TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

THESIS NO: 073/MSCS/654

Opinion Mining for the detection of Hate Speech on Social Media

by

Gajendra Acharya

A THESIS

SUBMITTED TO

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE IN COMPUTER SYSTEM AND KNOWLEDGE ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

LALITPUR, NEPAL

November, 2019

Opinion Mining for The Detection of Hate Speech On Social Media

by

Gajendra Acharya

073/MSCS/654

Thesis Supervisor

Prof. Dr. Subarna Shakya

A thesis submitted in partial fulfillment of the requirement for

the degree of Master of Science in Computer System and Knowledge Engineering.

Department of Electronics and Computer Engineering

Institute of Engineering, Pulchowk Campus

Tribhuvan University

Lalitpur, Nepal

November, 2019

COPYRIGHT ©

DECLARATION

I declare that the work hereby submitted for Master of Science in Computer Systems and Knowledge Engineering (MSCSKE) at IOE, Pulchowk Campus entitled “Opinion Mining for the Detection of Hate Speech on Social Media" is my own work and has not been previously submitted by me at any university for any academic award. I authorize Institute of Engineering, Pulchowk Campus to lend this thesis to other institution or individuals for the purpose of scholarly research.

Gajendra Acharya
073/MSCS/654
November, 2019

RECOMMENDATION

The undersigned certify that they have read and recommended to the Department of Electronics and Computer Engineering for acceptance, a thesis entitled "**Opinion Mining for The Detection of Hate Speech On Social Media**", submitted by **Gajendra Acharya** in partial fulfillment of the requirement for the award of the degree of "**Master of Science in Computer System and Knowledge Engineering**".

....

Supervisor: Prof. Dr. Subarna Shakya,

Department of Electronics and Computer Engineering,

Institute of Engineering, Tribhuvan University

....

External Examiner:

Prof. Dr. Bal Krishna Bal

....

Committee Chairperson: Dr. Aman Shakya,

Program Coordinator,

Computer System and Knowledge Engineering

Institute of Engineering, Tribhuvan University.

Date: November, 2019

DEPARTMENTAL ACCEPTANCE

The thesis entitled **“Opinion Mining for The Detection of Hate Speech On Social Media "**, submitted by **Gajendra Acharya** in partial fulfillment of the requirement for the award of the degree of **“Master of Science in Computer System and Knowledge Engineering"** has been accepted as a bonafide record of work independently carried out by him in the department.

…………………………………………………………

Associate Prof. Dr. Surendra Shrestha
Head of the Department
Department of Electronics and Computer Engineering,
Pulchowk Campus,
Institute of Engineering,
Tribhuvan University,
Nepal

ACKNOWLEDGEMENT

I would like to express my special thanks of gratitude to the Department of Electronics and Computer Engineering for giving a golden opportunity to carry out this thesis work. I owe my deep gratitude to my thesis supervisor Prof. Dr. Subarna Shakya, who took keen interest in this work and guided me by providing all the necessary information for developing a good system.

I am thankful to our program coordinator Dr. Aman Shakya for providing suitable platform to prepare this report.

I owe a debt of gratitude towards our Head of Department Assoc. Prof. Dr. Surendra Shrestha, Assoc. Prof. Dr. Diwakar Raj Panta, Prof. Dr. Shashidhar Ram Joshi, Assoc. Prof. Dr. Sanjeeb Prashad Pandey, for their encouragement and precious guidance during the thesis, and also to our other faculty members of the department as well as my friends for providing their valuable views and ideas.

Gajendra Acharya

073/MSCS/654

ABSTRACT

Since hate speech and offensive language is becoming a growing problem, with the massive availability and popularity of opinion-rich resources such as social media, online review sites Hate speech on social media has unfortunately become a common occurrence largely due to advances in mobile computing and the internet. This thesis work analyses the social media content to try and identify hate speech, and with the application of various machine intelligence models, improving the detection rate with higher accuracy in contrast to previous research works. Various deep learning approaches such as Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) are used in the proposed approach. With the application of sentiment weighing, the opinion mining is carried out. The final model shows overall precision of 0.82, recall 0.78 and F1 score of 0.80. The test accuracy achieved in the dataset is 83%. For the visual interpretation, ROC curve shows the distinction between the abusive or offensive and non-offensive content.

Keywords: CNN, LSTM, hate, opinion.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

BI-LSTM	Bidirectional Long Short-Term Memory
BOW	Bag of Words
CNN	Convolutional Neural Network
GRU	Gated Recurrent Unit
IDE	Integrated Development Environment
LSTM	Long Short-Term Memory
ML	Machine Learning
NLP	Natural Language Processing
POS	Part of Speech
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
SA	Sentiment Analysis
SVM	Support Vector Machines
TF-IDF	Term Frequency-Inverse Document Frequency

1. INTRODUCTION

1.1 Background

Social Media have become a trusted medium for seeking connectedness, and resulting our connections to grow wide and broad but shallower. On the other hand, Social Media, web forums and online conversations and debates have brought various issues like anxiety, decrease in empathy and antisocial behavior like hate speech. Machine learning models are currently being used in the field to detect abusive language and hate speech on social media platforms including YouTube, Facebook, Instagram, and so on. This study tries to emphasize on hate speech and abusive language detection approaches and build a new system through the study of those existing tools and techniques. Researchers have tried to perform a large number of research in the past decade to develop automatic methods for hate speech detection in blogs and social media. A more detailed discussion of the methods and applications for hate speech detection can be found in many of the surveys carried out in recent years.

