PROFILING INAPPROPRIATE USERS' TWEETS USING DEEP LONG SHORT-TERM MEMORY (LSTM) NEURAL NETWORK

By

ABUBAKAR UMAR * SULAIMON A. BASHIR **

LAUD CHARLES OCHEI *** IBRAHIM A. ADEYANJU ****

*-** Department of Computer Science, Federal University of Technology Minna Nigeria.
*** Robert Gordon University, Aberdeen, UK.
**** Department of Computer Engineering, Federal University, Oye-Ekiti, Nigeria.

Date Received: *Date Revised:* *Date Accepted:*

ABSTRACT

In recent times, big Internet companies have come under increased pressure from governments and NGOs to remove inappropriate materials from social media platforms (e.g., Twitter and Facebook, YouTube). A typical example of this problem is the posting of hateful, abusive, and violent tweets on Twitter which has been blamed for inciting hatred, violence and causing societal disturbances. Manual identification of such tweets and the people who post these tweets is very difficult because of the large number of active users and the frequency with which such tweets are posted. Existing approaches for identifying inappropriate tweets have focused on the detection of such tweets without identifying the users who post them. This paper proposes an approach that can automatically identify different types of inappropriate tweets together with the users who post them. The proposed approach is based on a user profiling algorithm that uses a deep Long Short-Term Memory (LSTM) based neural network trained to detect abusive language. With the support of word embedding features learned from the training set, the algorithm is able to classify the tweets of users into different abusive language categories. Thereafter, the user profiling algorithm uses the classes assigned to the tweets of each user to profile each user into different abusive language category. Experiments on the test set show that the deep LSTM-based abusive language detection model reached an accuracy of 89.14% on detecting whether a tweet is bigotry, offensive, racist, extremism-related and neutral. Also, the user profiling algorithm obtained an accuracy of 83.33% in predicting whether a user is a bigot, racist, extremist, uses offensive language and neutral.

Keywords: Twitter, Abusive Language, Tweet Classification, User Profiling Algorithm, Feature Representation, Machine Learning, Deep Learning.

INTRODUCTION

Social networking websites like Twitter create enormous opportunities for users to communicate with one another without having to worry about differences in moral and social values. Due to the popularity of social media sites such as Twitter, communication by users has not only been made convenient, but downright instantaneous allowing users to connect and communicate with anyone utilizing the Internet in seconds. Twitter enables users to post tweets (messages) on their profiles for viewing, commenting and sharing with other users. These tweets or messages are in the form of pictures, text and videos containing users expressions. Data on Twitters site is inherently unstructured because users are often negligent about the spelling and grammatical construction of sentences in their posts. As of July 2018, statistics show that Twitter has a monthly usage by 335 million active users with a daily exchange of more than 500 million tweets leading to the creation of massive amounts of content generated by users on the site (Iqbal,

2019). For this reason, Twitter has become an active platform for user profiling and research related to user personality identification. Recent studies conducted on Twitter include the identification of demographic information about users in the works of Ikeda, Hattori, Ono, Asoh, and Higashino (2013) the use of user tweets to predict brand events by Lee, Oh, Lim, and Choi, (2014) and the identification of user behaviour by Al-Quirishi, Aldrees, AlRubaian, Al-Rakhami, Rahman, and Alamri (2015). Research has also been carried out on learning how to classify hate and extremism related tweets of users on Twitter in the works of Albadi, Kurdi, and Mishra (2018), Alfina, Mulia, Fanany, and Ekanata (2017); Watanabe, Bouazizi, and Ohtsuki (2018); Agarwal and Sureka (2014); Abubakar, Bashir, Abdullahi, and Adebayo (2019).

The communication and exchange of ideas that social media is ideally suited for is crippled by the use of abusive languages. It has damaging effects and causes conflicts among social media users via the hostile environment the use of such type of language creates where users attack one another verbally because of differences in opinion, beliefs and race. This is occurring now more than ever because it is easy to sign up on Twitter and remain anonymous among hundreds of millions of users and tweets.

Existing approaches for detecting inappropriate tweets have focused on the detection of such tweets without identifying the users who post such tweets. Motivated by this problem, the aim of this paper is to find solutions that can identify inappropriate tweets and the users who post them to provide security analysts with a means to counter the spread of abusive posts and online radicalization on one of the largest social networking platforms on the Internet.

In this regard, we propose an efficient approach to detect different categories of abusive posts and users on Twitter social media platform. Our approach is a user profiling algorithm that identifies different categories of abusive users using a deep LSTM (Long Short-Term Memory) neural network trained on an abusive five-class dataset (bigotry, racist, offensive, extremist and neutral tweets) and using word embedding features to detect abusive language.

The main contributions of this paper are:

- An approach based on user profiling algorithm for automatic identification of different types of inappropriate tweets together with the users who post such tweets.
- The identification of a deep learning architecture for abusive language detection that utilizes features derived from messages posted by users online.
- The experimental evaluation of the abusive language detection model on Twitter dataset which demonstrates the top performance achieved on the classification task.

The rest of the paper is structured as follows: in Section 2, we describe some of the existing work and explain some machine learning algorithms briefly. Section 3 defines the objective of the paper and gives a detailed description of the method proposed for the detection of abusive tweets and users. Section 4 presents the results of ourexperimentsanddiscussionoftheresults.Section5istheli mitationsofthestudy.Section6and7providesfor the conclusion of the paper and the possible extensions of the work respectively.

1. Related Work

Research has been carried out recently on profiling users and classifying their behaviours and tweets online with the use of traditional classifiers Ikeda et al. (2013); Lee et al. (2014); Al-Qurishi et al. (2015); Agarwal & Sureka (2014). This is especially true for Twitter micro-blogging service because it is possible to derive user characterization and other useful information by looking at the contents produced by users, or at the actions they perform online with the goal of for example, doing sentiment analysis, classifying users Kang, Yoon, and Kim, 2016; Neethu and Rajasree (2013) and predicting the diffusion of information Rocha, Francisco, Calado and Sofia-Pinto, 2011.

Three different approaches of detecting hate speech related to religion on the Arabic Twitter space were investigated by Albadi et al. (2018), which are n-gram based, lexicon-based and deep learning methods. Two classifiers Support Vector Machine (SVM) and logistic

regression based on the n-gram model were trained in the first method. The labelled dataset was used to create three lexicon terms, each with a true score reflecting its discriminative power towards a polarity of sentiment in the second method. The third approach made use of a deep neural network called Gated Recurrent Unit (GRU) with pre-trained word embeddings as a feature representation method.

The Tumblr microblogging website was used as a case study by Agarwal and Sureka (2016), leading to the development of a cascaded learning model for the identification of user posts with a radicalized or racistintention. Their model was trained to identify different semantic and linguistic features using free text. The authors obtained a total of 3,228 text messages from 2,224 unique bloggers with 10,217 unique tags using the Tumblr Search API. The data was made publicly available to enable their experiments to be used for bench marking and comparison after all duplicate and non-English posts were removed from their dataset. They implemented three different binary classifiers, namely Naïve Bayes (NB), Random Forest (RF) and Decision Tree (DT), to identify and compare posts with extremism or racistintent.

Detection of hate speech in the Indonesian language: a dataset and a preliminary study was the work of Alfina et al. (2017). The authors used the bag of words feature representation method to represent the texts contained in their dataset. They used three types of features: n-gram character, n-gram word and negative emotion to train four machine learning algorithms, namely Naïve Bayes, Logistic Regression, SVM and Random Forest, to detect hate speech.

