Hindawi
Complexity
Volume 2020, Article ID 5639787, 14 pages
https://doi.org/10.1155/2020/5639787

Research Article

Towards a Framework for Acquisition and Analysis of Speeches to Identify Suspicious Contents through Machine Learning

Md. Rashadur Rahman,[1] Mohammad Shamsul Arefin,[1] Md. Billal Hossain,[1] Mohammad Ashfak Habib,[1] and A. S. M. Kayes[2]

[1]*Department of Computer Science & Engineering, Chittagong University of Engineering & Technology, Chattogram, Bangladesh*
[2]*Department of Computer Science & Information Technology, La Trobe University, Melbourne, Australia*

Correspondence should be addressed to A. S. M. Kayes; a.kayes@latrobe.edu.au

Received 13 July 2020; Revised 19 September 2020; Accepted 29 October 2020; Published 16 November 2020

Academic Editor: Abd E.I.-Baset Hassanien

The most prominent form of human communication and interaction is speech. It plays an indispensable role for expressing emotions, motivating, guiding, and cheering. An ill-intentioned speech can mislead people, societies, and even a nation. A misguided speech can trigger social controversy and can result in violent activities. Every day, there are a lot of speeches being delivered around the world, which are quite impractical to inspect manually. In order to prevent any vicious action resulting from any misguided speech, the development of an automatic system that can efficiently detect suspicious speech has become imperative. In this study, we have presented a framework for acquisition of speech along with the location of the speaker, converting the speeches into texts and, finally, we have proposed a system based on long short-term memory (LSTM) which is a variant of recurrent neural network (RNN) to classify speeches into suspicious and nonsuspicious. We have considered speeches of Bangla language and developed our own dataset that contains about 5000 suspicious and nonsuspicious samples for training and validating our model. A comparative analysis of accuracy among other machine learning algorithms such as logistic regression, SVM, KNN, Naive Bayes, and decision tree is performed in order to evaluate the effectiveness of the system. The experimental results show that our proposed deep learning-based model provides the highest accuracy compared to other algorithms.

1. Introduction

Speech has been the mostly used medium for conveying information among people all over the world since the dawn of civilization. Speech is the most effective method of addressing and communicating with the audience in order to deliver some message. Speech empowers an individual to reach a large number of people directly. It is a very dynamic way to shift a huge number of people's mindset or to reinforce their confidence in speaker [1]. Historically, it played a significant role in persuading the audience into a specific agenda [2, 3].

Speech can be used to influence people in both righteous and wrong ways. Speech has been used to escalate hatred among the communities [4–6]. Misguided speech leads people in the wrong direction, which raises social risk. The preservation of the right to freedom of speech is an integral aspect of modern democratic states [7]. People are freely expressing their emotion, thought, anger, and grudge through speeches. Often this freedom of expression is misused by certain people in society, which causes social controversies [8, 9]. The problem is more critical when the religious people give deceptive speeches. This is because, in general, people have love for their religions and in most of the cases they respect and rely on the speeches of their religious speakers [10]. Wrong speeches can contribute to crime and pose a threat to the government. Since Internet technology is growing rapidly, any misleading speech can be easily spread among different groups of people through various social media platforms [11]. Statistics show an alarming rate of growth of hate crime over the years[2]. It not only threatens a country's people's lives and livelihood but also undermines

worldwide peacekeeping. These types of threats are troubling in both national and international security and steps are being taken to avoid these kinds of crimes.

To prevent any potential crime resulting from speech, suspicious speeches must be identified in the shortest time possible. There are a lot of speeches being delivered daily, which are very difficult for manual inspection. Therefore, a good speech repository and speech monitoring system can be very useful in spreading good speeches and restraining suspicious speeches from spreading. If the suspicious speeches can be classified beforehand, it will be very handy for law-enforcement agencies to take proactive steps to deter any unwanted incidents. Moreover, if the speech audio can be converted into text, then the speech can be automatically interpreted to extract information from the converted text. Analyzing the speech from direct voice data is quite difficult due to noises in speech recording and varieties of speeches. Audio data analysis is a very complex task, which includes multidimensional analysis as a simple audio can have millions of different data segments. Many attributes like frequency, pitch, volume, dialect, and so forth must be considered in the analysis of audio data. Therefore, analysis of direct audio speech is quite complex and requires much more processing time; most importantly, it requires much computational power, which may be impractical for devices with limited processing ability like smartphones. So, converting the speeches into text before analysis and then analyzing the converted text are more convenient and time-efficient.

The usage of mobile phones has increased drastically over the last few years. It is approximately 3.5 times larger than PCs [12]. Nowadays, mobile phones not only are being used as a tool for making calls and writing SMSs but also act as a means for personal entertainment and communication with the world [13]. Almost every feature that is available in a PC can also be found in a smartphone. Smartphones are available to almost everyone and one of the most popular operating systems being used in those devices is Android, developed by Google [14].

Classification of speech refers to the task of classifying a speech into a set of predefined classes. Classifying speeches into suspicious and nonsuspicious is very necessary for reducing virtual social harassment, predicting criminal activities, social clashes, and riots, and ensuring overall national security. Such a system has not yet been built to detect suspicious Bangla speech. Here we proposed a framework for acquisition of speeches and a deep learning method which is based on LSTM to detect suspicious speeches. To the best of our knowledge, this is the first work to detect suspicious Bangla speech. The contributions of our work are summarized as follows:

(i) We develop a mobile application for the acquisition of speech efficiently along with the location of the speaker. The application stores the speeches, detects the language of the speech, and finally converts the speech into text using a speech recognition API.

(ii) We develop our own dataset for training and testing the models. The dataset contains about 5000 suspicious and nonsuspicious samples.

(iii) We propose a model based on LSTM for classifying the texts (converted from speech) into suspicious and nonsuspicious. We compare the accuracy of the model with other machine learning algorithms.

The rest of the paper is organized as follows. Section 2 includes a short overview of similar research works. The architecture of our proposed methodology is presented in Section 3. Section 4 provides a description of our dataset preparation. The implementation along with the evaluation of the system is shown in Section 5. Finally, Section 6 includes the conclusion and future research.

2. Related Works

The computational study of suspicious speech or hate speech detection, from computer science point of view, is in early phase. As we first convert our audio speeches into texts then classify the texts, this work falls into the domain of text classification. Machine learning has gained much more attention of the researchers in automatic detection of suspicious texts [15]. In machine learning, identification of suspicious speech is considered as classification problem. The vast majority of the studies found this as a problem of binary classification (suspicious speech versus nonsuspicious speech) [16].

Nobata et al. proposed a supervised classification method to detect abusive English comments [17]. They used their custom-build corpus for training and testing, which was developed by extracting comments on Yahoo! Finance and News. They used regression model for classification by analyzing different aspects of user comments. They divided the texts features into four classes: N-grams, Linguistic, Syntactic, and Distributional Semantics. For N-grams features, they used space included character N-grams (3 to 5 characters) and token unigrams and bigrams. Different combinations of features were used for classification for achieving notable accuracy. Vidgen and Yasseri developed an automated software tool to distinguish between strong-Islamophobic, weak-Islamophobic, and non-Islamophobic tweets [18]. They used 4000 annotated tweets as training set and used a combined feature selection model. Their SVM-based classifier obtained 77.6% accuracy. Oriola and Kotze [19] developed an English corpus of South African tweets and applied various machine learning algorithms to detect offensive speech. Their optimized SVM with character N-gram performed best with true positive rate of 0.894.

