determine whether this indicates racism or not.

2.3 Feature Extraction

Once the dataset has been preprocessed, the features that the learning algorithm uses for classification are extracted. The two types of feature representation used in this experiment are:

2.3.1 Bag of Words (BoW)

This approach is very simple and flexible. With this approach, keywords are filtered from training data with the help of some Natural Language Processing (NLP) methods, such as: Tokenization, Stemming, Entity Detection, and Relation Detection (Aphinyanaphongs et al., 2014). Information about the contents of the dataset can be obtained by the creation of such objects from text. The frequencies and appearance of specific keywords, entities serve as the basis for the BoW model. The reason it is referred to as "bag of words" is because it does not make use of any information concerning the order or structure of words in the dataset. The BoW model is not concerned with where words occur in the dataset, but whether known words occur in the dataset or not.

2.3.2 Word Embeddings

Using bag of words as text representation provides little information to the learning algorithms. Using the right text representation for a NLP task can improve the performance of the learning algorithms drastically. An alternative approach for word vector representation is to use word embeddings. This approach provides the learning algorithms with syntactic and semantic information by grouping similar words of a text document together in a vector space. With such a property, algebraic operations can be performed on the embeddings (Lundeqvist & Svensson, 2017). In order to produce accurate word embeddings, the vector space must be trained on a set of texts. There are different algorithms to train a vector space. One popular algorithm uses a shallow, two-layer, neural network to train and is called Word2vec, developed by Tomas Mikolov's team at Google (Mikolov, Chen, Corrado, & Dean, 2013). Another popular algorithm is GloVe, developed at Stanford University (Pennington, Socher, & Manning, 2014).

2.4 Model Setup

The traditional machine learning algorithms used the bag-of-words approach for representing Tweets as feature vectors in this experiment with the following settings:

A version of Naïve Bayes ideally suited for mult-iclass classification problems called Multinomial Naïve Bayes was used in this experiment with default parameters. Given that SVM is a binary classifier, a version that support multi-class cases called LinearSVC from sklearn's machine learning

library was used in this experiment. The normal SVM which has parameter kernel='linear' is very similar to the LinearSVC. The LinearSVC has more flexibility in the choice of penalties and loss functions which makes it possible to scale to large number of samples because it is implemented in terms of liblinear rather than libsvm. The KNN used has the number of nearest neighbors set to 5 and 10. The random forest algorithm was used without changing any default parameters. After finishing random forest, an assumption was made that the tweet classification model is linear. So the training data was fit with Logistic Regression to see if predictions can be improved. To apply Logistic Regression model on a multi-class classification problem, there are various parameters that can be adjusted to find the optimal candidate. For example, sklearn package gives an argument "multi class", which can be set as "multinomial" or "ovr". If the option chosen is "multinomial", the multinomial loss is minimized, which is fitted across the entire probability distribution. If the option chosen is "ovr", the model fits a classifier per class. The class is fitted against all the other classes for each classifier. This option is computationally efficient as only n_class classifiers are required and has the advantage of interpretability. "ovr" is a fair default choice because it is the most widely used strategy for multi-class classification problems. After trying both options, "ovr" was chosen, which perform slightly better and is much faster.

The deep learning algorithms evaluated in this experiment made use of word embeddings as feature vectors and where implemented with Keras deep learning library with theano backend. Keras is a wrapper to tensorflow, which is a deep learning library which makes it possible to implement deep learning algorithms in a few lines of code. The deep learning models in this experiment belong to the many-to-one architecture, where the models are fitted with sequences of inputs and predict one output. Keras offers two different methods to make use of word embeddings to train a neural network. One is to use embedding layer to learn word embeddings. In this layer, each word is represented with a unique integer because it requires that the input data be integer encoded. The Tokenizer API provided with Keras can be used in carrying out this data preparation step. Random weights are used to initialize the embedding layer and all the words in the training dataset will be used to learn embeddings. The Embedding layer is defined as the first hidden layer of the network and it must specify three parameters:

- *Input dimension:* This is the size of the vocabulary of the dataset. For example, 11 words would be the size of the vocabulary if the data is integer encoded to values between 0-10.
- *Output dimension:* This is the vector space size which words will be embedded with. It defines the size of the output vectors from this layer for each word. For example, it could be 32, 50, 100, or some greater value.
- *Input length:* This is the length of input sequences which is used to define any input layer of a Keras model. For example, the input length would be 1000 if all of your input documents are comprised of 1000 words.