Fatahillah, Suryati and Haryawan, (2017) implemented the Naïve Bayes algorithm on social media for the task of detecting hate speech in the Indonesian language. Their approach consisted of data collection using the Twitter rest API (Twitter, 2018). The collected tweets were then labelled into two categories (positive as in contains hate speech and negative as in does not contain hate speech). The authors then carried out pre-processingonthedataset. The dataset was then used to train Naïve Bayes machine learning algorithm for the problem of hate speech detection.

An approach was designed by Watanabe et al. (2018) for the detection of hate speech on Twitter. Their approach consisted of extracting four different features from their dataset which they annotated in to three different categories namely; offensive, hate fuland clean. The features made use of by the authors includes entiment features which they believed allowed them to extract polarity of a tweet (positive or negative), semantic features which allowed them to identify any featured expression, the detection of explicit forms of hate speech using unigram features and pattern features for the identification of any longer or implicit forms of hate speech. These features were then used to train three different machine learning algorithms, such as Random Forest, SVM and J48 graft.

Agarwal and Sureka (2014) studied how tweets promoting hate and extremism should be classified. The problem of hate and extremism tweets identification was formulated as a binary classification problem with several proposed linguistic features. The method proposed by the authors is a multi-step process consisting primarily of six phases: experimental dataset collection, dataset creation, pre-processing, extraction of features, classification and performance assessment. The authors implemented two standalone classifiers (KNN and LIBSVM) to classify a tweet as hate promoting or unknown.

Detecting offensive languages in tweets using deep learning was carried out by Georgios, Heri, and Helge, (2018). They combined word embedding features with are current neural network called Long Short Term Memory (LSTM) to determine whether a tweet is racist, sexistor neutral. They were able to obtain good results through the evaluation of their approach on a dataset of 16000 tweets.

Hate speech detection with comment embeddings was investigated by Nemanja, Jing and Robin, (2015). The authors used paragraph 2 vecto learn distributed word representations from their dataset which they used to train a binary classifier for the task of hate speech detection.

Abusive Language detection in online user content was

the work of Chikashi, Joel, Achint, Yashar, and Yi, (2016). The authors created a model with word embeddings used for feature representation.

Five different categories of abusive messages were identified in the works of Abubakar et al. (2019). The authors evaluated and compared the performance of several traditional machine learning algorithms and deep learning algorithms on the problem of hate speech detection so as to identify the best performing algorithm. They evaluated two different feature representation methods namely the Bag of Words model (BoW) and the word embeddings model. The authors identified the word embeddings model combined with the deep learning algorithm called Long Short-Term Memory to perform efficiently on the problem of abusive language detection.

These existing works on hate speech detection with the exception of the works by Abubakar et al. which is being extended by this paper focused on identifying abusive languages in one region or another and also to determine if a piece of text contains hate speech or not without identifying the type of hate speech and the users posting such types of messages. Therefore, the different categories of abusive languages expressed by some users online are identified in this paper together with the identification of users posting such types of abusive languages.

1.1 Machine Learning Algorithms

Many different machine learning algorithms have been used to detect abusive language and hate speech on social media platforms including traditional machine learning algorithms such as Naïve Bayes, SVMs, Logistic Regression, and Random Forest, as well as deep learning algorithms such as LSTM and GRU. A brief explanation of the working theory behind these algorithms is given below.

Naïve Bayes (NB): is a machine learning algorithm that is based on the Bayes Theorem and is particularly well suited to situations with high dimensional input feature space Wikarsa and Thahir (2015). It works by the theory that a feature used for classification does not depend on the value of any other feature. For example, a set of features ($X = x_1, x_2, \ldots, x_n$) extracted from tweets, with the respective target label $Y = y1, y2, \ldots, yk$. $P(Y \mid X) = (P(X \mid Y)*P(Y))/(P(X))$, is assigned by an algorithm to yi with the maximum posterior probability. With $P(Y|X)$ being the posterior probability, $P(X|Y)$ the likelihood $P(Y)$ is the independent probability of Y and $P(X)$ is the independent probability of X. The limitation of this algorithm is that features being classified are not always independent and is the reason why it is referred to as "Naïve". The algorithm works by predicting the classes of a given set of features.

Support Vector Machine (SVM): is a proven model ideally suited for problems of linear classification. It is also seen as one of the best supervised algorithms of machine learning Lundeqvist and Svensson (2017). Originally, it was designed for binary classification problems. With multiclass classification problems, it is extended using a strategy of one-against-one or one-against-all by breaking the problem down to several binary classifiers. When an SVM is presented with a binary classification problem with a dataset made up of input vectors $x = (\{xi\}i=0)$ n where $xi \in R(N- 1)$ and the classes of the input vectors $y = (\{yi\}i=0)$ n where $yi \in \{+1, -1\}$. The SVM must fulfil two main purposes: firstly, a hyperplane must be located in RN-1 separating the input space in two subspaces. This means that each class has one subspace; the second objective is to increase the margin from the separating hyper plane to the border vectors of both subspaces. The hyperplane equation is given as $w.x+b=0$. With w being the vector that defines the orientation of the hyper-plane and b is the bias that defines the offset of the hyperplane from the origin. SVM is a binary classification algorithm that is not based on the theory of probability because it defines a clear margin, by implicitly mapping from input to the high-dimensional feature space. The equation for SVM is given by: $f(x)=sgn(w.x+b)$.

Random Forest (RF): it is an algorithm that creates a forest with a number of trees (decision trees). The RF is also a form of nearest neighbour predictor an ensemble approach. Ensembles improve performance through the use of a divide-and-conquer strategy. The ensemble methods are based on the idea that a group of "weak

learners" can be combined to form a "strong learner". RF begins with a standard "decision tree" machine learning technique that is equivalent to a weak learner interms of the whole. When an entry on the top of the decision tree is entered, the data are collected into smaller sets as it run through the tree (Zhang, He, & Ni, 2018). This means that a multitude of decision trees are built during training, because it produces the mode of the classes for the problem of classification, the mean prediction of the individual trees for a regression problem. RF makes adjustments accordingly in order to counter act the habit of the decision trees in over-fitting to the training set.

Logistic Regression (LR): is a machine learning algorithm that is simple to implement and also powerful for binary classification problems O'Dea, Wan, Batterham, Calear, Paris and Christensen, 2015. LR is named after the logistic function because it is the theory it is based on. A real number is mapped by the logistic or sigmoid function to values between 0 and 1 but never exactly at these limits. The logistic function equationis: 1/(1+e-z). There al numerical value z is transformed wi the representing the base of the natural logarithms. It is extended by a strategy called "one- vs- all" when dealing with multi-class classification problems by collecting binary classifiers that estimate the most likely output by looking at each output separately from other outputs and then choosing the output with the highest probability. The equation representing LR is y = e(b+b* x) / 1+ e(b+ b * x). The predicted outputs are represented by y, b0 and b1, x being the biasor intercept term and the single input value coefficient (x). An associated b coefficient (a constant real value) must be learned from the training data in each column of the input data.

1.2 Deep Learning Algorithms

Deep learning is a sub-field of machine learning. It was introduced with the objective of bringing machine learning closer to one of its original goals; artificial intelligence. Deep learning algorithms help to understand data such as text, images and sound by learning different levels of representation and abstraction (Rahul & Salemt, 2017). Two deep learning algorithms known as LSTM and GRU were evaluated in this paper and the working theory behind these two algorithms is given below.