Deep learning is also applied to classify large amount of texts with notable accuracies. For classification of texts, deep learning methods enable the deep neural networks (DNN) by using their multiple stacked layers to learn abstract feature representation from input data. Most of the works in this domain use one-hot encoding based on word/characters as input features to their models [20, 21]. Some of the works combined multiple methods to classify texts. Zhang et al.

proposed a deep neural network model combining convolutional neural network (CNN) and gated recurrent unit (GRU) networks for detecting hate speech in Twitter [22]. Elastic net regularization was used along with optimized dropout and pooling layers. Word embedding was used for mapping texts into vectors. The output of the embedding layer was fed to one-dimensional convolutional layer for feature extraction. The extracted features were given to the GRU layer. They used publicly available twitter data. In [23], Risch et al. present various deep learning approaches for sentiment analysis in online platforms for detecting toxic comments. They propose fine-grained classification instead of binary classification. Salminen et al. considered four platforms, YouTube, Wikipedia, Twitter, and Reddit, and collected 197,566 comments and labeled these comments as hateful and nonhateful [24]. They experimented with several classification algorithms. They found that XGBoost performs better than others and BERT features are most impactful.

A number of researches have been carried out in the field of text classification of English texts. No significant research has yet been done in the Bangla text classification. For sentiment mining of Bangla text, Taher et al. proposed a system based on support vector machine (SVM) [25]. In their work, they applied both linear and nonlinear SVM to determine whether it is positive or negative sentiment. They generated their dataset from the comments of several Bangla online news sites. For the preprocessing of the text, they only considered adverb, adjective, selected nouns, and verbs. They reduced the verbs into their base forms to reduce the number of vectors. For vectorization, they applied N-gram method, where $N = 1$, 2, or 3. In terms of preserving sequence of the words (syntactic and semantic content), N-gram model is better than bag-of-words model for feature representation. However, within a sentence, related words can have a high distance, which may lead to misinterpretation of the context. In [26], Chy et al. applied Naive Bayes classifier for classifying web crawled Bangla news documents. They preporcessed the text by applying stemming, removal of the less significant words called stop-words, and single-letter words. For selecting features, they used inverse document frequency (IDF) method. Dhar et al. proposed method based on Multinominal Naive Bayes (MNB) classifier to classify Bangla documents into eight predefined classes [27]. For feature extraction and selection, they applied inverse class frequency (ICF) along with the term frequency- (TF-) inverse document frequency (IDF) feature selection method named as TF-IDF-ICF scheme. A comparison with other schemes TF-IDF and TF was also shown.

Sharif et al. proposed a system based on logistic regression for classifying suspicious Bangla text [28]. As there is no available dataset on suspicious Bangla text, they developed their own private corpus of suspicious Bangla text. Their proposed model was trained with only 1500 samples and the model was tested with 500 samples. Finally, they showed that logistic regression performed better in terms of accuracy (92%) among those algorithms. They used word frequencies as the feature of the model. Bag-of-words model is used for representation of features. The main limitation of bag-of-words approach is that it only counts the frequencies of the words but the sequence of the words is ignored. So, it can contribute to misclassification if the words are used to represent different contexts [16]. In [29], Ishmam et al. developed a dataset of 5,126 comments from public Facebook groups and classified them into six classes. Their gated recurrent unit (GRU) based model achieved at most 70.10% accuracy in detecting Bangla hate language. They have not clearly defined the six classes and a smaller number of training samples in each class yields poor accuracy. The RNN based model proposed in [30] achieved 82.20% accuracy on their collected 4,700 Bangla text samples from various online platforms. Islam et al. proposed a classification method based on Multinomial Naive Bayes (MNB) for spam detection of Bangla texts [31]. They only collected 1,965 instances from social media platforms like Facebook and YouTube. The model has an accuracy of 88.44%. To detect abusive languages and threats on social media, Chakraborty et al. showed a methodology using SVM with linear kernel, which obtained 78% accuracy [32]. Their dataset contains 5,644 comments and posts from different Facebook posts.

3. Methodology

In this section, we detail the overall architecture of our proposed system. As the input of our system is speech, we first convert the speech into corresponding texts before further analysis. First, we elaborate the speech acquisition and conversion into texts. Then we explain our LSTM based classification model in detail. Our overall framework consists of four modules: (i) speech acquisition module, (ii) speech storage module, (iii) speech recognition module, and (iv) speech analysis module. The overall graphical structure of the system is shown in Figure 1.

3.1. Speech Acquisition. Acquisition of speech includes two parts: one for speech recording and another for location tracking. The architecture of the speech acquisition module is shown in Figure 2. The speaker's sound is recorded by a microphone and stored in local storage as an audio file. For tracking the location of the speaker, latitude and longitude are calculated first. Making use of latitude and longitude, we determine the speaker's actual location. When the speaker finishes his speech, it is uploaded into the cloud database where further processing takes place.

3.2. Speech Storage. After acquisition of speech, it is stored. For each incoming audio file, a row is created in the database with a unique identifier. The file name of the incoming audio file may be the same as some other audio files in the database. So, the file name is renamed as follows: Unique ID + "." + File Extension. After analyzing the audio file, the converted text file of each audio is stored along with the speech location and language.

3.3. Speech Recognition. The language of the incoming speech is identified in the recognition module and the voice is translated to text. The overall architecture of the speech

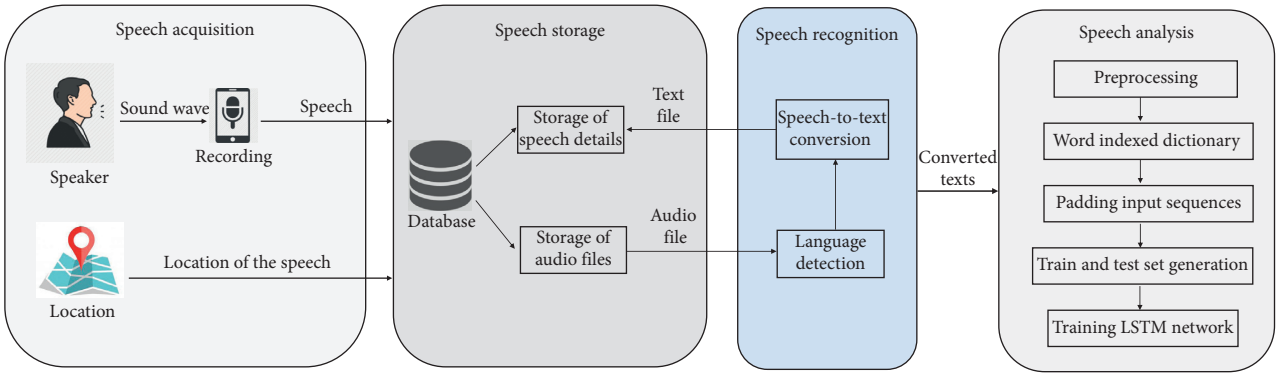


Figure 1: Overall graphical structure of the system.