Hate Speech

Despite the strong presence of hate speech in public debates, there is not a universally recognized definition of this phenomenon (Spallaccia and Beatrice 2017). While researchers try to determine the content, tone and nature of this discourse, the main definitional challenge of this debate still lies in the identification of the social categories attacked by hate speech. Commentators have supported the separation of misogyny from other forms of hate speech (e.g., racism, sexism, homophobia and other offensive forms) more or less explicitly, ranging from its absence in the definitions of this phenomenon to the overt justification of such exclusion. Building effective countermeasures for hate speech online requires as the first step, identifying and tracking hate speech (Zhang 2018). However, many of recent works in hate speech identification are based on racist and sexist slurs. The definition of hate speech is different among the authors based on geographical locations, cultures and nationalities. However, this study is based on the following definitions of hate speech. Other definitions are similar and revolve around the gist of these definitions.

Davidson (2017) defines hate speech as *the language that is used to express hatred towards a targeted group or is intended to be derogatory, to humiliate, or to insult the members of the group.*

George (2018) proposes the following list to identify hate speech. A hate speech is any speech that-

1. Uses a sexist or racial slur.
2. Attacks, seeks to silence, or criticizes a minority (without a well-founded argument).
3. Promotes but not directly use, hate speech or violent crime.
4. Blatantly misrepresents truth or seeks to distort views on a minority with unfounded claims.
5. Contains a screen name that is offensive, as per the previous criteria, tweet is ambiguous (at best), and tweet is on a topic that satisfies any of the above criteria.

Hate speech is characterized by any form of verbal or non-verbal attack targeting a specific group of people. It can be motivated by racial, ethnic, gender, religion or sexual orientation and other similar issues. Such issues are usually communicated through different media such as internet, hand-held devices, newspapers, magazines, television, radio broadcasts, and verbal person-to-person.

Opinion Mining

Opinion Mining, also interchangeably used as Sentiment Analysis (SA) is a field within Natural Language Processing (NLP) that builds systems that try to identify and extract opinions within the text however, the difference lies in a person's view about something and a feeling. Usually, besides characterizing opinion, these systems extract expression attributes, for example:

- *Opinion holder*: the entity or person that expresses the opinion.
- *Polarity*: the speaker expresses a *positive* or *negative* opinion,
- *Subject*: the thing that is being talked about,

The distinction lies in the sense that a sentiment is more of a feeling, and an opinion is more of a person's view about something, whereas. However, opinions imply positive or negative sentiments in most cases. Because detecting the polarity of text is often a step in sentiment analysis, the two fields are usually combined under the same umbrella as mentioned in [1]. Since negative opinions and hate speech are closely related, and it is safe to assume that usually negative sentiment might imply a hate speech message (Nobata et al. 2016).

1.2 Problem Statement

With the massive growing content on social media, the amount of hate speech online is also increasing. While hate speech and abusive language detection has become an important area for natural language processing research (Schmidt and Wiegand, 2017; George et al., 2017; Fortuna and Nunes, 2018), there has been little work addressing the potential for these systems to be biased. Such bias could result in negative impacts on the results. Due to subjective bias, far fewer data are classified as more offensive or hateful than their true category. Recent works done on the identification of hate speech on social media data prove to be less accurate due to erroneous classification of offensive language as hate speech (Davidson et. al., 2017), or having very small dataset (Mugambi, 2017), and some do not consider sentiment weighting (Mugambi, 2017). Some works on hate speech detection might have identified these problems but many studies still tend to combine various classification models for better results.

1.3 Objectives

a) To develop a hate speech detection system through the investigation of existing techniques.

b) To evaluate the performance of the proposed system.

2. RELATED THEORY

2.1 Related Works

Most of the recent methods primarily present the problem as a supervised text classification task which can be divided into two categories (Zhang et al. 2018): Classical methods that rely on feature engineering such as Naïve Bayes, SVM, Logistic Regression and Decision Trees; and deep learning-based methods that make use of neural networks to automatically learn multi-layers of abstract features from raw data.

Wafula [2] uses POS tagging and a bag of words model for supervised classification of tweets data for hate speech sentiment analysis by developing a social media digital forensics tool through the design, development and implementation of a software application. They also state that the system could go further by employing machine intelligence algorithms for predictive analysis. Schmidt et. al. [3] conducted a survey on hate speech detection with the assumption that usually negative sentiment pertains to a hate speech message. They employ two predominant supervised learning approaches, namely SVM and RNN models. Most of the recent works use lexicon based approach, and classifiers such as Naive Bayes and SVM, and a few use Maximum Entropy and Neural Network based models for sentiment analysis. Mugambi [4] applies machine learning techniques to automatically classify tweets as hate speech or not. Davidson et. al. (2017) uses crowd-sourcing to label a sample of these tweets into three categories: those containing hate speech, those with offensive language, and those with neither. They train a multi-class classifier to distinguish between these different categories.

