Vulnerable Hate Speech Classification Using Bi-directional Recurrent Neural Network Model

Thae Thae Han
University of Computer Studies(Meiktila)
thaethaehan@ucsmtla.edu.

Mar Lar Htun
University of Computer Studies(Pakokku)
marlarhtun19@gmail.com

Moe Moe Thein
University of Computer Studies (Pyay)
cummthein.74@gmail.com

Abstract

The effect of hate speech can cause an unprecedented rate of vulnerability not only to public emotions but also political trends. It also increases discrimination between different racist groups. So, this paper addresses the problem of spreading hate speech over social network by autonomously detection the posts/tweets of the network users. Firstly, we first perform the pre-processing of language context using NLP tools and then exploit deep learning model called bidirectional recurrent neural network (Bi-RNN) in order to detect if the tweets are vulnerable hate speech or not. The system is then implemented according to proposed architecture and tested with popular Twitter dataset for analysis of hate speech. The experimental works are executed and measured with evaluation metrics called precision, error rate and processing time.

Keywords: hate speech detection, social network, deep learning model, natural language processing, bidirectional recurrent neural network

1. Introduction

Spreading the hate speech over social network sites (SNS) has been baffling to the social network developers regarding how to control it although there are some intense researches having done in this area since recent years.

Simons [7] highlighted that there is a need to have analytical research for providing insights to empower victims, to increase campaign among the public and to discourage perpetrators. In the same way, Barlow and Awan [8] suggested that the social networks companies, like Twitter, Facebook, etc., should take corrective measures to watch and prevent online abuse against women and Muslims. Edwards [9] identified that women are recruited by terrorist organizations mainly to meet sexual requirements of the men.

Traditionally, the process of analyzing text information belongs to the area of Natural Language Processing (NLP). However, human languages are sparse and it is changing over the time due to more popular usage of internet slangs, emoticons, stickers, etc. than normal speaking languages. As a result, handling the human text alone with NLP becomes insufficient because they are too restricted to learn internet languages which is every likely to change depending on trends. Therefore, intuitively, it is uncontrollable to hate speech spread over SNS without powerful and self-learning language models instead of using pre-defined rules and assumptions.

To fill the gaps of abovementioned bottleneck of NLP, deep learning models take place to enhance the power of learning the human language under variety of context and meaning without pre-learning or pre-training through lexicon or language-rules.

This paper presents a deep learning model called bi-directional RNN to learn the nature of human Tweets over Twitter SNS, to decide whether they are hating related speech or not. In order to feed the text into RNN deep learning model, we first alter the textual context into numeric values because deep neural models can work only with numbers instead of categorical text. To do so, we change text into corresponding numeric vectors with the aid of NLP tools.

In this paper, there are two main parts to perform hate speech detection. The former one is dealing with NLP tools for numeric vectorization while the latter one is to detect or classify the input Tweet text using Bi-RNN model.

The rest of the paper is organized as follows. The related work is studied in Section 2 and the next section explores the underlying theory. The applied algorithms and their processes are explained in detailed at Section 4 and the experimental results are discussed at Section 5. The paper is concluded in Section 6 by describing also its potential future work.

2. Related Work

Due to an increase development of instant technology, online hate speech has been increasingly spread among mobile and internet users. Recent studies confirm that exposure to online hate speech has serious

offline consequences to historically deprived communities.

Similarity, the paper [3] presents a mobile edge computing architecture for regulating and reducing hate content at the user's level. In this regard, the profiling of hate content is obtained from the results of multiple studies from social network sites and synthetic datasets. It then separates the posts into different categories of hate content caused by gender, religion, race, and disability. An architectural framework is developed accordingly to regulate and reduce hate content at the user's level in the mobile computing environment. The author exclaims that their proposed architecture is novel and it could reduce hate content generation and its impact.

In paper [4], they propose a hate speech detection approach to identify hatred against vulnerable minority groups on social media. First of all, social network posts are automatically crawled with Spark API and pre-processed, and features are extracted using NLP tools such as n-grams and word embedding techniques such as Word2Vec. Second of all, deep learning algorithms are used to filter the hate speech. They are Gated Recurrent Unit (GRU) and a variety of Recurrent Neural Networks (RNNs). As a final step, hate words are clustered with methods such as Word2Vec to predict the potential target ethnic group for hatred.

A research group to which belongs to [5], defines hate speech as speech which either promotes acts of violence or creates an environment of prejudice that may eventually result in actual violent acts against a group of people [6]. In the case of Ethiopia, the use of hateful words has an intention to bring about hatred against a group of people based on their ethnicity, political attitude, religion and socio-economic status.

Deep learning networks are applied in the studies [1] and [2]. The work [1] uses deep network model to detect the paraphrase detection in short text messages. The study [2] develops their own Twitter-specific lexicon to replace traditional sentiment lexicons. They afterwards use dynamic neural deep network for sentiment classification using DAN2 and SVM.