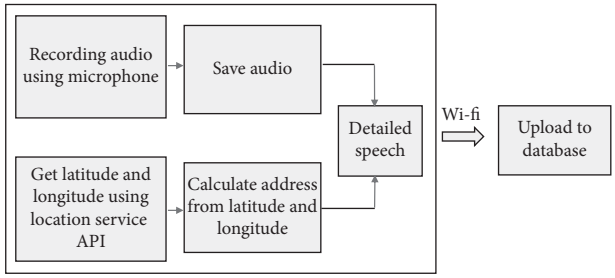


Figure 2: Architecture of speech acquisition module.

conversion process is shown in Figure 3. For converting the long-duration speech accurately, there are two possible ways. We can either split the speech into smaller chunks of constant size or split it based on the silence presented in the audio. If we split it by keeping a constant duration, then a word within the speech might get split and that word will not be detected. So, we choose to split the speech based on the silence presented. It is possible because when we speak, we pause for a small duration after finishing a sentence. That means we can consider each chunk as a sentence. The speech is split if the amplitude level in the audio is less than -16 dBFS for more than 0.5 seconds.

Real-life speeches can be in different languages. So, it is very important for your system to detect the language of the speech. Most of the time, speech contains more than one language and the system will not give good result for such type of mixed-language speeches. We need to specify the language to which we want to convert it. For example, if a speech contains Bengali language, then we need to specify language parameter as Bengali in the API. So, if we want to recognize speech containing multiple languages, we need to specify the language parameter manually. Our speech recognition module can detect three languages: Bangla, English, and Arabic. For this, we send two requests in the API for three different languages for each sentence or chunk. Then the converted text is split into words and then checked in the dictionary of that language if that word exists. If the word is found in the dictionary, then the counter for that language is increased. The language that has the maximum counter value is chosen as the language of that sentence. Algorithm 1 shows the process of detecting language and conversion of speech chunks into text.

Our system is developed to detect and convert speeches of three languages: Bangla, English, and Arabic. However, the analysis of the speech is performed only on Bangla speeches.

3.4. Speech Analysis. After converting the speeches into corresponding texts, our speech dataset has become a dataset of texts. This section describes the analysis of the converted texts from the speeches.

3.4.1. Preprocessing. Preprocessing of data is a vital part for training any machine learning model. As we train our model by text documents, the first step of preprocessing is tokenization. Tokenization is the process of splitting a text into group of streams of characters called tokens delimited by white space, new line, tab, and so on. After tokenization, each text document is a list of words (tokens). As all the words are not equally important in determining the context of the text, some words are removed to increase the accuracy of the model and to reduce the feature dimension. So, punctuation marks, English and Bangla numerals, special characters, least frequent words, and most frequent words are removed as they represent no significance to the context of the text.

3.4.2. Word Indexed Dictionary. Texts cannot be given as the direct input to the neural networks because neural networks do not accept direct text data as input. Neural networks only accept numeral inputs. As text is a sequence of words, if each word has an integer representation, then we can convert a sequence of words into a sequence of numbers, which can be fed to embedding layer. Algorithm 2 demonstrates the process of creating a word indexed dictionary, where each unique word is mapped to an integer. The algorithm starts by setting the index to 1 followed by iterating each word of all the text documents and updates the Word_to_Index dictionary. Same word can be encountered more than once. At each iteration, a word is encountered and it is checked whether or not the word is in the dictionary. If the encountered word is not in the dictionary, the current index is

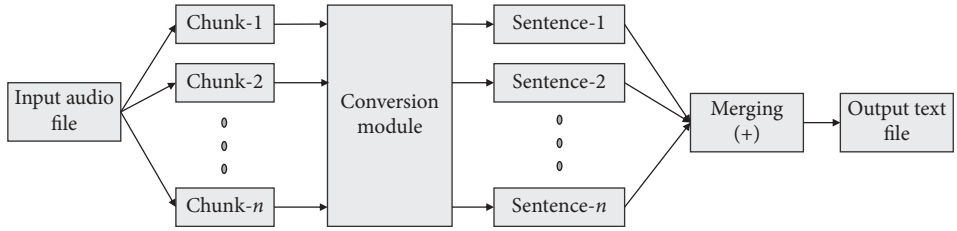


Figure 3: Framework architecture of the speech recognition module.

```
(1) Input: chunk
(2) Goal: language detection, converted text
(3) max ⟵ 0
(4) for i ⟵ 1 to 3 do
(5)     counter ⟵ 0
(6)     converted_text ⟵ recognition API (audio ⟵ chunk, language ⟵ i)
(7)     for each word in converted_text do
(8)       if word is in dictionary of language i then
(9)         counter ⟵ counter + 1
(10)      end if
(11)    end for
(12)    if counter > max then
(13)      max ⟵ counter
(14)      language ⟵ i
(15)      text ⟵ converted_text
(16)    end if
(17) end for
```

Algorithm 1: Detecting language and converting speech chunk into text.

assigned to that word and the index is increased; otherwise, the word is ignored.

Each text input is converted to a sequence of integers using the word indexed dictionary. Each word is represented by an integer.

3.4.3. Padding Input Sequence. The neural networks require inputs that have the same size and shape for both training and testing the network. Till now, we represented each piece of text data as a sequence of numbers. In our dataset, not all texts have the same number of words. So, the number sequences are of variable lengths, but the LSTM network takes input of the same length and dimension. So, we need to have the input sequences with the same size and that is why we need to pad the input sequences to the maximum length. The maximum length is set to 200. There are two types of padding: prepadding and postpadding. In prepadding, all input sequences, which are shorter than the maximum length sequence, are padded with zeros in the beginning. In case of postpadding, the sequences are padded with zeros in the ending. We used prepadding in our system as it is more suitable with LSTM [33].

3.4.4. Word Embedding. To represent the features of the words, we used word embedding technique. Word embedding is a form of word representation which allows words that are used in similar ways to have similar representations. In embedding, each individual word is represented as a real-valued vector in a predefined vector space. A word's position within the vector space is learned from text and is based on the words surrounding the word when used in the text. Each word is mapped to a real-valued vector of higher dimensions. We have used an embedding layer as the first hidden layer of our LSTM network in which word embedding is learned jointly with LSTM model. We defined our embedding layer with the size equal to the size of our vocabulary, a vector space of 200 dimensions in which words will be embedded.