More recently, new feature extraction techniques have been applied based on word embedding (also known as word vectors). This kind of representation makes it possible for words with similar meaning to have a similar representation, which can improve the performance of classifiers.

2.2 Approaches in Related works

Deep learning-based methods that make use of neural networks to automatically learn multi-layers of abstract features from raw data (Zhang et al. 2018). The most popular network architectures are CNN (Convolutional Neural Networks) and RNN (Recurrent

Neural Networks). CNN is well-known for its effective feature extraction capabilities (Zhang 2018). RNN assigns more weight to the previous data points of sequence. RNN learns word or character dependencies in tweets (Zhang 2018).

LSTM and GRU

Early RNNs had trouble with training because of exploding and vanishing gradients. With many time steps the gradients would often grow too steep, exploded, or they approached zero, vanished. This problem happened because the recurrent edge in a node always had the same weight, which resulted in the derivative of the error either exploding or approaching zero, at an exponential rate, as the number of time steps grew.

This was solved by introducing a memory cell. The new model was introduced by and is called LSTM. GRU is a simpler version of LSTM, and therefore less computationally expensive, but often without performing worse than LSTM.

Convolution Neural Network (CNN)

CNNs are the feed forward neural networks made up of many hidden layers. They consist of filters or kernels or neurons that have learnable weights or parameters and biases. Each filter does convolution on some inputs as shown in figure 2.1.
The components of CNN consist of following layers:
i. Convolutional Layer
ii. Rectified Linear Unit (ReLU) Layer
iii. Max Pooling Layer
iv. Fully Connected Layer

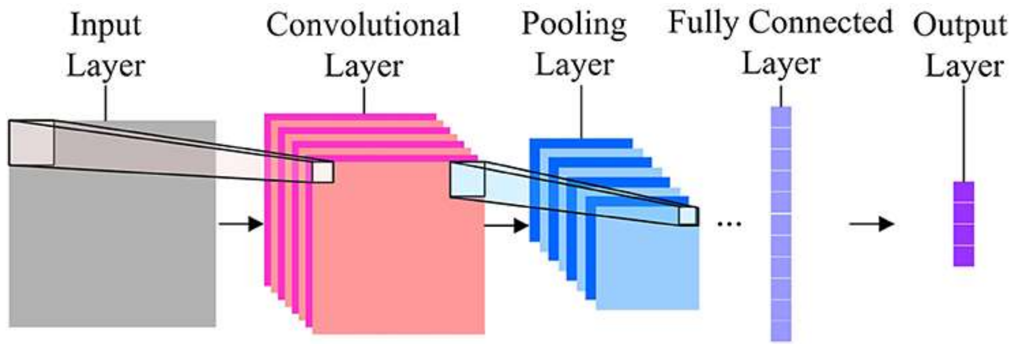


Figure 2.1: CNN Architecture (Source: semanticsscholar.org [online])

Recurrent Neural Networks (RNN)

Another neural network architecture that is addressed by the researchers for text mining and classification is Recurrent Neural Networks (RNN).

Therefore, this technique is a powerful method for text, string and sequential data classification. In RNN, the neural net considers the information of previous nodes in a very sophisticated method which allows for better semantic analysis of the structures in the dataset. A simple RNN is shown in figure 2.2.

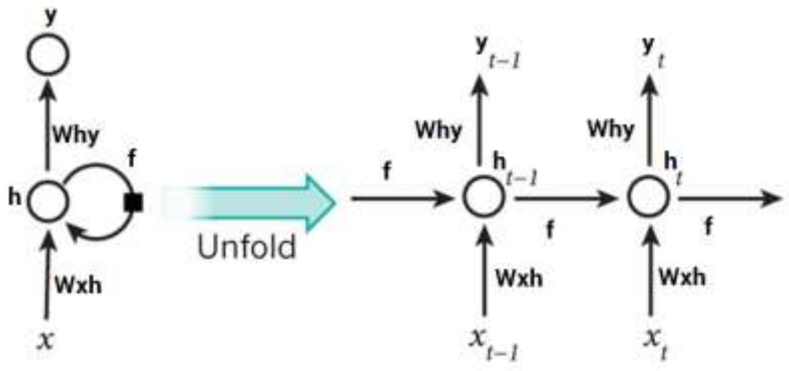


Figure 2.2: Recurrent Neural Network (Source: analyticsvidhya.com [online])

Long Short Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a special type of RNN that preserves long term dependency in a more effective way compared to the basic RNNs. This is particularly useful to overcome vanishing gradient problem. LSTM takes whole document as single sequence and the average of the hidden states of all words is used as feature for classification (Yang et al. 2016).

A General Recurrent Neural Network (RNN) can model temporal information by transforming a sequence of inputs to a sequence of outputs. LSTMs enable RNN's to remember their inputs over a long period of time. This is because LSTMs contain their information in a memory that is much like the memory of a computer because the LSTM can read, write and delete information from its memory, as shown in figure 2.3.