The second method is to make use of a pre-trained word embedding model (i.e. word embeddings learned somewhere else), a type of transfer learning. A common practice in the field of NLP is to make available free downloadable word embeddings. These two methods of using word embeddings were used to evaluate the two deep learning models in this experiment. The parameters and layers of both the LSTM and GRU models are shown in Table 1.

3. Experiments

A dataset of 7323 Tweets was used to train and validate the models trained in this experiment. The dataset is divided into 1506 bigotry tTweets, 1678 offensive Tweets, 534 racist Tweets, 1205 extremism related Tweets, and 2400 neutral Tweets. The Tweets were identified using certain keywords that indicate abusive language using Twitter search function.

The Tweets were then downloaded from Twitter using its publicly available API Streaming APIs. The dataset was split into two different sets. One set containing 87% of the total tweets on which the classifiers were trained on and the

Model	Embedding Layer	LSTM Layer	LSTM Layer	Dense Layer
LSTM	Input Dimension = 16709			
	16709	Embedding Size = 250	Embedding Size = 250	Output Dimension = 5
	Output Dimension = 250	Dropout = 0.9	Dropout = 0.9	Activation Function = Sigmoid
	Input Length = 34	Return Sequences = True	Return Sequences = True	
GRU				
Input Dimension	16709	Embedding Size = 250	Embedding Size = 250	Output Dimension = 5
Output Dimension	250	Dropout = 0.9	Dropout = 0.9	Activation Function = Sigmoid
Input Length	34	Return Sequences = True	Return Sequences = True	

Table 1. LSTM and GRU Model Parameters and Layers

other set consists of the remaining 13%, which served as the test set and was used to evaluate the models.

The experiment was carried out on a machine with the configuration of 2.6 GHz duo-core processor and 4 GB RAM memory.

4. Results

The models were trained on an abusive language labeled dataset of tweets with the results obtained shown in Table 2. Naïve Bayes fitted well on both the training data and test data as shown in the table and the confusion matrix is shown in Figure 3. The model obtained an accuracy of 62% on the test data. However, it can be observed that the default Naïve Bayes fitted extremely well on neutral class, but had poor performance for the racist and extremist class. This model could not separate racist speech from speech that is considered offensive and those that do not contain any abusive language accurately. It also had the problem of separating extremism related Tweets from neutral Tweets. This may be due to the data being imbalanced and the algorithm might be biased towards the majority classes because the loss function did not take the data distribution into consideration. SVM model obtained an accuracy of 71% and from the confusion matrix in Figure 4, it can be seen that the model fitted well on all the classes with above 60% accuracy on all classes. The RF model obtained an accuracy of 67% and from the confusion matrix in Figure 5, it can be observed that the default RF fitted well on all the classes. Its performance is however biased towards the majority classes as it got higher scores for those

Models	Metrics				
	Accuracy		Precision	Recall	F1-Score
	Train	Test			
NB	80.00	62.00	74.00	62.00	65.00
SVM	98.00	71.00	71.00	71.00	71.00
KNN (K = 5)	47.00	44.00	64.00	44.00	35.00
KNN (K = 10)	64.00	56.00	61.00	56.00	54.00
LR	83.00	70.00	72.00	70.00	70.00
RF	97.00	67.00	68.00	67.00	67.00
LSTM_E	96.48	87.16	92.82	89.30	91.00
GRU_E	95.64	88.00	90.52	87.38	88.90
LSTM_w2v	95.82	87.67	91.50	85.32	87.97
GRU_w2v	95.31	87.56	89.72	86.17	87.88