3.4.5. Training and Test Set Generation. After converting the speeches into corresponding text documents, we considered the accuracy of the conversion for considering as a candidate in training or test set. The speeches which are converted with more than 90% accuracy are considered as the candidates of our training and testing set. Our training set $T = t_1, t_2, t_3, \ldots, t_n$ contains n number of text documents. Each of the text documents is labeled as either suspicious or nonsuspicious. The suspicious class is denoted as (C_S) and nonsuspicious class is denoted as (C_{NS}). Our test set also contains labeled text documents; this set is used to validate the model. Our dataset contains about 5000 samples of both suspicious and nonsuspicious speech. We considered 4000 speeches for training the model and 1000 ones for validating the model.

```
(1) Input: lists of tokenized documents, where Documents[i] represents the list of words present in i^th document.
(2) Goal: a word-to-index dictionary of input documents, Word_to_Index[w] gives the index of the word w
(3) index ⟵ 1
(4) Word_to_Index ⟵ {}
(5) for document_i in do
(6)       for word in document_i do
(7)          if word not in Word_to_Index then
(8)             Word_to_Index[word] ⟵ index
(9)             index ⟵ index + 1
(10)         else
(11)            do nothing (the word is already added to the dictionary)
(12)         end if
(13)      end for
(14) end for
```

Algorithm 2: Creating word indexed dictionary.

3.4.6. LSTM Network. Long short-term memory (LSTM) is a variant of RNN, a class of deep neural networks (DNN). RNNs emerged as efficient learners of sequential data. As text is sequence of word and preserving sequence is very important to interpret the actual context of the text, RNN based models are most suitable for text analysis [34]. RNNs are suitable for sequential data because, unlike other neural networks where all the inputs are independent from each other, in RNN inputs are interrelated.

Although RNNs are very powerful for learning sequence, they are practically vulnerable to the vanishing gradient problem [35]. Vanishing gradient problem means RNN fails to remember things in long past. It is possible that the sentiment of a text document can highly rely on the beginning portion of the text. So it can lead to misclassification of the text document as simple RNN cannot remember long-term dependencies. In order to handle the vanishing gradient problem, an improved variant of simple RNN is developed, which is called LSTM [36]. The structure of our proposed LSTM network is shown in Figure 4.

The length of the maximum sequence of words is set to 200. Zero padding is used for the text shorter than 200. Each text document is converted to a vector of integers $I_t = [w_1, w_2, \ldots, w_{200}]$, where each integer w_i represents a unique word. Our model takes vector of length 200 as input. We used an embedding layer of dimension 200. The embedding layer transforms each word (represented by integer) into a vector representation of length 200. The output of the embedding layer is then given to the LSTM layer. The LSTM layer contains 128 memory units. Each of the feature values is multiplied by the weights of each LSTM cell. The activation function "tanh" is used in the LSTM unit and the recurrent activation function is "sigmoid." The weighted sum of the dense layer of 128 units is used to map the output of the final output layer of single unit. We used "Binary Cross-Entropy" as the loss function and "Adam" optimizer to train the model [37].

3.4.7. Classification. Our proposed deep learning model based on LSTM is a binary classifier which classifies the input texts into suspicious (C_S) and nonsuspicious (C_{NS}) classes. More specifically, as it is binary classification problem, we have only one output neuron in our network. The output is given as the probability that the given text is suspicious. If the probability is greater than 0.5, then it is classified as suspicious (C_S); otherwise, it is classified as nonsuspicious (C_{NS}).

4. Dataset Preparation

The most important aspect of every experiment is the creation of a dataset. Dataset quality plays a prime role in any experiment's performance. The available amount of dataset in Bangla language is very low. In the domain of suspicious or hate speech detection, authors do not usually use publicly available datasets and they do not publish their own datasets [11]. There is no standard dataset available on Bangla suspicious text data, so we built our own dataset of suspicious and nonsuspicious Bangla text. Developing such dataset is a time-consuming and tedious task, as it demands a lot of attention; thus, defects cannot be introduced into the upcoming systems as long as sufficient data can be integrated into the dataset for robust program assessment.

One of the most challenging tasks of creating this dataset is to define what is suspicious speech. Nobata et al. define such speech as the language that attacks a group or community, which is based on religion, ethnic origin, gander, age, and disability [17]. There are many different definitions of suspicious speech or hate speech from different sources. Fortuna et al. identified four dimensions in which the comparison of these definitions can be made [16]. Several properties of the suspicious activity are defined by U.S. Department of Homeland Security[3]. To label a speech as suspicious, we must follow some criteria. We have set some properties for suspicious speech. If any speech meets one or more of these properties, the speech is labeled as suspicious. We sum up these properties in four main domains for collecting suspicious speech. These are the following:

(i) Religious humiliation: speech with intentional and malicious intent to offend religious feelings of any

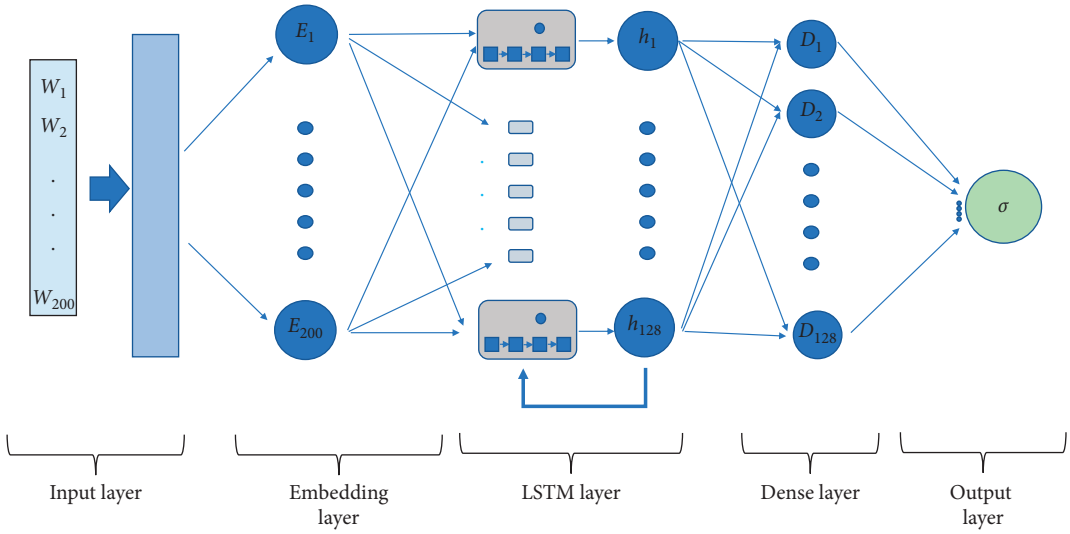


Figure 4: Proposed LSTM network.

class or speech aimed at insulting any religion or the religious beliefs.

(ii) Violence: speech that motivates people for any kind of terrorist activities, contains threat to the lives of people, or provokes violent activities.

(iii) Antigovernment: speech that provokes people against the government, law-enforcement agencies, or any community.

(iv) Offending nationalism: speech that disrespects the country or any national feeling.

The suspicious speeches are collected manually from YouTube, Facebook, and online blogs like Dhormockery 4, Shongshoy 5, and Istishon 6. The nonsuspicious speeches are also manually collected from various online resources.