In regard to controlling the hate speech spread over social networks, studies [10] and [11] perform analysis over social activities such as posts, comments, tweets, etc., from the online users in order to detect and control their vulnerable intention to the public.

Inspired to those abovementioned works, our paper deals with hate speech detection and classification using NLP tools as preliminary preparation before being fed to deep neural network called Bi-RNN model.

3. Background Theory

This section provides a glimpse of underlying theories which are used in this paper.

3.1. Hate Speech Sentiment

Sentiments refer to opinions, and emotions. Generally speaking, hate speech could be seen as the expression of vulnerable intentions towards an individual or some community (politics, business, etc) owing to a characteristic they share, or a group to which they belong. In [3], the term personal attack is used to describe offensive online behavior, while other studies focus on offensive or abusive speech and online harassment [10,11].

In order to autonomously analysis hate speech detection, natural language processing, text analysis, and computational linguistics are mostly used in earlier studies. The purpose is to reveal the attitude of a speaker or writer towards a specific topic such as positive or negative attitude. In this paper, we find out that if they are related with hatred or not.

3.2 NLP-based machine learning methods

The main goal of hate speech detection is to develop a tool that can process tweets in social media and highlight the most likely tweets or posts containing hate speech for manual inspection.

The concept behind NLP is a concept called word embedding. Word embeddings are representations of words as vectors, which is mapped to one vector composed of numeric values for each word found in a sentence. They are then fed into learning model to learn how each work is associated with each other and which way they lead, positive or negative, etc.

3.3 Bi-RNN Deep learning methods

The idea behind RNNs is to make use of sequential information. In a traditional neural network, we assume that all inputs (and outputs) are independent of each other. To predict the next word in a sentence we better know which words came before it. RNNs are called recurrent because they perform the same task for every element of a sequence, with the output being depended on the previous computations and you already know that they have a “memory” which captures information about what has been calculated so far. Therefore, we choose RNN for analyzing text classification of social network posts to discriminate them as “hate” or “non-hate” speech.

4. Tweets Hate Speech Detection

4.1 System Architecture

In this paper, we propose a hate speech detection system using the tweets collected from social media. The

overall architecture of the proposed method is depicted in Fig. 1.

There are two main stages in architecture - training phase that use dataset to create trained model and testing phase to predict the class label of new Tweet as hate or non-hate tweets. Both phases use the same backbone of system architecture: NLP related tools and Bi-RNN model. The detailed explanation of each process is provided in the following sections.

4.2 Text Preprocessing

As a preliminary step of hate speech classification, we start with smoothing the text, called pre-processing the text. To split the sentences into simpler words we tokenize the text. In here, NLTK tokenizer is used. Thereafter, we remove stopwords and perform lemmatization using Python library. After pre-processing steps, only sensible words, which are in base form, are collected.

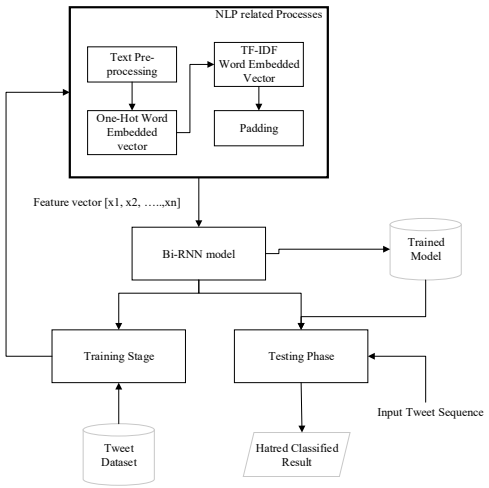

Figure 1. System Architecture

4.3 Word Embedding

For the deep neural models, we need embeddings for the text because numerical values can capture the linguistic meaning of a word in its context, that is, which word plays in what kind of roles (such as positive, negative, toxic, fake, etc). In order to do so, the text data are converted into high dimensional space called embedded vector, the represents corresponding numeric values that gives information of the word is how significant to that context.

However, there are tremendous number of words in English language, which is difficult to build every word in Tweet as vector because large size of vector can slow down the speed of learning model. Apart from that, it does not necessarily need to preserve all words occurrence for every Tweet because every Tweet does not need to use every English word.

Therefore, in this paper, we use one-hot-vector encoding that signals the usage of words, that is, its value is 1 if it is used in the Tweets. Otherwise, it will be 0. Depending on that one-hot-vector, we just extract the words which indicates 1, that used in the Tweet. Depending on their 1s, we extract embedded values from an embedded vector as shown in Fig 2. In the figure, 8676 will be the index of a word included in Tweet context from the dataset. For example, the word of 8676 index may be “girl” or “park” or any word in Tweet.