Long Short Term Memory (LSTM): is a specialty peofre current neural network (RNN). It is the algorithm that was developed in order to get around the disappearing gradient problem experienced by RNNs on long sequences of data (Lundeqvist & Svensson, 2017). As part of their default behaviour, LSTM has the ability to remember information without difficulty for a long time. LSTM has been created by adding a new structure called a Memory Cell to the RNN architecture. The new structure consists of four main components: a self- recurring neuron, an input gate, an output gate and a forget gate. The nature of the hidden units is changed from "sigmoid" or "tanh" to memory cells controlled by gates of this new structure. These gates control the interactions between the memory cell (ct) it self and its environment. The input gate (it) function is to determine whether incoming signals can alter or block the memory cell status. The forget gate (ft) allows the cell to remember or forget its previous state as required because it has the ability to control the self-recurring connection of the memory cell. Finally, the output gate (ot) allows the memory cell to affect or prevent other neurons. The gating equations for the LSTM network are:

$$c_t = f_t * c_{t-1} + i_t * \hat{C}_t$$

$$i_t = \sigma(w_i . [h_{t-1}, x_t] + b_i)$$

$$f_t = \sigma(w_f . [h_{t-1}, x_t] + b_f)$$

$$o_t = \sigma(w_o . [h_{t-1}, x_t] + b_0)$$

$$\hat{C}_t = \tanh(w_c . [h_{t-1}, x_t] + b_c) \; h_t = o_t * \tanh(c_t)$$

With c_t, w, b h_t, x_t, σ, and tanh being the cell state vector, weights, biases, LSTM unit output vector, input vector, sigmoid and hyperbolic tangent activation functions respectively.

Gated Recurrent Unit (GRU): is also an RNN based on the LSTM network architecture. The internal structure of the GRU is simpler than that of LSTM, making it faster to train because it requires fewer calculations to update its hidden state and it also preserves the resistance of the LSTM to the disappearing gradient problem. The GRU cell structure consists of two gates, an update gate (z) and a

reset gate (r) compared to the LSTM cell's three gates (Rahul & Salemt 2017). The update gate function is to determine what information to keep from the previous memory and to determine how to combine new inputs with previous memory. The big difference between LSTM and GRU is that while LSTMs control the exposure to memory (cell state), the GRU exposes its entire cell state to other network units. Another difference is that while LSTM units have different inputs and forget gates, GRU performs these two operations together via its reset gate. The GRU gating equations are:

$$zt = \sigma(w_z . [h_{t-1}, x_t])$$

$$rt = \sigma(w_r . [h_{t-1}, x_t])$$

$$\hat{h}t = \tanh(W . [r_t * h_{t-1}, x_t]) \quad ht = (1-z_t)*h_{t-1} + z_t*\hat{h}_t$$

The update gate, reset gate, weight, input vector, output vector, the sigmoid and the hyperbolic tangent activation function are represented by z_t, r_t, w, x_t, h_t, σ and tanh respectively.

2. Methods

2.1 Data Collection

Two different datasets were created, user profiling dataset and abusive language detection dataset.

2.1.1 User Profiling Dataset

The tweets of users who post abusive messages together with their account names are collected, with the tweets labelled into the corresponding abusive language categories that users will be classified into (bigots, racists, extremists, and offensive) using a classifier. The user profiling dataset was obtained in a semi-automatic manner with Twitter account handles for each class first manually gathered over the Internet using the Twitter search function and then using the Twitter API to extract the Twitter feed of users. The account handles obtained during this research are anonymized and will not be disclosed to the public so as not violate the privacy of any user. For each user identified to belong to our interest categories, 500 of their most recent tweets are collected as from January, 2018. Users that don't have upto 500 tweets are ignored and there tweets of users are also ignored. After collection, the dataset is classified using our detection model. There are five classes: 0 denotes that a user tweets content that is bigotry; 1 denotes that a user tweets content that is offensive; 2 denotes that a user tweets content that is racist; 3 denotes that a user tweets content that is of extremist views; 4 denotes that a user tweets content that does not contain any abusive language. An assumption is also made that if the messages that a user posts belongs to one category, then his connections (followers and following) are also likely to be tweeting content that be long to that same category. So randomly for each user, ten (10) of his/her connections (five user followings and five user followers) most recent 200 tweets are also collected. It can be argued that the accounts followed by users has nothing to do with the ideology or acceptance of the opinions expressed in the tweets of the account being followed. We take this into consideration in our user profiling algorithm by giving the user polaritya higher score than the following and follower polarity. Given that we needed a classifier to classify users tweets into different categories of abusive languages, we also created an abusive language detection dataset.

2.1.2 Abusive Language Detection Dataset

Tweets belonging to all abusive language categories were obtained with the help of the Twitter API. A set of twenty keywords and key phrases were used to identify different categories of abusive languages. They include: "kill", "bomb", "nigger" "Muslims are terrorists", "Jews are dirty", "fuck", "Islam is evil", "wetbacks", "greasy monkey", "kill whites", "kill blacks", "should die", "porch monkey", "burn in hell", "religion of peace", "killing apostates", "creeping sharia". These keywords were obtained with the help of the Twitter Search function which made it possible to view different categories of abuse towards different groups of people. After data collection, we manually annotated the tweets into five classes: 0 – is bigotry; 1 - is offensive; 2 - is racist; 3 - contains extremist views; 4 - does not contain any abusive language. Though it is possible for a bigotry or racist tweet to also belong to the offensive class, we make a distinction between an offensive tweet which targets religion or race and an offensive tweet in general. These classes were chosen based on the works of Albadi et al. (2018); Alfina et al. (2017); Agarwal and

Sureka (2014), on the detection of hate speech towards religion, race and ethnicity. These classes were extended to include offensive speech and extremist views in this paper for the task of detecting different categories of abusive languages. A total of 20, 237 tweets was collected. After annotating the dataset, the dataset was discovered to be unbalanced (2361 bigotry tweets, 4676 offensive tweets, 3360 racist tweets, 3185 extremist tweets and 6655 neutral tweets). From the work of Alfina et al. (2017), an unbalanced data set was said to have a negative effect on the accuracy of classification because the unequal distribution between majority and minority classes in the dataset tends to make the majority class more accurate than the minority class. With this in mind, an under-sampling method was used to reshape the original dataset into a balanced dataset. The original 2361 bigotry tweets were retained in the new dataset and randomly chose 2361 out of 4676 offensive tweets, 2361 out of 3360 racist tweets, 2361 out of 3185 extremist tweets and 2361 out of 6655 neutral tweets. The balanced dataset has a size of 11, 805 tweets.

2.2 Data Pre-Processing

Pre-processing is an important step in text classification. In this step, the dataset is cleaned and transformed into a form the learning algorithms can understand. By cleaning the dataset, noise and unwanted features are removed from the dataset which can give more accurate learning algorithms. We adopted some of the pre-processing steps used by Albadi et al. (2018); Watanabe et al. (2018); Agarwal and Sureka (2014) and Neethu and Rajasree (2013). The steps followed in this stage include:

- Conversion of each word to lower case: this avoids having multiple copies of the same words (e.g. disaster and Disaster will be taken as different words if they are not both converted to lowercase).
- Removal of URLs (e.g.https://bit.ly/mEwrt).
- Removal of emoticons and special characters.
- Removal of characters that are not in a–z.
- Removal of stop words from the dataset: commonly occurring words in the English Language are removed from the dataset (e.g. if, to, and, of, you).
- The dataset is to kenized (conversion of text into a sequence of words or sentences).