4.1. Data Annotation. Three human annotators who are experts in the study of suspicious contents annotated the dataset. Based on our predefined four main domains of suspicious properties, all the 5,000 samples of data have been annotated blindly by these three experts. We calculated Fleiss' kappa [38], which is a statistical measure for determining the credibility of agreement between numbers of raters. The interrater agreement across the 5,000 samples was very satisfactory. The value of Fleiss' kappa was 0.959, which refers to almost-perfect agreement. There were still some disagreements between the annotators. In such case of disagreement, the speech was assigned to the class based on majority voting. The number of samples in each domain of the suspicious speech is shown in Figure 5.

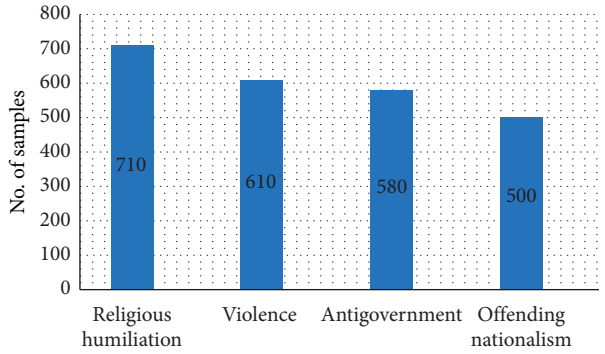


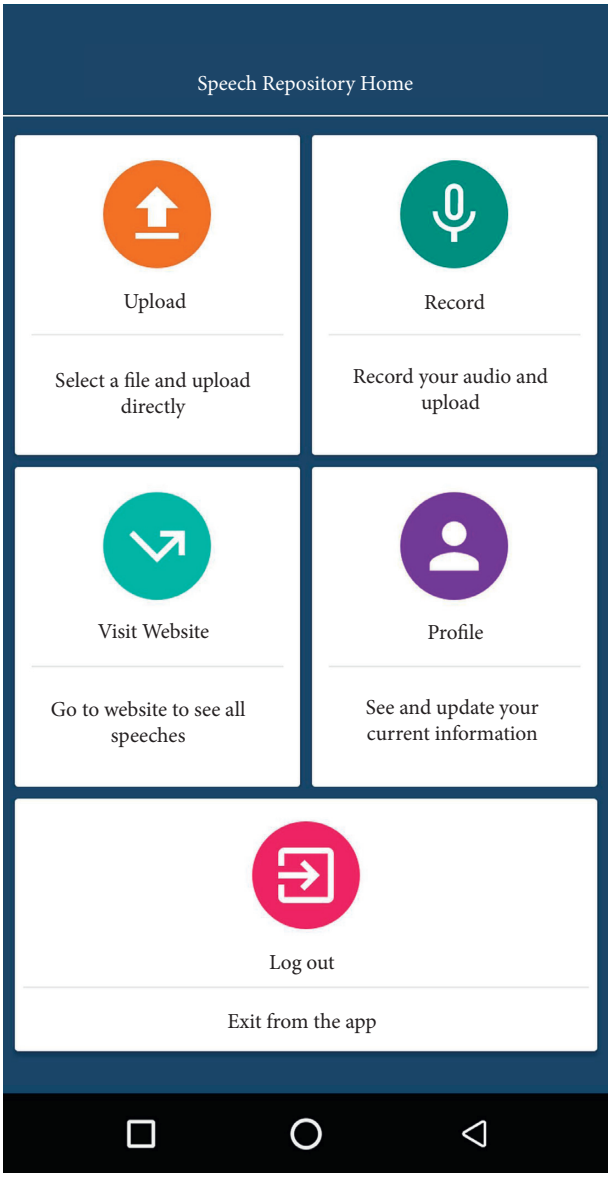
Figure 5: Number of samples in each domain of suspicious class.

5. Implementation and Evaluation

5.1. Implementation. For the purpose of acquisition of speech, we have developed an Android application. Acquisition of speech contains two parts: (i) recording of the speech and (ii) tracking the location of the speech. The phone's microphone is used for the purpose of recording the audio file. After recording, the file is stored in local storage. To track the location, the latitude and longitude are calculated. The user can also upload any audio file from the local storage. The interfaces of our Android application are shown in Figure 6.

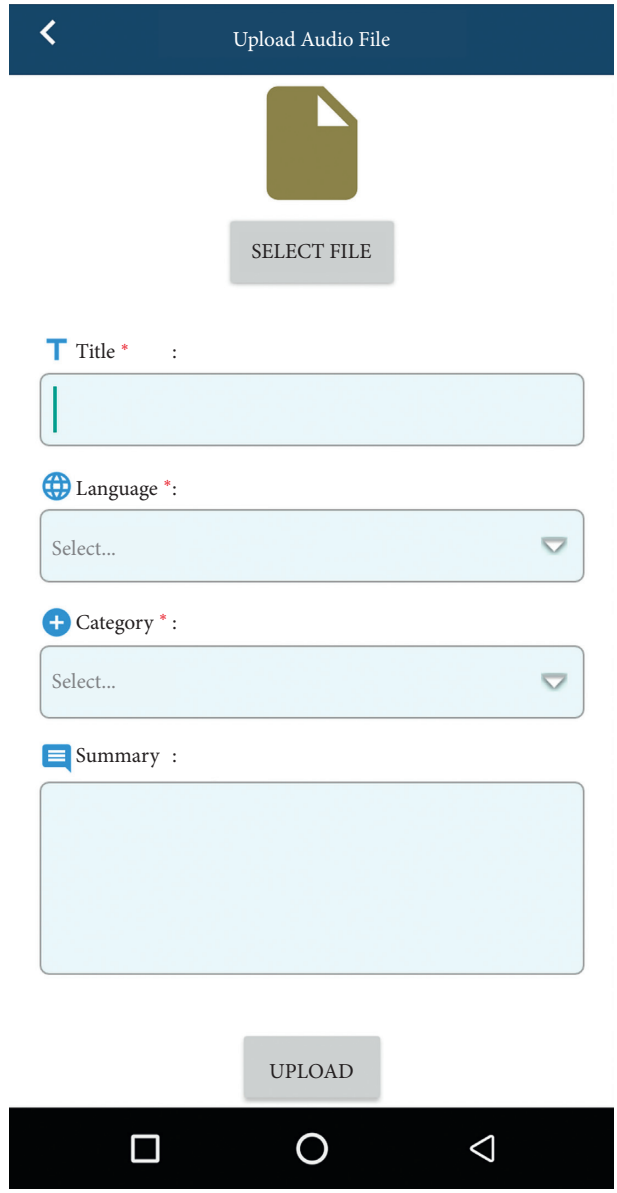
We used Google speech-to-text (gSTT) converter as our speech recognition API 7. From the latitude and longitude, the actual address of the speaker is calculated, which contains city, postal code, state, and country name. For validating the framework, we collected about 200 speeches from 10 different locations of Bangladesh from various speakers, both males and females. The performance of the speech acquisition process for 10 different locations is shown in Table 1. From the table, it can be seen that some location information was not available in some region (represented by null). It is because of the variation in geocoding detail. Moreover, our system can perform acquisition of speech with about 100% accuracy.