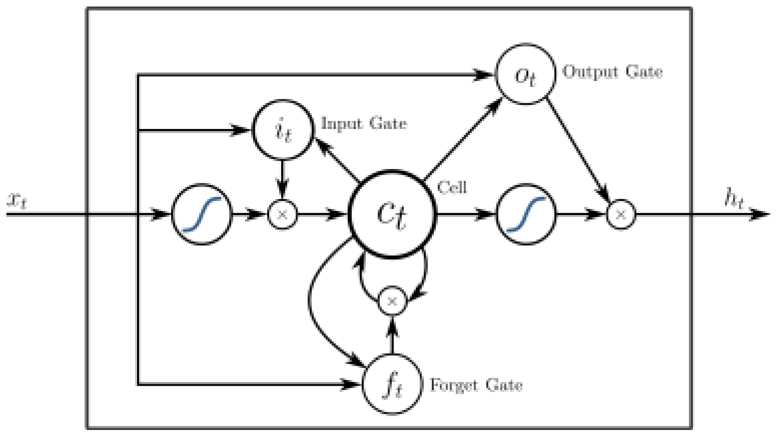


Figure 2.3: LSTM Architecture (Source: Hochreiter et. al., 1997)

Other variants of LSTM such as CuDNNLSTM (CuDNN is a library for deep neural nets built using CUDA (Compute Unified Device Architecture), NVIDIA's API for programming on the graphics card) also exist, which run on GPU only, with the TensorFlow backend.

Gated Recurrent Unit (GRU)

GRU is a gating mechanism for RNN which was introduced by J. Chung et al. and K. Cho et al. GRU is a simplified variant of the LSTM architecture, but there are differences as in figure 2.4: GRU contains two gates and does not possess any internal memory and finally, a second non-linearity is not applied (*tanh* in figure).

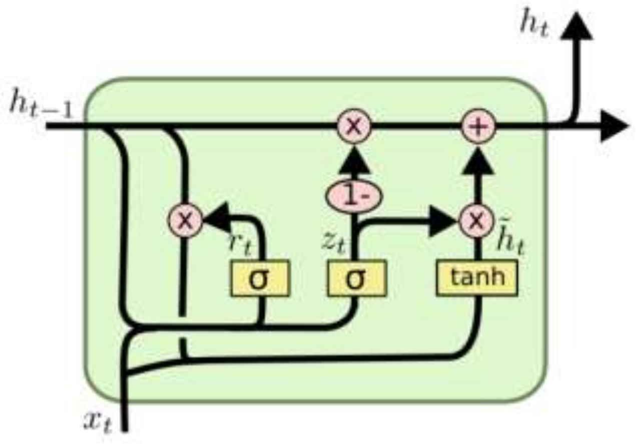


Figure 2.4: Gated Recurrent Unit (GRU) Architecture

(Source: Cho et. al., 2014)

Similarly, other variants of GRU, such as CuDNNGRU also exist, which run on GPU only, with the TensorFlow backend.

Some of the previous works perform comparison of these different architectures with different results that an architecture outperforms other in different datasets, which is discussed in the following sections (chapters).

3. LITERATURE REVIEW

3.1 Hate speech detection systems

Social Media data constitutes of what is known as big data. Given the relatively high prevalence of offensive language and “curse words” on social media, this makes hate speech detection particularly challenging (Wang et al. 2014). Most of the recent methods primarily present the problem as a supervised text classification task which can be divided into two categories (Zhang et al. 2018): Classical methods that rely on feature engineering such as Naïve Bayes, SVM, Logistic Regression and Decision Trees; and deep learning-based methods that make use of neural networks to automatically learn multi-layers of abstract features from raw data. Lexical-based methods only identify potentially offensive terms but are inaccurate at identifying hate speech [5].

3.2 Classical methods

These methods are based on the manual conversion or encoding of data features into feature vectors, which are then used by the classifiers. Simple features such as n-grams, bag-of-words and POS vectors. Opinion mining makes use of the degree of polarity as expressed by the opinion holder. Among the classifiers, however Logistic Regression, SVMs, Naïve Bayes and Random Forests are most popular.

3.3 Deep Learning - based methods

These are powerful set of learning techniques in neural networks. Deep learning mean using more layers. This does not just apply to neural networks, the layers of other artificial intelligence methods can also be stacked, but this work generally focuses on neural networks. Neural networks and deep learning currently provide the best solutions to many problems in image recognition, speech recognition, and natural language processing. Using deep layers, they can learn abstract feature representations through their multiple stacked layers for classification of text. The input can vary from raw data to various feature encodings including those used in classical methods. The most popular network architectures are CNN (Convolutional Neural Networks) and RNN (Recurrent Neural Networks).

Feature based approaches have been leveraged to better identify the intensity of hate speech, most works focus on detecting profanity, using features such as n-grams, bag of words, TF-IDF and part-of-speech features (Davidson et al. 2017, Wafula et al. 2016). Many studies suggest that there are good and robust ways to identify abusive languages. George et al. (2018) present a list of criteria based in critical race theory to identify racist and sexist slurs.

3.4 Hate Speech Opinion Mining

Over the past years, interest in online hate speech detection and particularly the automatization of this task has continuously grown, along with the societal impact of the phenomenon. Opinion Mining has been handled as a Natural Language Processing task at many levels of granularity. A number of methods used for hate speech detection have been documented by the sources cited in this proposal.