2.3 Feature Representation / Extraction

After the pre-processing stage, the features that the learning algorithms will use for classification are extracted. The two widely used methods for classifying tweets and detecting hate speech are the Bag of Words (BoW) and the Word Embedding model. However, from the results of the research conducted and according to the findings of Alfina et al. (2017), the Bag of Words model does not perform well in detecting hate speech. Hence, word embeddings were chosen as a feature representation method in this paper.

2.3.1 Bag of Words

This model uses Natural Language Processing (NLP) techniques such as: stemming, tokenization, relationship detection and entity detection to extract keywords from training data Aphinyanaphongs, Ray, Statnikov, and Krebs (2014). Creation of these objects from text makes it possible to obtain useful information about the contents of the dataset. The BoW model is based on frequently occurring keywords and entities within a document. The BoW does not care where words in the dataset occur, but whether or not known words occur in the dataset. It is therefore referred to as the "bag of words" because it does not use information about the order or structure of words in the dataset.

2.3.2 Word Embeddings

The performance of any learning algorithm can be improved significantly for any NLP task with the right feature representation method. With word embedding, syntactic and semantic information is provided to the learning algorithms by grouping together words from a text document in a vector space. This makes it possible for algebraic operations to be performed on the embeddings Lundeqvist and Svensson (2017). The vector space needs to be trained on a set of texts so as to produce accurate word embeddings. Two popular algorithms for learning word embeddings are Word2Vec and GloVe. Word2Vec trains using a shallow two-layer neural network. It was developed by Tomas Mikolov□s

team at Google Mikolov, Chen, Corrado and Dean (2013) while GloVe was developed at Stanford University by Pennington, Socher and Manning (2014). In this paper, Word2Vec was used to learn word embeddings with aid of Gensim library Rehurek (2019). There are two different methods of using word embeddings. The first method uses the embedding layer of a neural network to learn embeddings. The words in this layer are represented by unique entries because the input data needs to be integer encoded. The embedding layer is initialized with random weights and all the words in the dataset are used to learn embeddings. The second method is to use word embedding learned elsewhere, a type of transfer learning. This is possible by using Word2Vec or GloVe to learn word embeddings from a text document. These two methods of making use of word embeddings were implemented together with different hyper-parameters so as to evaluate different LSTM network models with the objective of identifying the best performing model.

2.4 User Profiling Algorithm

The user profiling algorithm developed in this paper is capable of multidimensional analysis of the labelled dataset of tweets of users and their connections for identifying the different abusive category of each user in the dataset. The algorithm works by frequency analysis of the labels assigned to the tweets of each user and his/her connections by the abusive language detection model. The algorithm looks at the labels of each users' tweets and outputs the label with the highest number of occurrences (user polarity) i.e. the mode (Mo). It also looks at each users' connections (following and followers) and outputs the mode of those connections (following polarity and follower polarity).

The user profiling algorithm as shown in Figure 1 takes the tweets of users in our interest categories and their connections (followings & followers) as inputs. These tweets are then classified by the LSTM model that has been pre-trained to recognise and classify tweets that belong to any of the five interest categories of our work. The details of the LSTM network architecture are presented insection 3.6.

Algorithm 1: User Profiling Algorithm

Data: user_tweets= {t_1...t_n}
k_followers tweet = {{t_{11}...t_{1n}}, {t_{21},…., t_{2n}},…, {t_{k1}...t_{kn}}
k_following tweets = {{t_{11}...t_{1n}}, {t_{21},…,t_{2n}},…,{t_{k1}...t_{kn}}, where t ={text, y}y ϵ{0..4}
/* The tweets of users, followers and followings are the algorithms' input */
Result: user_Category ϵ {0...4}
for each *tweet* in user_tweets
{usertweetLabelList.append(LSTMclassify(tweet))
}
for k = 1 to numberOfFollowers {
for each tweet in following_userk_tweet
{userkflwgLabelList.append(LSTMclassify(tweet))
}
for each tweet in follower_userk_tweet
{userkflwrLabelList.append(LSTMclassify(tweet))
}
followerTweetLabels.append(userkflwrLabelList)
followingTweetLabels.append(userkflwgLabelList)
}
userPolarity = ComputePolarity(usertweetLabelList) followerPolarity = ComputePolarity(followerTweetLabels)
followingPolarity = ComputePolarity (followingTweetLabels)
userCategory = userCategoryRules (userPolarity, followerPolarity, followingPolarity) return userCategory

Figure 1. User Profiling Algorithm

After obtaining the labels of the n-tweets of the target user, his/her k-followers and k-following, the algorithm then calculates three separate values namely user polarity, following polarity and follower polarity by making use of Algorithm 2 shown in Figure 2. The algorithm returns the polarity for a set of given input tweet labels by computing the frequently occurring labels in the set of given labels. This can be taken as the mode of the set of given tweet labels.

The label that is returned as the mode represents the polarity of the tweets for a given user. It also indicates that the majority of the tweets that the entity (user, following,

Algorithm 2: Compute Polarity

Data: tweet_labels= {{l11, l12,…..,l1n}, {l21, l22,…..,l2n},…, {ln1, ln2,…..lnn}}where
l ={y}y □{0..4}
/* The labels assigned to the tweets of each user is taken as the algorithm's input */ Result: polarity □ {0...4}
for each label in tweet_labels
{polarity. append (Mode(labels))
}

return polarity

Figure 2. Algorithm for Computing Polarity

followers) posts belong to that category. Finally, the user profiling algorithm then determines the final category to place the target user by invoking another algorithm with the three polarities (i.e. target user polarity, follower polarity, and following polarity). The algorithm for determining the definitive profile for the target user is shown in Figure 3 as Algorithm 3.

The algorithm compares the input polarities given to it

Algorithm 3: user Category Rules

```
Data: userPolarity = {p1, p2,.....,pn}
      followerPolarity={p11, p12,.....,p1n}
      followingPolarity={p11, p12,.....,p1n}where p ={y}y ∈{0..4}
/* User polarity, follower polarity, and following polarity are taken as the
algorithms input */
Result: user_Category ∈ {0...4}

If userPolarity and followingPolarity = followerPolarity
      {User_Category = followerPolarity
      }
elseif userPolarity and followerPolarity = followingPolarity
      {User_Category = followingPolarity
      }
elseif followingPolarity and followerPolarity = userPolarity
      { User_Category = userPolarity
      }
else {
      User_Category = userPolarity
      }
return user_Category
```

Figure 3. Algorithm for Comparing Polarities

based on the if-then-rules we defined (i.e. if user polarity = following polarity or follower polarity then user category = user polarity). The assumption of the if-then-rule for determining the final profile or category of the target-user given the polarities of his followers and followings is that, the influence of the followers on the target user and that of its following user has a little impact compared to the willingness of the user itself. However, a user polarity is consolidated and validated when his follower and following polarities agreed with the user polarity.

2.5 Justification for Using Deep Learning for Tweet Classification

Finding the best classifier for the abusive language detection problem involved using our dataset on different machine learning algorithms combined with approaches in the existing works together with our approach and carrying out performance evaluation. The results shown in Tablels how that the LSTM-based and the GRU-based network combined with word embeddings obtained the highest classification accuracy followed by SVM with the Bag of Words Approach. Based on this result, these two deep learning algorithms were further evaluated with two different methods of using word embeddings. The results of this evaluation are shown in Table 2.