The performance of the speech recognition is dependent on the quality of the speech. The performance of our recognition module is shown in Table 2. From the table, we can see that speeches, which contain single language, have higher recognition accuracy than the speeches containing mixed languages. For a speech, if the total number of words is T,



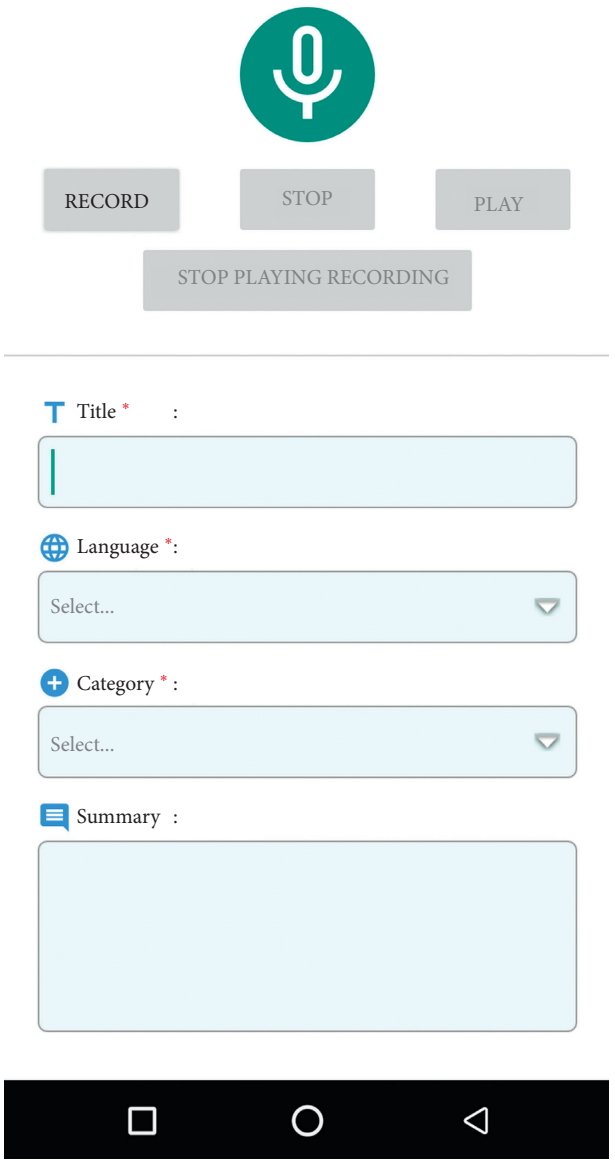
(a)

Figure 6: Continued.



(b)

Figure 6: Continued.



(c)

Figure 6: Interfaces of our Android application for acquisition of speech (a) Home interface, (b) Upload interface, (c) Record interface.

Table 1: Performance evaluation of speech acquisition process.

Speaker	Actual location	Detected location	Speech quality
1	Pahartoli, Chattogram, 4349, Bangladesh	Pahartoli, Chattogram, 4349, Bangladesh	Highly satisfactory
2	Habiganj, Sylhet, 3310, Bangladesh	Habiganj, Sylhet, 3310, Bangladesh	Highly satisfactory
3	Durgapur, Chandpur, 3640, Bangladesh	Durgapur, Chandpur, Null, Bangladesh	Satisfactory
4	Dhanmondi, Dhaka, 1208, Bangladesh	Dhanmondi, Dhaka, 1208, Bangladesh	Highly satisfactory
5	Gulshan, Dhaka, 1213, Bangladesh	Gulshan, Dhaka, 1213, Bangladesh	Highly satisfactory
6	Dinajpur, Rangpur, 5262, Bangladesh	Dinajpur, Rangpur, 5262, Bangladesh	Highly satisfactory
7	Jamalpur, Mymensingh, 2030, Bangladesh	Jamalpur, Mymensingh, 2030, Bangladesh	Highly satisfactory
8	Bogura, Rajshahi, 5892, Bangladesh	Bogura, Rajshahi, 5892, Bangladesh	Highly satisfactory
9	Barguna, Barisal, 8730, Bangladesh	Null, Barisal, 8730, Bangladesh	Satisfactory
10	Bagerhat, Khulna, 9301, Bangladesh	Bagerhat, Khulna, Null, Bangladesh	Highly satisfactory

total number of missing words is M, and total number of incorrect words is W, then the accuracy of the recognition is computed as

$$\text{accuracy} = \left(1 - \frac{M + W}{T}\right) \times 100\%. \tag{1}$$

5.2. Evaluation Metrics. For the purpose of assessing how good the classifier is at predicting the class of the sample, we need to evaluate some performance measures. We derived a set of values from the confusion matrix, True Positive (TP) denotes the number of nonsuspicious samples that were correctly classified as nonsuspicious, and True Negative (TP) denotes the number of suspicious samples that were correctly classified as suspicious. False Positive (TP) denotes the number of suspicious samples incorrectly classified as nonsuspicious, and False Negative (TP) denotes the number of nonsuspicious samples incorrectly classified as suspicious. Based on these numbers, we evaluated several performance measures. Precision refers to the measure of exactness. It specifies what percentage of samples classified as nonsuspicious is actually nonsuspicious. Precision is evaluated by the following equation:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{2}$$

Recall, also known as Sensitivity or True Positive Rate (TPR), is the measure of completeness. It specifies what percentage of nonsuspicious samples is classified as nonsuspicious. Precision can be calculated by the following formula:

$$\text{recall, sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{3}$$

F_1 – score denotes the harmonic mean of Precision and Recall. The overall system performance can be depicted by the F_1 – score. It is calculated by the following formula:

$$F_1 - \text{score} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \tag{4}$$

5.3. Results. Our system classifies the input text into suspicious (C_S) and nonsuspicious (C_{NS}) classes based on the prediction probability. For the training of our LSTM model, we used 4000 samples and the remaining 1000 samples are used for validating the model as test data. Confusion matrix is a very useful tool for evaluating classification algorithms. As our problem is a binary classification, the confusion matrix is a 2×2 matrix. The model is evaluated by the 1000 samples and evaluated confusion matrix is shown in Table 3.

Our model was compared with other models based on other machine learning algorithms like Naive bayes, SVM, decision tree, k-nearest neighbor, and logistic regression. The comparison is summarized in Table 4. Naive Bayes is a simple probabilistic approach based on Bayes theorem [39]. By counting frequencies and combinations of values in the specified dataset, it calculates sets of probabilities and made the class prediction based on these probabilities. The working procedure of SVM is to find a maximum distant hyperplane between classes [40]. The support vectors create a hyperplane for binary classification that divides the cases into two nonoverlapping groups. We used Linear Kernel SVM for classification. Decision tree is a very popular algorithm in the field of text classification. It works by breaking down a set of data into smaller pieces. External nodes represent the class of decision, while internal nodes have the necessary features to render classification [41]. In our paper, we used CART (Classification and Regression Trees) algorithm for decision tree, which is very similar to

Table 2: Recognition accuracy of the framework for different languages.