Context-driven Sentiment Analysis scheme are also present (Sharma et. al [6]) with the objective of refining the degree of subjectivity during Sentiment Analysis. Ceron, et. al. [7] presents an algorithm, named iSA (integrated Sentiment Analysis) for social networks and Web 2.0 sphere (Twitter, blogs) opinion analysis. Instead of performing an individual classification and then aggregate the predicted values, the algorithm directly estimates the aggregated distribution of opinions. Various social media contents-based sentiment analysis and prediction system exist that detect events in real time out of the massive social media contents (Yoo et. al. [8]). In these systems, users' sentiments are classified using deep learning methods such as Convolutional Neural Networks.

Various systems such as ontology-based sentiment analysis systems (Thakor and Sasi [9]) where tweets with negative sentiments only are used for polarity detection. In these sysems sentiment is detected from the built ontology. Positive polarity corresponds to positive sentiment and negative polarity correspond to the negative sentiment.

Recent methods make use of new feature extraction techniques based on word embeddings. This kind of representation makes it possible for words with similar meaning to have a similar representation, which can improve the performance of classifiers.

Deep learning-based ensemble model for intent detection of spoken language understanding are also developed, (Ekbal et. al. [10]) which primarily focus on extraction of intended meaning. Various deep learning architectures such as CNN and variants of recurrent neural networks (RNN) like LSTM and multi-layer perceptron (MLP) are combined. Ensemble learning are also applied to NLP tasks, e.g. POS tagging, chunking and word sense disambiguation. Word embeddings such as Glove and Word2vec were combined in this approach.

Some of the previous works perform comparison of these different approaches (deep learning architectures) with different results such as an approach outperforms other in different datasets. Example, for sentiment analysis tasks, (Mekolov et al. 2013) found that in deep learning approaches GRU outperforms LSTM and CNN in sentiment analysis of Russian tweets. Although CNNs are also considered good at extracting local and position-invariant features (Yin et al. 2018), they are outperformed by RNN in similar sentiment analysis problems because GRUs are better when sentiment is determined by the entire sentence or a long-range semantic dependency.

However, the performance of these systems cannot be determined just alone by the output without varying the hyperparameters. For example, variations in batch sizes and hidden size could cause large fluctuations. Some authors also argue that the use of a linguistic characteristic with the use of BI-LSTM based on the occurrence of hateful words in data could also help improve the performance of the model (Sarracen et al. 2018).

4. RESEARCH METHODOLOGY

4.1 System Block Diagram

The methodology implemented in this work is machine learning-based and has followed the workflow as shown in figure 4.1.

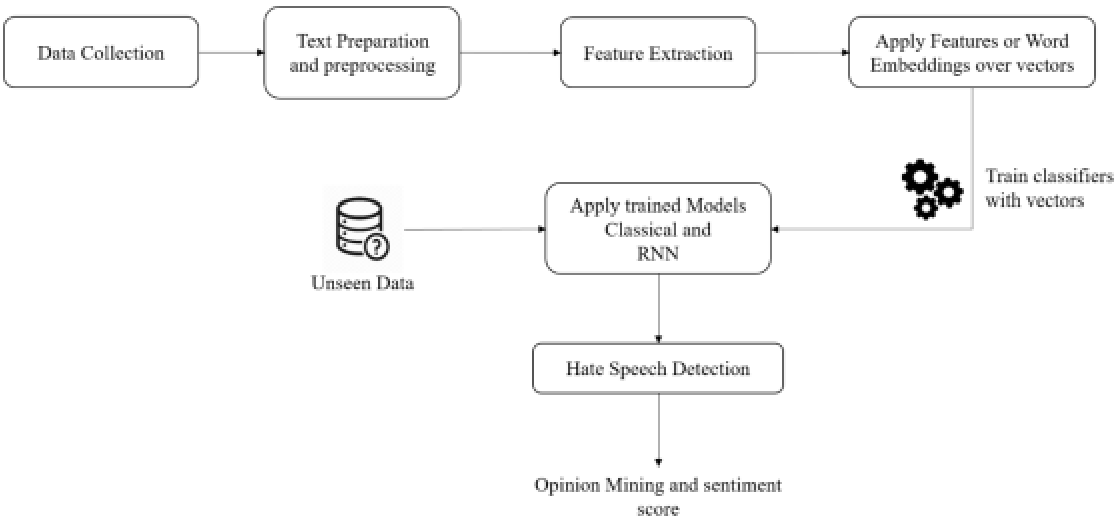

Data Collection
Text Preparation and preprocessing
Feature Extraction
Apply Features or Word Embeddings over vectors
Train classifiers with vectors
Apply trained Models Classical and RNN
Unseen Data
Hate Speech Detection
Opinion Mining and sentiment score