Existing Work	Algorithm	Features	Precision	Recall	F1-Score	Accuracy (%)
Albadi et al., 2018	Logistic Regression	Ngram	69	68	68	68.00
	SVM		69	69	69	69.00
	GRU	Word Embeddings	69	67	67	87.31
Sureka & Agarwal, 2016	Naïve Bayes	Bag of Words	69	66	67	66.00
	SVM		70	70	70	70.00
	Decision Tree		67	65	65	65.00
Alfina et al., 2017	Naïve Bayes	Bag of Words	69	66	67	66.00
	SVM		70	70	70	70.00
	Logistic Regression	Ngrams	69	69	69	69.00
	Random Forest		67	68	67	68.00
Fatahillah et al., 2017	Naïve Bayes	Bag of Words	69	66	67	66.00
Watanabe et al., 2018	SVM	Unigrams	69	69	69	69.00
	Random Forest		67	67	67	67.00
	SVM	Bag of Words	70	70	70	70.00
	Random Forest		67	68	67	68.00
Our Approach	LSTM	Word Embeddings	70	70	69	88.16
	Naïve Bayes	Bag of Words	74	62	64	62.00
	SVM		70	71	70	70.00
	Random Forest		67	67	67	67.00
	Logistic Regression		69	68	68	68.00

Table 1. Justification for using Deep Learning Model

Deep Learning Models	Metrics				
	Accuracy (%)		Precision	Recall	F1-Score
	Train	Validation			
LSTM_E	99.18	88.62	72	71	71
GRU_E	99.87	87.51	69	69	69
LSTM_w2v	97.57	88.84	72	71	72
GRU_w2v	99.80	86.21	69	67	67

Table 2. Comparison of Two Recurrent Nueral Network Methods

The results in Table 2, shows the results of the comparison between the performance of the two deep learning algorithms evaluated in this paper. Both models were configured with the same settings with the hyper parameter of both algorithms set to the same values. The embedding dimension was set to 250 with two LSTM/GRU layers made of 250 memory units with a dropout value of 0.9 for both layers, the last layer is adense layer of 5 units with sigmoid activation function and an epoch value of 15. The results show that the two algorithms obtained high scores that are comparable. However, the LSTM-based deep neural networks obtained higher accuracy and precision than the GRU-based deep neural networks with both networks trained using two different embeddings method.

The networks in which the embedding layer was used to learn word embeddings are (LSTM_E & GRU_E) while the networks trained using Word2Vec learned embeddings are (LSTM_w2v & GRU_w2v). Using these results, LSTM-based deep neural network was chosen as the abusive language detection model in this paper.

2.6 Design of Long Short-Term Memory (LSTM) Recurrent Neural Network Architecture for Tweet Classification

The user profiling algorithm presented in section 3.4 relies on the LSTM recurrent neural network model to determine the labels of the tweets. This section presents the design of LSTM architecture that was employed for the task of tweet classification which was embedded into the profiling algorithm. The LSTM was implemented using the Keras deep learning library (Keras, 2018) with Theano backend Rami et al. (2016) in python programming language and word embedding as feature vectors. Keras is a deep learning library and TensorFlow (TensorFlow, 2018) wrapper that makes the implementation of deep learning algorithms in a few lines of code easy. The deep learning models implemented in this paper belong to the many-to-one architecture in which the models have feature vector sequences as inputs and predict one output.

Tweets were converted into word embeddings by assigning integer indexes to unique words in the dataset. Entire index sequences were then padded with zeros so that all sequences havethesamelengthof36(thelongest tweet length is 34 words). The sequences were used as inputs in the embedding layer that maps word indexes to pre-trained word embeddings. The embedding model used is 250 in dimension and contains more than 28, 391 feature vectors trained on approximately 20, 217 English tweets. The embedding layer produced a feature vector with a dimension of (36,250), which served as the input to an LSTM layer with 20 hidden units with a dropout rate of 0.9. Dropout is a technique for regularization used to prevent over fitting of the model. The LSTM layer enables long-distance semantic and contextual information to be captured. The output layer of our model made use of a sigmoid activation function and adam as the optimizer. The output of this layer is a vector that has a dimension of (None, 5). For every tweet, this layer predicts the probability of the class to which the tweet belongs (from zero to one). Training was conducted in size 50 batches. An illustration of this network architecture is shown in Figure 4.

However, to obtain the best model for the task of tweet classification different settings of the hyper parameters of the LSTM network was tested to find the optimal settings for our abusive language detection model. The hyper-parameters adjusted are; embedding dimension, Number of LSTM units, dropout, activation function, batch size and epoch.

The dataset was divided into three different sets for the

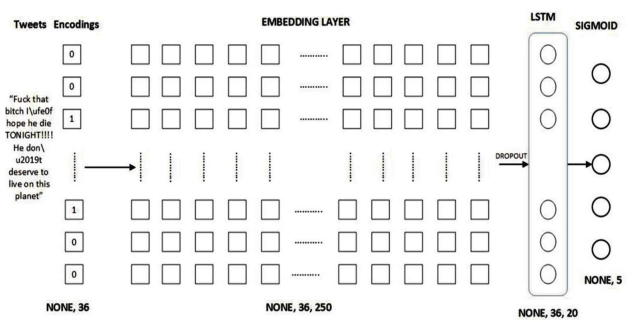

Figure 4. LSTM Network Architecture

tweet classification experiment. The three sets are:

2.6.1 Training Set

this set contained 80% of the entire dataset (9443 tweets) which is evenly distributed across all five classes (bigot, offensive, racist, extremist & neutral): with each class having 1887 tweets. This is the set that will be used for learning. It will be used to fit the parameters of our classifier.

2.6.2 Validation Set

this set contains 10% of the dataset (1181 tweets) which is evenly distributed all five classes: with each class having 236 tweets. This is the set that will be used to tune the hyper parameters of our classifiers.

2.6.3 Test Set

this set also contains 10% of the dataset (1181 tweets) which is evenly distributed across all five classes also: with each class having 236 tweets. This is the set that will be used to evaluate the performance of our classifier.

2.7 Performance Evaluation Metrics

The use of classification accuracy alone when evaluating the performance of a classification algorithm can be misleading, especially if the dataset is unbalanced or contains more than two classes. Hence, a confusion matrix along with other metrics that also includes accuracy are usually used to evaluate the performance of a classifier.

Confusion Matrix (CM): the confusion matrix M is an N-dimensional matrix, where N is the number of classes that summarizes the classification performance of a classifier with respect to the test data [15]. Each column of the matrix represents predicted classifications and each row represents actual defined classifications as shown in Figure 5.

Accuracy: is the most commonly used performance measure which measures the proportion of all predictions that are correct. Accuracy is obtained by dividing the values in the diagonal with the total sum of the confusion matrix. It can be formalized as follows

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Actual \ Predicted	Positive	Negative
Positive	TP	FN
Negative	FP	TN

Figure 5. The Confusion Matrix M of a 2-Class Problem with Relationship between Actual Samples and Predicted Samples

Precision: is calculated class-wise and is a measure of how many predictions of a class were predicted correctly. Precision of class c can be formalized as:

$$\Pr ecision_c = \frac{TP}{TP + FP}$$

Recall: just as precision is calculated class-wise and is a measurement of how many instances of a class was predicted correctly. Recall of class c can be formalized as:

$$\mathrm{Re}\, call_c = \frac{TP}{TP + FN}$$

F1-score: is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1-score is usually more useful than accuracy, especially if you have an uneven class distribution. It can be formalized as follows:

$$F1 - Score = 2X \frac{(\mathrm{Re}\, call * \Pr ecision)}{(\mathrm{Re}\, call + \Pr ecision)}$$

3 Results

3.1 Results For LSTM Network Architecture Design

Table 3 presents the results of the hyper parameter tuning carried out to identify the best performing deep LSTM-based neural network for the task of abusive language detection on the validation and test dataset. How well each model performed with respect to precision, recall, f1-score and accuracy is visualized in the Figure 6 below. The embedding dimension value, the dense layer unit and the batch size was set to a constant value of 250, 5 and 50 respectively and the rest of the hyper parameter settings for all the models evaluated are presented in Table 4.