Audio file	Actual speech	Converted text	Detected language	Number of missing words	Number of wrong words	Accuracy %
001.wav	আমি ভাল। আমি। তুমি কেমন আছো? তুমি মে ওখালন যালে?	আমি ভাল। আমি তুমি কেমন আছো তুমি মে এখালন যালে	Bengali	0	1	90
002.wav	Birds are flying in the sky.It seems so beautiful when they fly.	Birds are flying in the sky it seems so beautiful when they fly	English	0	0	100
003.wav	الْحَمْدُ ه لِلّ رَ ه بِ ا لعَالَمِينَ الرحمن الرحيم مالك يوم ال ه دين	الْحَمْدُ ه لِلّ رَ ه بِ ا لعَالَمِينَ الرحمن الرحيم مالك يوم ال ه دين	Arabic	0	0	100
004.wav	Fearlessness is like a muscle. I Know from my own life. ধন্যবাদে।	Hear is a music I know from my own life ধন্যবাদে।	Mixed	1	1	83.33
005.wav	بِسْمِ اللهِ الرَّحْمٰنِ الرَّحِيْمِ How are you? কতারিা মে ভাল। আলো?	بِسْمِ اللهِ الرَّحْمٰنِ الرَّحِيْمِ How are you কতারিা আলো	Mixed	2	p	84.61

Table 3: Confusion matrix.

Confusion matrix	Predicted nonsuspicious	Predicted suspicious
Actual nonsuspicious	True positive TP = 675	False negative FN = 31
Actual suspicious	False positive FP = 19	True negative TN = 275

Table 4: Comparison of performance.

Classification algorithm	Accuracy	Error	Precision	Recall	F_1-score
Naive Bayes	0.87	0.13	0.90	0.92	0.91
SVM	0.90	0.10	0.94	0.91	0.93
Decision tree	0.88	0.12	0.95	0.87	0.92
k-nearest neighbor	0.71	0.29	0.78	0.82	0.80
Logistic regression	0.92	0.08	0.96	0.91	0.93
Proposed LSTM based model	**0.94**	**0.06**	**0.96**	**0.95**	**0.95**

C4.5. We used Gini impurity measure as the function to measure the quality of split. KNN simply assigns the class of an unknown sample object by considering the majority of votes among the *k*-nearest neighbors. Logistic regression is a suitable choice for binary classification. It classifies a given sample into one of two classes [42]. All of these algorithms are trained and tested with the same dataset and using same train-test ratio (80 : 20).

From the comparison, we can see that our proposed LSTM based model performs much better in terms of accuracy. The main difference between our proposed LSTM based model and other machine learning models such as logistic regression, SVM, KNN, and Naive Bayes is the way of learning from data. For general machine learning methods like logistic regression, SVM, KNN, and Naive Bayes, we need to extract the features from the texts before applying the algorithm. For feature extraction from documents, we use TF-IDF (term frequency-inverse document frequency) vectorizer as it offers a way to determine the significance of the word on the basis of how much it appears in different documents [43]. However, these algorithms are highly dependent on the frequency of the words and fail to remember things in the past in an efficient manner. As text is sequential data, LSTM suits best in performing this task by remembering long-term dependencies within the sentence. Among all these algorithms, *k*-nearest neighbor performs poorly in terms of accuracy because KNN performs classification based on majority voting instead of learning from data.

6. Conclusion and Future Research

In this study, we proposed a framework for acquisition and detection of suspicious Bangla speeches. Bangla is one of the most spoken languages in the world[8], but, to the best of our knowledge, no work was done to detect suspicious Bangla speech. Our proposed system can classify Bangla speech into suspicious and nonsuspicious categories. As there is no such dataset available, we developed a dataset that contains 5000 samples. We developed an Android application for the acquisition of the speech. The application stores the speech and converts the speech into text. Our proposed framework for classifying suspicious speech was evaluated on the test data and a comparison with other machine learning algorithms like Naive Bayes, SVM, decision tree, *k*-nearest neighbor, and logistic regression was made. Among these, our proposed LSTM based model performs better in terms of accuracy.

There are some scopes for future research in our work. The recognition accuracy of multilingual (when a speech

contains multiple languages) speeches can be improved. Currently, our system can recognize speeches in Bangla, English, and Arabic languages but can classify speeches only in Bangla language; in the future, this classification can be enhanced for English and Arabic languages as well. Moreover, the dataset can be enriched by incorporating more speech samples from various sources. The classification categories can be increased to classify suspicious speeches into more specific classes instead of binary classification. Multiclass classification will make the detection much more precise.

With the rapid growth of access to Internet and online platforms, the misapplications of technologies have also increased rapidly. Ill-intentioned speeches can create unwanted situations in the society. An automatic tool that can detect suspicious speeches benefits governments and social network platforms to prevent unexpected situations.

Data Availability

The data cannot be made available on websites for public use due to some restrictions. However, the data can be collected for further research upon request to the first author via email: rsdrcse14@gmail.com.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was funded and supported by the project "Establishment of IT Business Incubator at CUET" via Reference no. 56.02.0000.023.16.032.18.93 (date: 06/11/2018).

References

[1] P. Suedfeld, S. Bluck, E. J. Ballard, and G. Baker-Brown, "Canadian federal elections: motive profiles and integrative complexity in political speeches and popular media," *Canadian Journal of Behavioural Science/Revue Canadienne des Sciences du Comportement*, vol. 22, no. 1, pp. 26–36, 1990.

[2] H. Holzer, *Lincoln at Cooper Union: The Speech that Made Abraham Lincoln President*, Simon and Schuster, New York, NY, USA, 2004.

[3] M. Vail, "The "integrative" rhetoric of Martin Luther King Jr.'s "I have a dream" speech," *Rhetoric & Public Affairs*, vol. 9, no. 1, pp. 51–78, 2006.

[4] K. Gelber, "Terrorist-extremist speech and hate speech: understanding the similarities and differences," *Ethical Theory and Moral Practice*, vol. 22, no. 3, pp. 607–622, 2019.

[5] K. Gelber and L. McNamara, "Evidencing the harms of hate speech," *Social Identities*, vol. 22, no. 3, pp. 324–341, 2015.

[6] J. Waldron, *The Harm in Hate Speech*, Harvard University Press, Cambridge, MA, USA, 2012.

[7] E. Barendt, *Freedom of Speech*, Oxford University Press, Oxford, UK, 2nd edition, 2005.

[8] A. Buyse, "Words of violence: "fear speech," or how violent conflict escalation relates to the freedom of expression," *Human Rights Quarterly*, vol. 36, no. 4, pp. 779–797, 2014.

[9] A. K. Chen, "Free speech and the confluence of national security and internet exceptionalism," *Fordham Law Review*, vol. 86, no. 2, pp. 379–399, 2017.

[10] M. Islam and M. Islam, "Islam, politics and secularism in Bangladesh: contesting the dominant narratives," *Social Sciences*, vol. 7, no. 3, p. 37, 2018.