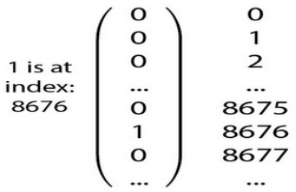

Figure 2. How embedded vector and one-hot vector works

Furthermore, in this paper, we bring the value of TF-IDF as the weight values for each vector element so that we can outweigh the roles of each word, based on their occurrences in Tweets according to (1). Due to limited space, the readers are requested to explore TF-IDF calculation at the references [12].

$$N_i = W_i * X_i \qquad (1)$$

where W_i and X_i stand for weight and word element in the number n (i=1,….,n).N_i is the neuron to be fed into the Bi-RNN network. In this case, the number of input neuron will vary depending on its number of words included, which is quite impossible for Bi-RNN model to handle varying length of input. It is because the model can work only with fixed number of input neurons.

In order to solve this problem, we limit the size of input number as 256 for maximum, which is the most suitable size to grasp the most required information from a Tweet. However, the problem arises how to deal with the input number which is less than that size. For this case, we use zero-padding for required spaces as shown in Table 1.

In table 1, the first row shows the number of words, which is 256 in maximum. The second row is input text sequence and the last row is transformed word vector from input text, which is padded with zero to have fixed 255 length. The last row of the table will act as input vector for Bi-RNN model.

Table 1. Zero-padded input sequence with their word embedded vector

0	1	2	3	4	5	6	.	.	255

th e	gi rl	is	walki ng	in	th e	par k			
23 4	32	45 6	1264	34 2	23 4	12	0	0	0

4.4 Bi-RNN for Hate Speech Classification

The reason of choosing bi-directional RNN model for the proposed approach is that language text is related in both forward and backward directions. For example, there are two sentences which have similar meaning but different word positions and word usages. For example, let first and second sentence be "He dislikes those who write rude posts over the Facebook." and "The people whom he dislikes are the ones who post toxic information over SNS."

The idea of Bi-RNN is to split the state neurons of a regular RNN in a part that is responsible for the positive time direction (forward states) and a part for the negative time direction (backward states). By doing so, it can explore more semantic relationships between the predicates (subject, object, etc). Therefore, this paper choses Bi-RNN model to handle language-based text data classification as depicted in Figure 3.

There are four layers in our model and the model is looped back into two directions until it reaches the expected loss value.

The input layer will have 10 timesteps (noted as t_i, i=1…,T) with a feature vector, which is in size of 256. The first hidden layer will have 32 memory units and the output layer will be a fully connected layer that outputs one value per timestep. The dropout function is added at the end of hidden layer in order to regularize between input and expected output values.

The next hidden layer is a TimeDistributed wrapper layer which is used around the output layer to estimate given the full sequence provided as input. This requires that the RNN hidden layer returns a sequence of values in every timestamp rather than a single value for the whole input sequence.

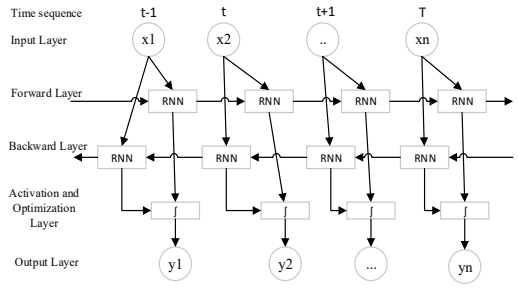

Figure 3. Bi-RNN model for hate speech classification

A sigmoid activation function is used on the output to predict the binary value (hate or non-hate). This layer mainly uses log loss entropy in finding the difference between expected and gained values to determine if the model should stop or go back to the beginning. To optimize the results, an efficient ADAM optimization algorithm is used for calculating and updating the values of weights and accuracy for each epoch.

5. Experimental Works

5.1. Dataset

The paper implements Tweets hate speech detection upon offline datasets [12], which is crawled from Twitter using Twitter API. In the dataset, there are two packages – one is for training and another one is for testing purpose. The former one includes three attributes: id, class label (0 for non-hate and 1 for hate) and tweet. The training dataset contains total number of 32k tweets while the testing one includes 17k tweets without class label. We use both packages for training and testing purpose accordingly.

5.2. Experimental Setup

The proposed system is developed using Python programming with window 10, i5. For NLP tools, NLTK library is used; and, for implementing Bi-RNN, tensorflow-2.0 and sklearn1.5 are used, which are very popular packages for deep learning and machine learning models. We use CPU version instead of GPU or TPU units.

This paper can only run the experiments with different lesser volume of data than its original size (32K for training and 17k for testing). Intuitively, testing large volume of data needs special processing speed and power like GPU or TPU, so, unfortunately, we can use only at most 8k for training and 4k for testing to be compatible according to our CPU processor's capabilities.