Models	Precision	Recall	F1-Score	Accuracy %		
				Train	Validation	Test Accuracy
LSTM_1	70	70	70	99.01	87.81	88.47
LSTM_2	72	71	71	99.18	88.62	88.70
LSTM_3	70	69	69	96.48	87.26	87.84
LSTM_4	69	68	69	96.17	86.81	87.82
LSTM_5	71	70	70	99.32	65.66	69.77
LSTM_6	70	70	70	99.77	86.77	88.13
LSTM_7	71	70	70	99.50	67.45	70.19
LSTM_8	69	69	69	90.66	66.13	69.01
LSTM_9	71	70	70	99.07	66.70	69.94
LSTM_10	72	70	70	99.52	63.78	69.52
LSTM_11	69	69	69	99.62	66.89	68.84
LSTM_12	74	73	73	99.01	87.58	89.14
LSTM_13	69	69	69	99.57	67.17	68.76
LSTM_14	71	70	70	99.38	66.60	70.19

Table 3. Results on the Validation set of all the Tweet Classification Models during Hyper Parameter Tuning

Based on the results of all the 14 abusive language detection models evaluated as shown in Table III and as visualized in Figure 6, LSTM_12 was identified to be the best model. It obtained an accuracy of 87.58% and 89.14% on the validation and test dataset respectively with a precision, a recall and an f1-score of 74, 73 and 73 respectively. LSTM_8 was identified to be the worst model. It obtained an accuracy of 66.13% and 69.01% on the validation and test dataset respectively with a precision, are call and an f1-score of 69.

3.2 Results for Evaluation of user Profiling Algorithm

The result for 20 users which the user profiling algorithm obtained when used on a dataset of 52 users and 16,582 tweets is shown in Figure 7. The results of the algorithm as shown in the above table, outputs the user names of each user in our user profiling data set together with the user polarity, following polarity, follower polarity and the classification category of the users.

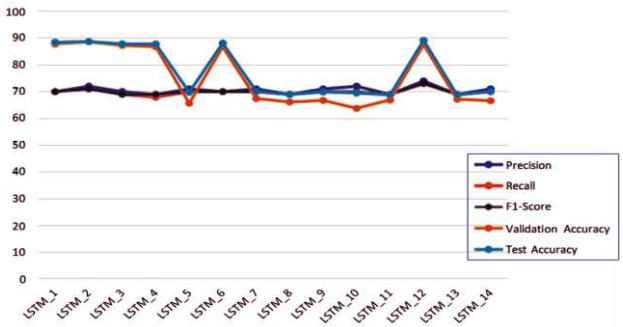

Figure 6. Precision, Recall, F1-Score and Accuracy Graph of all the Models Evaluated

	A	B	C	D	E
1	User	User Polarity	Following Polarity	Followers Polarity	User Category
2	user1	4	4	3	4
3	user10	4	3	4	4
4	user11	3	0	0	3
5	user12	3	0	0	3
6	user13	3	0	3	3
7	user14	3	3	3	3
8	user15	4	0	0	0
9	user16	1	3	3	1
10	user17	4	4	0	4
11	user18	4	0	0	0
12	user19	3	0	0	3
13	user2	4	4	4	4
14	user20	1	0	0	1
15	user21	3	0	0	3
16	user22	3	0	0	3
17	user23	3	0	0	3
18	user24	3	0	0	3
19	user25	3	3	0	3
20	user26	3	1	1	3

Figure 7. Result Obtained by the user Profiling Algorithm

To evaluate our user profiling algorithm, we created a test set by collecting the tweets of users who are known to have views that belong to our interest categories. Tweets of right-wing leaning politicians known to use offensive language or express extremist views. Tweets of accounts

Models	Hyperparameters					
	Embedding Dimension	LSTM Units	Embeddings	Activation	Epochs	Dropout
LSTM_1	250	500	Word2Vec	Sigmoid	15	0.9
LSTM_2	250	500	Embedding Layer	Sigmoid	15	0.9
LSTM_3	250	100, 100	Embedding Layer	Sigmoid	3	0.9
LSTM_4	250	100, 100	Word2Vec	Sigmoid	15	0.9
LSTM_5	250	100, 100	Embedding Layer	Softmax	2	NIL
LSTM_6	250	500	Embedding Layer	Sigmoid	15	NIL
LSTM_7	250	500	Embedding Layer	Softmax	15	NIL
LSTM_8	250	100, 100	Embedding Layer	Softmax	15	0.9
LSTM_9	250	100, 100	Word2Vec	Softmax	15	0.9
LSTM_10	250	100, 100	Word2Vec	Softmax	15	NIL
LSTM_11	250	500	Word2Vec	Softmax	15	NIL
LSTM_12	250	20	Word2Vec	Sigmoid	20	0.9
LSTM_13	250	20	Embedding Layer	Softmax	15	0.9
LSTM_14	250	20, 20	Word2Vec	Softmax	15	0.9

Table 4. Hyper Parameter Settings of all the Evaluated LSTM Models

that use offensive language and accounts that belongs to separatist groups are also collected. A total of 37, 532 tweets were collected for 18 users. After collecting the tweets of these users, the user profiling algorithm which determines the type of abusive user was applied to the newly collected dataset. The results obtained by the algorithm are shown in Figure 8.

3.3 Discussion of the Results

From the results obtained, the existing approach with the dataset created in this thesis which used traditional machine learning algorithms such as Naïve Bayes, SVM, Logistic Regression and Random Forest with Bag of Words and N-grams as feature vectors performed poorly. The results show that when bag of words was used as features, SVM, Random Forest and Logistic Regression obtained the highest accuracies on the test dataset. They obtained 70%, 68% and 69% on the test dataset respectively. Naïve Bayes algorithm obtained the lowest accuracy on the test dataset. It obtained 66% on the test dataset respectively. The use of N-gram based features did not improve the performances of these algorithms. In fact, it had the opposite or similar effect on the performance of the algorithms. The existing work which used a GRU-based deep neural network with pretrained word embedding performed significantly better than the traditional machine learning algorithms. Based on this outcome, two deep learning algorithms that are ideal for handling sequential data such as texts were chosen and further evaluated. The results of the evaluation revealed that

	A	B	C	D	E
1	User	User Polarity	Following Polarity	Followers Polarity	User Category
2	user1	3	0	4	3
3	user2	3	1	4	3
4	user3	4	4	3	4
5	user4	3	4	3	3
6	user5	3	3	3	3
7	user6	0	4	0	0
8	user7	4	4	4	4
9	user8	4	4	4	4
10	user9	1	3	3	3
11	user10	4	1	4	4
12	user11	3	3	0	3
13	user12	0	0	1	0
14	user13	1	1	1	1
15	user14	1	4	1	1
16	user15	1	4	4	1
17	user16	3	4	4	3
18	user17	3	3	4	3
19	user18	3	3	4	3

Figure 8. Results of user Profiling Algorithm on our Test Users

LSTM-based neural network performed slightly better than the GRU-based neural network. The LSTM-based neural network trained with pretrained word embeddings and word embeddings learned using the embedding layer obtained accuracies of 88.84% and 88.62% on the validation dataset respectively while the GRU-based neural network trained with pretrained word embeddings and word embeddings learned using the embedding layer obtained accuracies of 86.21% and 87.51% respectively on the validation dataset. Consequently, LSTM-based neural network was chosen to identify different categories of abusive languages.