Table 2. Results of Machine Learning Models

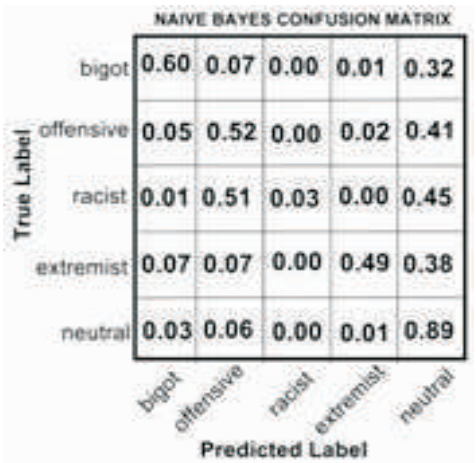

Figure 3. Naïve Bayes CM

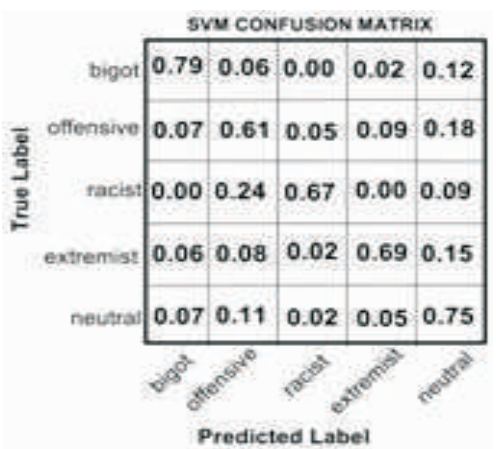

Figure 4. SVM CM

classes with many training examples. This is due to unbalanced nature of the dataset. Logistic regression model obtained an accuracy of 70% on the test set and from the confusion matrix in Figure 6, it can be seen that the model fitted well on all the classes except racist class. It classified 33% of the racist tweets as offensive and 19% as neutral. This may be due to the data being imbalanced and the algorithm might be biased towards the majority classes because the loss function did not take the data distribution into consideration.

The KNN model with k set to 5 obtained the worst accuracy of 47% on the training data and 44% on the test data meaning that it did not fit well on both the training data and test data. Also, as can be seen from the confusion matrix in Figure 7, it did not fit well on all the classes as it classified majority of all the Tweets into the neutral class.

The KNN model with k set to 10 performed a little better than the other model with k set to 5 as shown by the confusion matrix in Figure 8. However, the model still classified majority of all the Tweets as neutral. This may be due to the data being imbalanced and the algorithm might be extremely biased towards the majority class because the loss function did not take the data distribution into consideration.

From the results in Table 2, GRU_E obtained an accuracy of 88.00% outperforming LSTM_E which got an accuracy of 87.16% on the test dataset, and it also shows that LSTM_w2v obtained an accuracy of 87.67%, which is a little higher than the accuracy of GRU_w2v which obtained 87.56% on the test dataset.