5.3. Performance Analysis

The evaluation is performed in both training and testing phase and results are shown according to their precision, recall and processing time. Precision is measured upon how the system can accurately classify the hate speech based on the number of all its classified speeches. Conversely, error rate is measured how much mistakenly classified by the system. The processing time is regarded from the starting to end time of the system.

In this paper, we compare our own system under different volumes ranges, alternated NLP tools without comparing with others, because, as we mentioned earlier, our system implementation is under limitation in processing power unlike other approaches [1][2][10][11], which used powerful processing unit called TPU or GPU. Therefore, it is not reasonable to

compare with their results when the background developing tools are different.

Table 2. Performance Analysis upon different data volumes

Different phase of our proposed Bi-RNN model	Precision (%)	Error Rate (%)	Processing Time (seconds)
Training	93	16	450
Testing	91	19	78

In Table 2, therefore, we measure the results by repeatedly running our own program and then averaging the values accumulated from 8 times running from training phase with 1k data volume for each run, and 4 times for testing phase. The results have no big difference between training and testing phase, which proved that it can avoid overfitting while it preserves the optimized results.

Table 3. Performance Analysis upon different NLP techniques

Different NLP Techniques with Bi-RNN model	Precision (%)	Error Rate (%)	Processing Time (seconds)
WE	84	27	134
TF-IDF+WE	91	19	78

Table 3 evaluates the performance of the system in testing phase. These results are collected by running Bi-RNN model with different tuned NLP techniques.

As one can be seen, the values with TF-IDF weighting values are higher than that without those values, which uses random weight values by WE. It has shown that the weighting value of TF-IDF used by our whole Bi-RNN system achieves better results in all evaluation metrics compared to absence of using it, which used only WE.

To conclude the experimental results, our full designed architecture reveals more better results (in both Table 1 and Table 2) than partial implementation of it. Moreover, the testing phase also achieves as same as training phase, due to intensive learning (bi-directional) of RNN without consuming too much time.

6. Conclusion

The paper presented a complete vulnerable hate-speech detection system over Twitter network. It had showed that it achieves significant results in various evaluation metrics. The system is implemented with Python programming and its libraries for NLP tools and Bi-RNN model. According to the experiments, we achieve the promising performance results such as precision, recall and processing time.

As a future work, we would like to extend this paper to cluster the vulnerable community depending on vulnerable rate to the public.

References

[1] Basant Agarwal, Heri Ramampiaro, Helge Langseth, Massimiliano Ruocco, A deep network model for paraphrase detection in short text Messages, *Information Processing and Management*, 54 (2018) 922–937

[2] M.Ghiassi, J.SkinnerbD.Zimbra, Twitter brand sentiment analysis: A hybrid system using n-gram analysis and dynamic artificial neural network, *Expert Systems with Applications*, Volume 40, Issue 16, 15 November 2013, Pages 6266-6282

[3] Naganna Chetty, Sreejith Alathur, An architecture for digital hate content reduction with mobile edge computing, *Digital Communications and Networks*, 2019.

[4] Zewdie Mossie, Jenq-Haur Wang, Vulnerable community identification using hate speech detection on social media, *Information Processing and Management*, July 2019.

[5] Saleem, H. M., Dillon, K. P., Benesch, S., & Ruths, D. A web of Hate: Tackling hateful speech in online social spaces. *Proceedings of the first workshop on Text Analytics for Cybersecurity and Online Safety* (TA-COS), collocated with LREC 2016.

[6] Susan, Benesch. Defining and diminishing hate speech. *State of the world's minorities and indigenous peoples* 2014 (pp. 18–25). 2014.

[7] Simons, R.N. Addressing Gender-Based Harassment in Social Networks: *A Call to Action. In iConference 2015 Proceedings*, http://hdl.handle.net/ 2142/73743. 2015.

[8] Barlow, C., & Awan, I. You Need to Be Sorted Out with a Knife: The Attempted Online Silencing of Women and People of Muslim Faith Within Academia. Social networks + Society, 1-11. 2016.

[9] Edwards, S. S. Cyber-Grooming Young Women for Terrorist Activity: Dominant and Subjugated Explanatory Narratives. In Cybercrime, Organized Crime, and Societal Responses (pp. 23-46). Springer, Cham. 2017.

[10] Zewdie Mossie, Jenq-Haur Wang, Vulnerable community identification using hate speech detection on social media, *Information Processing and Management*, Volume 57, Issue 3, May 2020, 102087

[11] Polychronis Charitidis, Stavros Doropoulos, Stavros Vologiannidis,Ioannis Papastergiou, Sophia Karakeva, Towards countering hate speech against journalists on social media, *Online Social Networks and Media*, 17 (2020) 100071

[12] https://www.kaggle.com/vkrahul/twitter-hate-speech#train_E6oV3lV.csv