The LSTM-based neural network was further evaluated with the tuning of hyper parameters to determine the best abusive language detection model. The adjustment of hyper parameters such as the values of Embedding dimension, LSTM units, types of Activation function and dropout helped with the identification of an abusive language detection model LSTM_12 that performed better than the 14 models evaluated in this paper. The accuracy graph of the model shows that the accuracy of the model improved significantly from when training started to the end of training as compared with the other models. The model achieved aprecision of 74, are call of 73, aF1-Score of 73 and a validation and test accuracy of 87.58% and 89.14% respectively. The confusion matrix of the model showed that it classified 78% bigotry correctly and 11% as extremism related, 69% of offensive tweets were correctly classified with 12% classified as neutral and extremism related, it classified 84% of racist tweets correctly with 7% classified as offensive, it correctly classified 62% extremism related tweets and classified 23% as offensive, and it classified 70% neural tweets correctly and 14% as offensive. Although, results show that LSTM_2 obtained high scores in four out of five metrics. It obtained an accuracy of 88.62% on the validation set with a precision of 72, recall of 71 and anf1-score of 71. However, going by its confusion matrix, it did not obtain a high enough classification accuracy across all categories as compared with other models with high accuracies in all five metrics. LSTM_8 was identified to be the worst performing model. The validation and testing

accuracy obtained by the model was 66.13% and 69.01%, respectively. It achieved a precision of 69, are call of 69 and an F1 score of 69. From its confusion matrix, it classified 80% of bigotry tweets, 47% of offensive tweets, 84% of racist tweets, 62% of extremism related tweets and 71% of neutral tweets correctly. By analysing the values of all five metrics, LSTM_12 was chosen as the model with the best performance since it obtained high classification accuracy across all five categories.

The identified best performing abusive language detection model was then applied to the user profiling dataset by the user profiling algorithm so as to classify the tweets of users in our interest categories into different abusive language categories. The new user profiling dataset which contains features such as user names, user tweets and the labels assigned to the tweets was fed into the user profiling algorithm which calculates values such as user polarity, following polarity and follower polarity and compares these three values in order to determine the type of abusive language category each user belongs to. In order to evaluate the user profiling algorithm, a new dataset of users known to belong to our interest categories was collected. The Twitter feed of far-right leaning politicians, celebrities, and political groups, together with the tweets of extremist and separatist groups known to use abusive language and violence are all collected. A total of 37, 532 tweets were collected for 18 users. After collecting the tweets of these users, the user profiling algorithm was used to classify the tweets and calculates the user polarity, following polarity, and follower polarity which it uses to determine the type of abusive user our test users are. Result obtained by the user profiling algorithm shows that the algorithm made predictions that are inline with our initial categorization for 15 out of18 users which indicates 83.33% accuracy.

4. Limitations of the Study

The focus of this paper is on the development of solutions that can identify abusive messages and users on Twitter. The messages made use of are the ones written in English only, non-English messages are discarded. Also, the images, videos, emoticons contained in such messages are discarded and not made use of also. The pre-processing step of performing spelling correction on tweets is not performed in this study. But non-correction of spelling does not affect the results presented in the study.

Conclusion

In this research, we have provided an approach for automatically detecting different types of inappropriate tweets together with the users who post such tweets, to contribute to knowledge in machine learning and data mining fields.

The contributions of this paper are summarised as follows:

- We have proposed a novel approach based on a user profiling algorithm that uses a deep Long Short-Term Memory (LSTM) based neural network trained to detect abusive language.
- The identification of a deep learning architecture for abusive language detection that utilizes features derived from messages posted by users online.
- The experimental evaluation of the abusive language detection model on Twitter dataset which demonstrates the top performance achieved on the classification task.

Future Work

Future extensions of this work include: the implementation of a real time application for the continuous profiling of users. Also, the text contained in tweets was used to train all the abusive language detection models in this paper. Therefore, the multi modal analysis of tweets that includes emoticons, images, and videos which will makeit possible for this approach to be applied to other social media platforms like YouTube and Facebook is also an important direction to explore in the future.

References

[1]. Abubakar, U., Bashir, S. A., Abdullahi, M. B., & Adebayo, O. S. (2019). Comparative Study of Various Machine Learning Algorithms for Tweet Classification. *i-manager's Journal on Computer Science,* 6(4), 12-24.

[2]. Agarwal, S., & Sureka, A. (2016). But I Did Not Mean It! Intent Classification of Racist Posts on Tumblr. *European Intelligence and Security Infromatics Conference* (pp.

124-127). IEEE. doi:10. 17632/hd3b6v659v.2

[3]. Albadi, N., Kurdi, M., & Mishra, S. (2018). Are They Our Brother? Analysis and Detection Of Religious Hate Speech in the Arabic Twittersphere. *International Conference on Advances in Social Networks Analysis and Mining* (pp. 69-76). IEEE.doi:10.1109/ASONAM.2018.8508247

[4]. Alfina, I., Mulia, R., Fanany, M., & Ekanata, Y. (2017). Hate Speech Detection in the Indonesian Language: A Dataset and Preliminary Study. *International Conference on Advanced Computer Science and Information Systems (ICACSIS)* (pp. 233-238). IEEE. doi:10.1109/ICACSIS.2017.8355039

[5]. Al-Quirishi, M., Aldrees, R., AlRubaian, M., Al-Rakhami, M., Rahman, M. S., & Alamri, A. (2015). A New Model for Classifying Social Media Users According to Their Behaviors. *World Symposium on Web Applications and Networking (WSWAN)* (pp. 1-5). IEEE. doi:10.1109/WSWAN.2015.7209085

[6]. Al-Rfou, R., Alain, G., Almahairi, A., Angermueller, C., Bahdanau, D., Ballas, N.,.....& Zhang, Y., (2016). Theano: A Python framework for fast computation of mathematical expressions. doi:arXiv:1605.02688v1 [cs.SC]

[7]. Aphinyanaphongs, Y., Ray, B., Statnikov, A., & Krebs, P. (2014, August). Text classification for automatic detection of alcohol use-related tweets: A feasibility study. In *Proceedings of the 2014 IEEE 15th International Conference on Information Reuse and Integration (IEEE IRI 2014)* (pp. 93-97). IEEE. doi:10.1109/IRI.2014.7051877

[8]. Chikashi, N., Joel, T., Achint, T., Yashar, M., & Yi, C. (2016). Abusive Language Detection in Online User Content. *International World Wide Web Conference (pp. 145-153). Montréal, Québec, Canada: Association of Computing Machinery (ACM)*. doi:dx.doi.org/10.1145/2872427.2883062

[9]. Cufoglu, A. (2014). User Profiling - A Short Review. *International Journal of Computer Applications,* 108(3), 1-9. doi:10.5120/18888-0179

[10]. Dey, R., & Salemt, F. M. (2017, August). Gate-variants of gated recurrent unit (GRU) neural networks. In *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)* (pp. 1597-1600). IEEE. doi:10.1109/MWSCAS.2017.8053243

[11]. Djuric, N., Zhou, J., Morris, R., Grbovic, M., Radosavljevic, V., & Bhamidipati, N. (2015, May). Hate speech detection with comment embeddings. In *Proceedings of the 24th international conference on world wide web* (pp. 29-30). ACM. doi:dx.doi.org/10.1145/2740908.2742760.