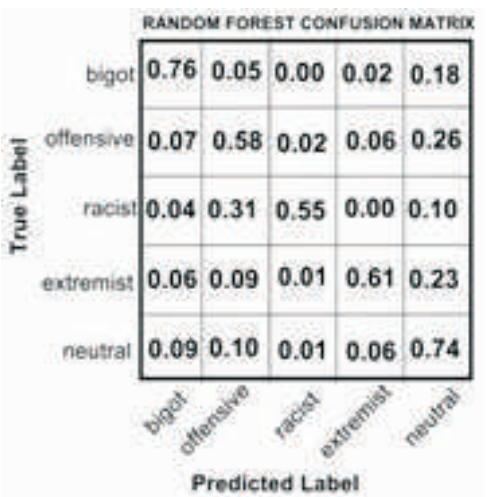

Figure 5. RF CM

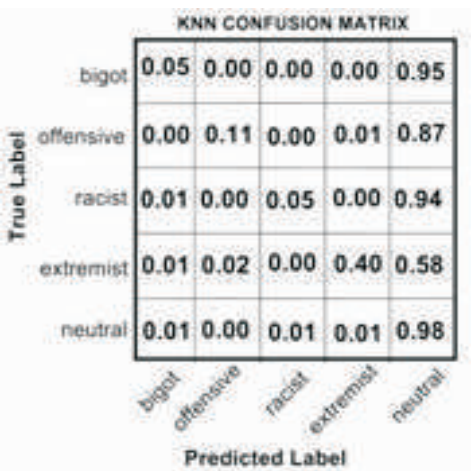

Figure 7. KNN (k=5) CM

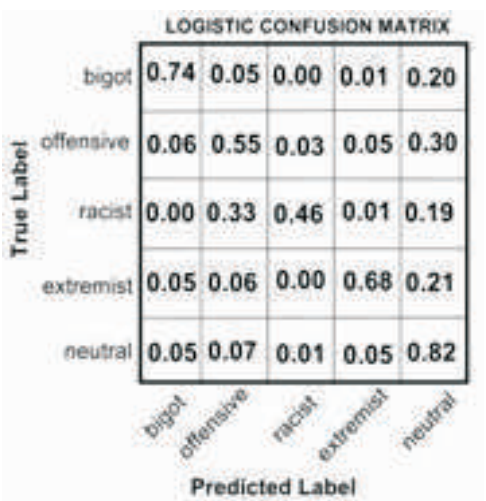

Figure 6. LR CM

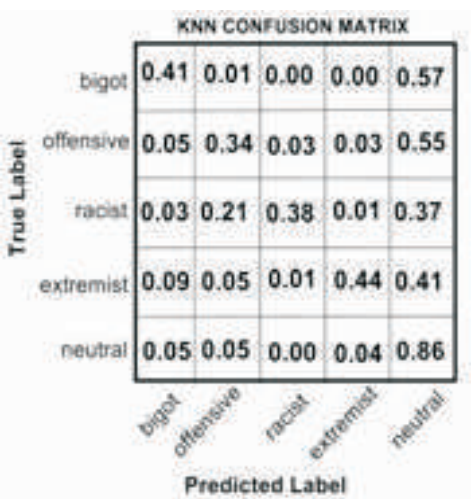

Figure 8. KNN (K =10) CM

5. Discussion

The results from all the machine learning models in this experiment showed that four of the models generally performed well in classifying Tweets into five different categories except when it came to bigot and racist classes. KNN which was the worst performing model got the lowest accuracy of 47% and 56%. This may be due to imbalance in the dataset and the algorithms may be biased towards the majority classes because the loss function did not take the data distribution into consideration. To prove this assumption to be true, there is a need to balance the dataset or to increase the number of Tweets in the racist and extremist classes for training. The deep learning models achieved higher accuracies than all the traditional machine learning models combined despite the dataset being small (7323 Tweets) because deep learning requires a lot of data for training. When the embedding layer was used to learn word embeddings, GRU performed better than LSTM and when weights from trained word embeddings was used to seed the embedding layer, LSTM performed better than GRU.

Conclusion

The aim of this paper was to evaluate the performance of traditional machine leaning algorithms and deep learning algorithms on the task of Tweet classification with a view to determine which algorithm performs better.

The best performing model was identified by comparing the accuracies of all machine learning models trained in this experiment. Out of all the models evaluated, the two deep learning models outperformed all the traditional machine learning algorithms trained in this experiment. LSTM with trained word embeddings and GRU with embedding layer both obtained the highest accuracies in this experiment. And out of the five traditional machine learning models evaluated, the results showed that the overall best performing model was the linear SVM, which outperformed the other models. How well each model performs on Tweet classification can be influenced by different factors, such as the size of the dataset, how balanced the dataset is, the chosen parameters, and how the preprocessing of the raw data is performed.