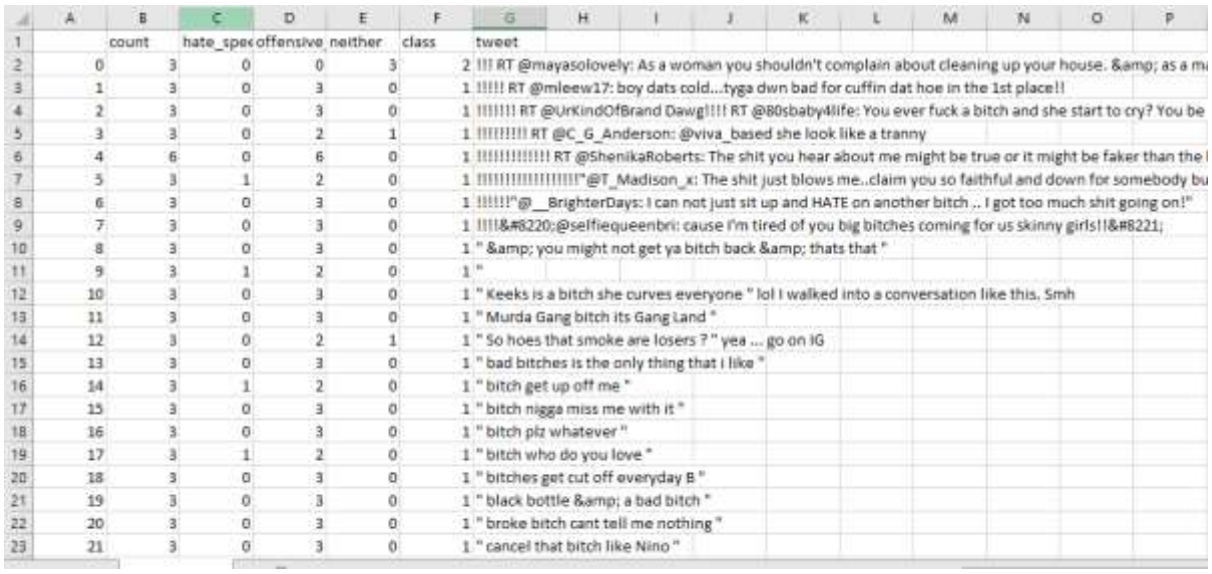
Figure 4.1: System Block diagram

The methodology in this system consists of the following steps:

4.2 Data Collection and Sources

The dataset used in this approach is a dataset compiled by hataebase.org, that creates a multilingual dictionary of words used in hate speech (phrases identified by internet users as hate speech) and consists of a sample of 3k tweets containing the hate speech lexicon. The Tweets are labeled as ‘hate’, ‘offensive’ and ‘neither.’ The dataset is classified based on crowdsourcing by the online users’ community. For testing the model, the streaming Twitter API and the scraped tweets were used.

Figure 4.2 illustrates a sample of the labeled dataset.



	A	B	C	D	E	F	G
1		count	hate_spe	offensive	neither	class	tweet
2	0	3	0	0	3	2	!!! RT @mayasolovely: As a woman you shouldn't complain about cleaning up your house. &amp; as a m
3	1	3	0	3	0	1	!!!!! RT @mleew17: boy dats cold...tyga dwn bad for cuffin dat hoe in the 1st place!!
4	2	3	0	3	0	1	!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby4life: You ever fuck a bitch and she start to cry? You be
5	3	3	0	2	1	1	!!!!!!!!! RT @C_G_Anderson: @viva_based she look like a tranny
6	4	6	0	6	0	1	!!!!!!!!!!!!! RT @ShenikaRoberts: The shit you hear about me might be true or it might be faker than the
7	5	3	1	2	0	1	!!!!!!!!!!!!!!!!!!"@T_Madison_x: The shit just blows me..claim you so faithful and down for somebody bu
8	6	3	0	3	0	1	!!!!!!"@__BrighterDays: I can not just sit up and HATE on another bitch .. I got too much shit going on!"
9	7	3	0	3	0	1	!!!!&#8220;@selfiequeenbri: cause I'm tired of you big bitches coming for us skinny girls!!&#8221;
10	8	3	0	3	0	1	" &amp; you might not get ya bitch back &amp; thats that "
11	9	3	1	2	0	1	"
12	10	3	0	3	0	1	" Keeks is a bitch she curves everyone " lol I walked into a conversation like this. Smh
13	11	3	0	3	0	1	" Murda Gang bitch its Gang Land "
14	12	3	0	2	1	1	" So hoes that smoke are losers ? " yea ... go on IG
15	13	3	0	3	0	1	" bad bitches is the only thing that i like "
16	14	3	1	2	0	1	" bitch get up off me "
17	15	3	0	3	0	1	" bitch nigga miss me with it "
18	16	3	0	3	0	1	" bitch plz whatever "
19	17	3	1	2	0	1	" bitch who do you love "
20	18	3	0	3	0	1	" bitches get cut off everyday B "
21	19	3	0	3	0	1	" black bottle &amp; a bad bitch "
22	20	3	0	3	0	1	" broke bitch cant tell me nothing "
23	21	3	0	3	0	1	" cancel that bitch like Nino "