[12]. DL4J (2017, August 12). Deep Learning and Neural Network Glossary. Retrieved March 23, 2018, from https://deeplearning4j.org/cn/glossary

[13]. Fatahillah, N., Suryati, P., & Haryawan, P. (2017). Implementation of Naive Bayes Classifier Algorithm on Social Media (Twitter) to the Teaching of Indonesian Hate Speech. *International Conference on Sustainable Information Engineering and Technology* (pp. 128-131). IEEE. doi:10.1109/SIET.2017.8304122

[14]. Ikeda, K., Hattori, G., Ono, C., Asoh, H., & Higashino, T. (2013). Twitter user profiling based on text and community mining for market analysis. Knowledge-Based Systems, 51, 35-47. doi:10.1016/j.knosys.2013.06.020 In *2016 International Conference on Big Data and Smart Computing (BigComp)* (pp. 231-238). IEEE. doi:10.1109/BIGCOMP.2016.7425918

[15]. Iqbal, M. (2019, February 27). Twitter Revenue and Usage Statistics. Business of Apps, Retrieved March 02, 2019, from http://www.businessofapps.com

[16]. Kang, K., Yoon, C., & Kim, E. Y. (2016, January). Identifying Depressive Users in Twitter Using Multimodal Analysis.

[17]. Keras. (n.d.). The Python Deep Learning Library. Retrieved July 10, 2018, from https://keras.io/

[18]. Lee, W. J., Oh, K. J., Lim, C. G., & Choi, H. J. (2014, February). User profile extraction from Twitter for personalized news recommendation. In *16th International conference on advanced communication technology* (pp. 779-783). IEEE. doi:10.1109/ICACT.2014.6779068

[19]. 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.

[20]. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Represenations in Vector Space. CoRR, abs/1301.3781. doi:https://arxiv.org/abs/1301.3781

[21]. Neethu, M. S., & Rajasree, R. (2013, July). Sentiment Analysis in Twitter Using Machine Learning Techniques. *International Conference on Computing, Communications and Networking Technologies (ICCCNT)* (pp. 1-5). IEEE. doi:10.1109/ICCCNT.2013.6726818

[22]. 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. doi:10.1016/j.invent.2015.03.005

[23]. 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).

[24]. Pitsilis, K. G., Ramampiaro, H., & Langseth, H. (2018). Detecting Offensive Langauge in Tweets Using Deep Learning. New York: Cornell University. doi:arXiv:1801.04433v1 [cs.CL]

[25]. Rehurek, R. (2019, April 10). Topic modelling for humans. Retrieved July 10, 2018, from Gensim: https://radimrehurek.com/gensim/

[26]. Rocha, E., Francisco, P. A., Calado, P., & Sofia-Pinto, H. (2011). User Profiling on Twitter. Semantic Web Journal. Retrieved May 12, 2018, from http://www.semantic-web-journal.net

[27]. Sureka, A., & Agarwal, S. (2014, September). Learning to classify hate and extremism promoting tweets. In *2014 IEEE Joint Intelligence and Security Informatics Conference* (pp. 320-320). IEEE. doi:10.1109/JISIC.2014.65

[28]. TensorFlow. (n.d.). An end-to-end open source machine learning platform. Retrieved July 10, 2018, from https://www.tensorflow.org/

[29]. Twitter. (n.d.). Filter Realtime Tweets. Retrieved January 20, 2018, from Twitter Developer: https://developer.twitter.com/en/docs/tweets/filter-realtime/api-reference/post-statuses-filter.html

[30]. 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. doi:10.1109/ACCESS.2018.2806394

[31]. Wikarsa, L., & Thahir, S. N. (2015, November). A text mining application of emotion classifications of Twitter's users using Naive Bayes method. In *2015 1st International Conference on Wireless and Telematics (ICWT)* (pp. 1-6). IEEE. doi:10.1109/ICWT.2015.7449218

[32]. 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. doi:10.1016/j.trc.2017.11.027

ABOUT THE AUTHORS

Umar Abubakar received the B.Tech degree in Computer Science from the University of Technology Minna, Nigeria, in 2014 and is currently pursuing M.Tech Degree in Computer Science from the same Institution. Currently, he majors in Computational Intelligence, Machine Learning and Data Mining. He works as a Program Analyst in the Information and Communication Department of Raw Materials Research and Development Council (RMRDC), Maitama, Abuja, Nigeria from March 2018 till Date. 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 where he teaches and supervises both undergraduate and graduate students in Computer Science. He obtained Bachelor (B.Tech) from Ladoke Akintola University of Technology (LAUTECH) Ogbomoso, Nigeria, in 2003, and Master (MSc.) degrees in Computer Science from University of Ibadan, Nigeria, in 2008. He was awarded PhD in Computing by Robert Gordon University Aberdeen UK in 2017 for his thesis titled: Change Detection for Activity Recognition. The PhD was funded by the National Information Technology Development PhD Scholarship award. He has published several research papers in peer-reviewed journals and conferences in varying aspects of machine learning applications. He is a member of the ACM and Nigeria Computer Society. His research interests include Application of Machine Learning to Activity Recognition, Social Media Mining and Intelligent Healthcare Systems.

Dr. Laud Charles Ochei holds a PhD in Computing from Robert Gordon University, Aberdeen, UK. He has a broad range of research and software development experience in various academic and industry collaborations. His research interests are in software engineering, distributed systems, internet of things, and cloud application architectures. He has published several research papers in peer-reviewed International Conferences and Journals.

Dr. (Engr.) Ibrahim Adepoju ADEYANJU is currently a Senior Lecturer at the Department of Computer Engineering and Deputy Dean in the Faculty of Engineering, Federal University, Oye-Ekiti, Nigeria. He obtained a Bachelor degree (B.Tech.) in Computer Engineering (2004) from Ladoke Akintola University of Technology (LAUTECH), Ogbomoso, Oyo state, Nigeria and his Masters (MSc.) degree in Computing Information Engineering (2007) and Doctorate (PhD) degree in Computing (2011) from the Robert Gordon University, Aberdeen, United Kingdom. He was a Lecturer at LAUTECH from 2006 to 2015 where he taught several undergraduate and postgraduate courses as well as supervised many undergraduate projects and postgraduate theses. He is a recipient of several local and international awards such as the Stephen Awokoya Scholarship for Science Education, Petroleum Technology Development Fund (PTDF), UK Scottish Overseas Research Students Award Scheme (ORSAS), and the Massachusetts Institute of Technology Empowering the Teachers (MIT-ETT) fellowship among others. He is a registered Computer Engineer with the Council for the Regulation of Engineers in Nigeria (COREN) and belongs to several other professional bodies including Institute of Electronic and Electrical Engineers (IEEE), and the Nigerian Young Academy (NYA). Dr. Adeyanju's research interest is in Intelligent Systems and Microprocessor Control/ Embedded Systems, and currently has several peer reviewed journal articles and conference papers across different research areas of Computer Science and Engineering including Microprocessor based systems, Pattern Recognition, Distributed Constraints Satisfaction, Artificial Intelligence, Information Retrieval, Information Systems and Machine Learning.