The results in this study showed that machine learning models performed poorly in classifying Tweets that belong to categories with small number of training examples. This leaves the authors with the conclusion that the performance of the models will be improved if the dataset is balanced or increased.

Future Work

The models in this paper were trained on the text contained in Tweets. The multimodal analysis of tweets that includes images, videos, and emoticons is an important future work. Also, profiling users based on their tweets to detect users with tendency to spread hate speeches, or cause racial tension is another future work.

References

[1]. Ahmed, H., Razzaq, M. A., & Qamar, A. M. (2013, December). Prediction of popular tweets using Similarity Learning. In *Emerging Technologies (ICET), 2013 IEEE 9th International Conference on* (pp. 1-6). IEEE.

[2]. Alabbas, W., al-Khateeb, H. M., Mansour, A., Epiphaniou, G., & Frommholz, I. (2017, June). Classification of colloquial Arabic tweets in real-time to detect high-risk floods. In *Social Media, Wearable and Web Analytics (Social Media), 2017 International Conference on* (pp. 1-8). IEEE.

[3]. Aphinyanaphongs, Y., Ray, B., Statnikov, A., & Krebs, P. (2014, August). Text classification for automatic detection of alcohol use-related tweets: A feasibility study. In *Information Reuse and Integration (IRI), 2014 IEEE 15th International Conference on* (pp. 93-97). IEEE.

[4]. Deep Learning Tutorial (2015). LISA lab, University of Montreal.

[5]. Dey, R., & Salemt, F. M. (2017, August). Gate-variants of Gated Recurrent Unit (GRU) neural networks. In *Circuits and Systems (MWSCAS), 2017 IEEE 60th International Midwest Symposium on* (pp. 1597-1600). IEEE.

[6]. Lee, H. S., Lee, H. R., Park, J. U., & Han, Y. S. (2018). An abusive text detection system based on enhanced abusive and non-abusive word lists. *Decision Support Systems*, 113, 22-31.

[7]. Lundeqvist, E., & Svensson, M. (2017). Author profiling:

A machine learning approach towards detecting gender, age and native language of users in social media. UPPSALA University, Department of Information Technology.

[8]. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781.*

[9]. O'Dea, B., Wan, S., Batterham, P. J., Calear, A. L., Paris, C., & Christensen, H. (2015). Detecting suicidality on Twitter. *Internet Interventions*, 2(2), 183-188.

[10]. Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1532-1543).

[11]. Semberecki, P. & Maciejewski, H. (2017). Deep learning methods for text classification of articles FedCSIC. *ACSIS,* 11, 357-360, Doi: 10.15439/2017F414.

[12]. Sureka, A., & Agarwal, S. (2014, September). Learning to classify hate and extremism promoting tweets. In *Intelligence and Security Informatics Conference (JISIC), 2014 IEEE Joint* (pp. 320-320). IEEE.

[13]. Tsapatsoulis, N., & Djouvas, C. (2017, July). Feature extraction for tweet classification: Do the humans perform better? In *Semantic and Social Media Adaptation and Personalization (SMAP), 2017 12th International Workshop on* (pp. 53-58). IEEE.

[14]. Twitter. (n.d). Retrieved from https://about.twitter.com/company on February 6, 2018.

[15]. Twitter Streaming APIs. (n.d). Retrieved from https://dev.twitter.com/streaming/overview on January 3, 2018.

[16]. Uysal, A. K., & Murphey, Y. L. (2017, August). Sentiment classification: Feature selection based approaches versus deep learning. In *Computer and Information Technology (CIT), 2017 IEEE International Conference on* (pp. 23-30). IEEE.

[17]. Wan, Y., & Gao, Q. (2015, November). An ensemble sentiment classification system of twitter data for airline services analysis. In *Data Mining Workshop (ICDMW), 2015 IEEE International Conference on* (pp. 1318-1325). IEEE.