Figure 4.2: A sample of the labeled tweets dataset

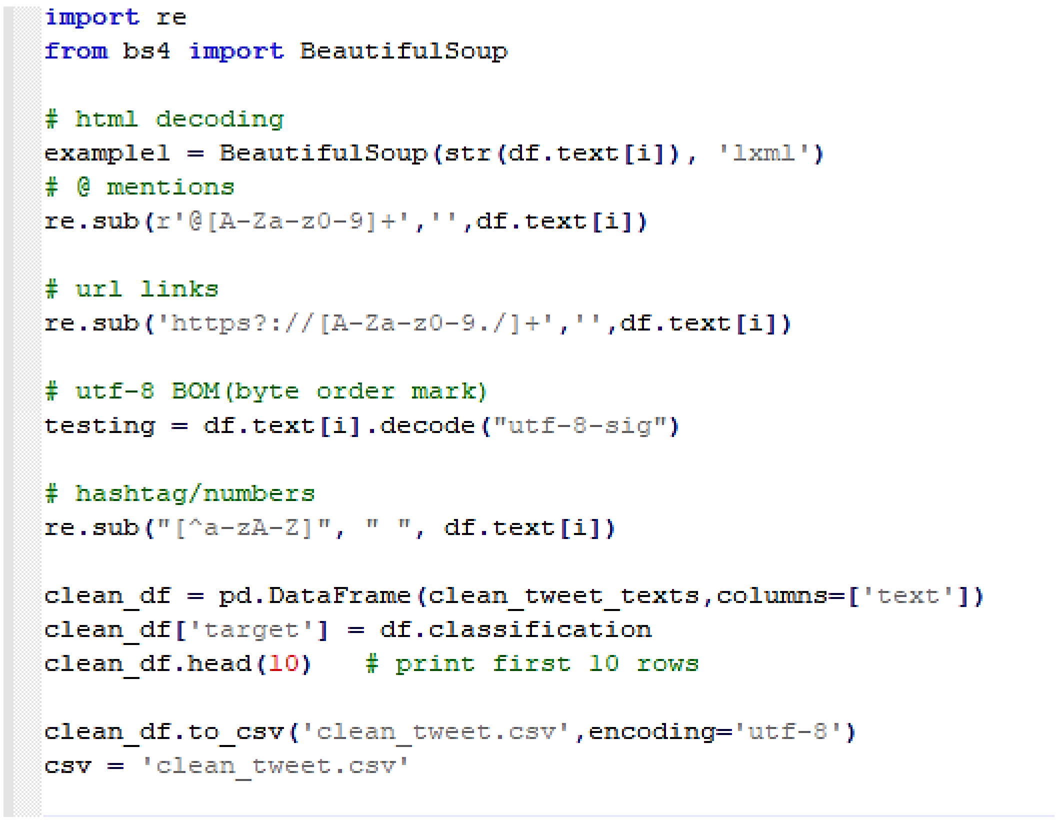
4.3 Corpus Collection

The corpus collection begins with carrying out an initial search of common slangs and terms used in social media such as twitter. Based on this sample, the public Twitter search API was used to collect the entire corpus containing tweets written in English, ensuring that non-offensive and potentially offensive words were also included in the tweets. For example, even though 'trash' is one of the most frequent words in hate or offensive tweets, it also occurs in neither of the category, such as "*Power has been trash for a while now and I decided that I'm not gonna watch this season. My wife can tell me how it's gonna end.*"

Identification and annotation: Although they can be expressed without any such terms, it is almost easy to identify offensive terms and hate speech. Also, sometimes it is difficult to identify hate speech for humans due to difference in exposure and knowledge of hate speech. The hate speech understanding here is based on the definitions presented in the introduction section.

4.4 Text Preparation and Preprocessing

The text data collected is basically in an unstructured format that is not suitable for machine learning. Since opinions and feelings are expressed in different ways, with different vocabulary, context of writing, usage of short forms and slang, making the data huge and disorganized, this step basically includes identifying and eliminating non-textual content and content that is irrelevant to the area of study from the data. For example, tweets are short, noisy and ungrammatical. Before classifying sentiment of a tweet as positive and negative, good amount of pre-processing has to be done. For this, tweets undergo some or all of the processes of pre-processing as shown in figure 4.3.



```
import re
from bs4 import BeautifulSoup

# html decoding
example1 = BeautifulSoup(str(df.text[i]), 'lxml')
# @ mentions
re.sub(r'@[A-Za-z0-9]+','',df.text[i])

# url links
re.sub('https?://[A-Za-z0-9./]+','',df.text[i])

# utf-8 BOM(byte order mark)
testing = df.text[i].decode("utf-8-sig")

# hashtag/numbers
re.sub("[^a-zA-Z]", " ", df.text[i])

clean_df = pd.DataFrame(clean_tweet_texts,columns=['text'])
clean_df['target'] = df.classification
clean_df.head(10)    # print first 10 rows

clean_df.to_csv('clean_tweet.csv',encoding='utf-8')
csv = 'clean_tweet.csv'
```

Figure 4.3: Cleaning of tweets for preprocessing

a) **Conversion to UTF-8**: Many tweets might contain unusual or non-standard characters, which can be problematic for downstream processing. To address these issues, we can use for example, a combination of *BeautifulSoup* and *Unidecode* to convert and transliterate all tweets to UTF-8.

b) **Removal of Empty (Null) Tweets**: After completing all of the other pre-processing, any empty tweets can be deleted.

c) **Removal of stop words** such as ‘is,’ ‘a,’ ‘the,’ ‘to,’ that don’t contain much information.

The pre-processing of the extracted data and proper training is very important aspect than applying the algorithm and getting the results, which help in improving the accuracy of speech classification. The accuracy of classification also depends on the features obtained such as tf-idf, bag of words and n-grams.