[11] M. Mondal, L. A. Silva, and F. Benevenuto, "A measurement study of hate speech in social media," in *Proceedings of the 28th ACM Conference on Hypertext and Social Media*, pp. 85–94, Prague, Czech Republic, July 2017.

[12] N. Gandhewar and R. Sheikh, "Google Android: an emerging software platform for mobile devices," *International Journal on Computer Science and Engineering*, vol. 1, no. 1, pp. 12–17, 2010.

[13] R. E. Rice and J. E. Katz, "Comparing internet and mobile phone usage: digital divides of usage, adoption, and dropouts," *Telecommunications Policy*, vol. 27, no. 8-9, pp. 597–623, 2003.

[14] P. Kaur and S. Sharma, "Google Android a mobile platform: a review," in *Proceedings of the 2014 Recent Advances in Engineering and Computational Sciences (RAECS)*, pp. 1–5, Chandigarh, India, March 2014.

[15] G. S. Chavan, S. Manjare, P. Hegde, and A. Sankhe, "A survey of various machine learning techniques for text classification," *International Journal of Engineering Trends and Technology*, vol. 15, no. 6, pp. 288–292, 2014.

[16] P. Fortuna and S. Nunes, "A survey on automatic detection of hate speech in text," *ACM Computing Surveys*, vol. 51, no. 4, pp. 1–30, 2018.

[17] C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang, "Abusive language detection in online user content," in *Proceedings of the 25th International Conference on World Wide Web—WWW'16*, pp. 145–153, Montreal, Canada, May 2016.

[18] B. Vidgen and T. Yasseri, "Detecting weak and strong Islamophobic hate speech on social media," *Journal of Information Technology & Politics*, vol. 17, no. 1, pp. 66–78, 2019.

[19] O. Oriola and E. Kotze, "Evaluating machine learning techniques for detecting offensive and hate speech in South African tweets," *IEEE Access*, vol. 8, pp. 21496–21509, 2020.

[20] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma, "Deep learning for hate speech detection in Tweets," in *Proceedings of the 26th International Conference on World Wide Web Companion—WWW'17 Companion*, pp. 759-760, Perth, Australia, April 2017.

[21] B. Gambäck and U. Sikdar, "Using convolutional neural networks to classify hate-speech," in *Proceedings of the First Workshop on Abusive Language Online*, pp. 85–90, Vancouver, Canada, August 2017.

[22] Z. Zhang, D. Robinson, and J. Tepper, "Detecting hate speech on Twitter using a Convolution-GRU based deep neural network," in *Proceedings od the The Semantic Web. ESWC 2018*, pp. 745–760, Heraklion, Greece, June 2018.

[23] J. Risch and R. Krestel, "Toxic comment detection in online discussions," in *Algorithms for Intelligent Systems*, pp. 85–109, Springer, Singapore, 2020.

[24] J. Salminen, M. Hopf, S. A. Chowdhury, S.-g. Jung, H. Almerekhi, and B. J. Jansen, "Developing an online hate classifier for multiple social media platforms," *Human-centric Computing and Information Sciences*, vol. 10, no. 1, 2020.

[25] S. M. A. Taher, K. A. Akhter, and K. M. A. Hasan, "N-gram based sentiment mining for Bangla text using Support Vector Machine," in *Proceedings of the 2018 International Conference*

on Bangla Speech and Language Processing (ICBSLP), pp. 1–5, Sylhet, Bangladesh, September 2018.

[26] A. N. Chy, M. H. Seddiqui, and S. Das, "Bangla news classification using naive Bayes classifier," in *Proceedings of the 16th International Conference on Computer and Information Technology*, pp. 366–371, Khulna, Bangladesh, March 2014.

[27] A. Dhar, N. Dash, and K. Roy, "Classification of Bangla text documents based on inverse class frequency," in *Proceedings of the 2018 3rd International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU)*, pp. 1–6, Bhimtal, India, February 2018.

[28] O. Sharif and M. Hoque, "Automatic detection of suspicious Bangla text using logistic regression," in *Proceedings of the International Conference on Intelligent Computing & Optimization*, pp. 581–590, Koh Samui, Thailand, December 2019.

[29] A. M. Ishmam and S. Sharmin, "Hateful speech detection in public Facebook pages for the Bengali language," in *Proceedings of the 2019 18th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 555–560, Boca Raton, FL, USA, December 2019.

[30] E. A. Emon, S. Rahman, J. Banarjee, A. K. Das, and T. Mittra, "A deep learning approach to detect abusive Bengali text," in *Proceedings of the 2019 7th International Conference on Smart Computing & Communications (ICSCC)*, pp. 1–5, Sarawak, Malaysia, June 2019.

[31] T. Islam, S. Latif, and N. Ahmed, "Using social networks to detect malicious Bangla text content," in *Proceedings of the 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*, pp. 1–4, Dhaka, Bangladesh, May 2019.

[32] P. Chakraborty and M. H. Seddiqui, "Threat and abusive language detection on social media in Bengali language," in *Proceedings of the 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*, pp. 1–6, Dhaka, Bangladesh, May 2019.

[33] M. Dwarampudi and N. V. S. Reddy, *Effects of Padding on LSTMs and CNNs*, https://arxiv.org/abs/1903.07288, 2019.

[34] G. Weiss, Y. Goldberg, and E. Yahav, "On the practical computational power of finite precision RNNs for language recognition," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pp. 740–745, Melbourne, Australia, July 2018.

[35] D. Britz, A. Goldie, M. Luong, and Q. Le, "Massive exploration of neural machine translation architectures," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1442–1451, Copenhagen, Denmark, September 2017.

[36] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[37] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *Proceedings of the 3rd International Conference for Learning Representations*, San Diego, CA, USA, May 2015.

[38] J. L. Fleiss, "Measuring nominal scale agreement among many raters," *Psychological Bulletin*, vol. 76, no. 5, pp. 378–382, 1971.

[39] S. Wang and C. D. Manning, "Baselines and bigrams: simple, good sentiment and topic classification," in *Proceedings of the 50th Annual Meeting of The Association for Computational Linguistics: Short Papers-Volume 2*, pp. 90–94, Jeju Island, South Korea, July 2012.

[40] M. Karan and J. Šnajder, "Cross-domain detection of abusive language online," in *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pp. 132–137, Brussels, Belgium, October-November 2018.

[41] J. Ali, R. Khan, N. Ahmad, and I. Maqsood, "Random forests and decision trees," *International Journal of Computer Science Issues*, vol. 9, no. 5, pp. 272–278, 2012.

[42] E. F. Unsvag and B. Gambäck, "The efects of user features on twitter hate speech detection," in *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pp. 75–85, Brussels, Belgium, October-November 2018.

[43] J. Salminen, H. Almerekhi, M. Milenković, S. Jung, J. An et al., "Anatomy of online hate: developing a taxonomy and machine learning models for identifying and classifying hate in online news media," in *Proceedings of the International AAAI Conference on Web and Social Media (ICWSM 2018)*, pp. 330–339, Palo Alto, CA, USA, June 2018.