[18]. Wikarsa, L., & Thahir, S. N. (2015, November). A text mining application of emotion classifications of Twitter's users using Naïve Bayes method. In *Wireless and Telematics (ICWT), 2015 1st International Conference on* (pp. 1-6). IEEE.

[19]. Xianghui, Z., Yuangang, Y., Xiaoyi, W., & Zhan, Z. (2015, December). A classification method to detect if a Tweet will be popular in a very early stage. In *Computing, Communication and Security (ICCCS), 2015 International Conference on* (pp. 1-5). IEEE.

[20]. Zhang, Z., He, Q., Gao, J., & Ni, M. (2018). A deep learning approach for detecting traffic accidents from social media data. *Transportation Research Part C: Emerging Technologies*, 86, 580-596.

ABOUT THE AUTHORS

Umar Abubakar works as a Program Analyst in the Department of Information and Communication of Raw Materials Research and Development Council (RMRDC), Maitama, Abuja, Nigeria from March 2018 till date. He received the B.Tech Degree in Computer Science from the University of Technology Minna, Nigeria, in 2014 and is currently pursuing an M.Tech Degree in Computer Science from the same Institution. Currently, he majors in Computational Intelligence, Machine Learning, and Data Mining. He has up to two Academic Publications.

Dr. Sulaimon A. Bashir is a Lecturer in the Department of Computer Science at the Federal University of Technology, Minna, Nigeria. He is a member of the ACM and Nigeria Computer Society. He received B.Tech, M.Sc and Ph.D Degrees in Computer Science from Ladoke Akintola University of Technology Ogbomoso, Nigeria, in 2003, University of Ibadan, Nigeria, in 2008, and Robert Gordon University Aberdeen UK in 2017, respectively. A recipient of the National Information Technology Development Fund PhD Scholarship (2012) and has various publications to his credit. His research interests include Application of Machine Learning to Activity Recognition, Social Media Mining, and Intelligent Healthcare Systems.

Dr. Muhammad Bashir Abdullahi is a trained Mathematician and Computer Scientist. He joined the services of the Federal University of Technology, Minna in 2001 as a Graduate Assistant in the Department of Mathematics/Computer Science. He later transferred from defunct Department of Mathematics/Computer Science to Department of Computer Science in 2012 after obtaining his PhD degree where he is currently the Head of Department since September 2012 to-date. He received his Doctor of Philosophy (PhD) Degree in Computer Science and Technology from Central South University, Changsha, Hunan, P. R. China in 2012. Dr. M. B. Abdullahi has been involved in teaching and research for over 16 years at both Postgraduate and Undergraduate levels. His research interests are majorly in the areas of Trust, Security and Privacy issues in Wireless Sensor and Ad-Hoc Networks, Internet of Things (IoTs), Software Defined Networking, Peer-to-Peer Networking, Cloud Computing, Big Data Analytics, Machine Learning, Data Mining, Ambient Intelligence, and Computer Science Education. He has supervised/co-supervised several Undergraduate, Master's and PhD students. His research results are published in refereed Journals and Conference Proceedings.

Dr. Olawale S. Adebayo is a Fellow of Institute of Classical Entrepreneurship, a member of Computer Professional Registration Council of Nigeria (CPN), Nigeria Computer Society (NCS), IEEE (Computer Society), Global Development Network, and International Association of Engineers (IAENG) among others. He is a reviewer of many local and International Journals including Computer and Security - Elsevier, Information Sciences - Elsevier, Communication and Security Network among others. Olawale is a lover of peace, justice, fairness, and equity. He earned his PhD in Computer Science from the International Islamic University Malaysia in January, 2017. He earned his MSc in Computer Science from University of Ilorin, Nigeria and Bachelor of Technology in Mathematics and Computer science from Federal University of Technology, Minna, Nigeria in 2009 and 2004, respectively. His current research themes, include Machine Learning, Cryptography, Computer and Information Security. He has published many Academic papers in his Research themes.