4.5 Feature Extraction from Text

The first step in a machine learning text classifier requires transforming the text into a numerical representation, usually a vector. Usually, each component of the vector represents the frequency of a word or expression in a predefined dictionary (e.g. a lexicon of polarized words). This process is known as feature extraction or text vectorization and the classical approach has been bag-of-words or bag-of-ngrams with their frequency.

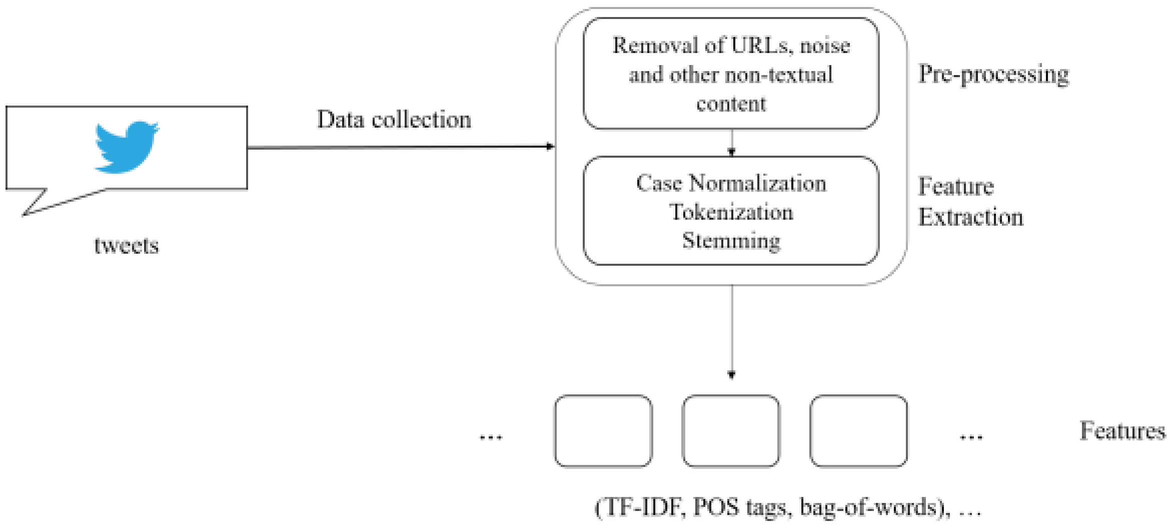


Figure 4.4: The feature extraction process

Figure 4.4 shows the feature extraction process. It includes the given processes:

a) **Case Normalization**: In this step entire document is converted into lowercase.

b) **Tokenization**: Tokenization is splitting up the systems of text into personal terms or tokens.

c) **Stemming (Snowball)**: Stemming is the procedure of decreasing relevant tokens into a single type of token. This procedure contains the recognition and elimination of suffixes, prefixes, and unsuitable pluralization.

d) **Generate features:** Character n-grams are 'n' nearby figures from a given feedback sequence. For example, a 3- gram of a phrase 'FORM' would be '_ _ F', '_FO', 'FOR', 'ORM', 'RM_', 'M_ _'. N-grams of dimension one are known as 'unigram', two dimensional grams are known as 'bigram', three-dimensional grams are known as 'trigram'. And for the rest of the dimensions it is called n-grams. Other features include bag of words, part of speech features and TF-IDF, which are explained as follows.

Part-of-speech Features: Parts-of speech features i.e. nouns, adverbs, adjectives, etc. in each tweet are tagged.

Bag-of-words: A bag-of-words is a representation of text that describes the occurrence of words within a document. As the vocabulary size increases, so does the vector representation of documents. As such, there is pressure to decrease the size of the vocabulary when using a bag-of-words model. There are simple text cleaning techniques that were used as a first step, such as ignoring case, punctuation and frequent words that do not contain much information, called stop words, reducing words to their stem (e.g. "play" from "playing") using stemming algorithms. For this, the tweets are stemmed using Snowball method in the Natural Language Toolkit library (NLTK).

Some of the words appear nearly in each message, and therefore, provide less distinctive information. Therefore, different weights are assigned for each word based on how often they appear in different messages using the Term Frequency — Inverse Document Frequency weighting (TF-IDF). TF-IDF gives higher importance for the words, which are only in few documents (or tweets in this case).

The TF-IDF assigns the weight to a word as shown in equation 4.1:

$$w_{ik} = tf_{ik} \; x \; \log(\frac{N}{n_k}) \text{ …………….. (4.1)}$$

Where,

tf_{ik} = frequency of term T_k in document D_i

N = total no. of documents in the collection C

n_k = the no. of documents in C that contain T_k

T_k = term k in document D_i

idf_{ik} = $\log(\frac{N}{n_k})$ (the inverse document frequency of T_k in C)

Word Embeddings

Word Embedding converts a word to an n-dimensional vector. Related words are mapped to similar n-dimensional vectors, while dissimilar words have dissimilar vectors. In this way the 'meaning' of a word can be reflected in its embedding, a model is then able to use this information to learn the relationship between words. In this case, the *text_to_word_sequence* function from the Keras preprocessing library automatically converts a string to a list of word tokens and at the same time clean the data by removing punctuation and capitalization.

4.6 Model Selection

For identifying the appropriate model, some baseline methods were explored and then the proposed approach was developed. This method uses embeddings generated for text classification and thus used with a classifies as its feature representation. The influence of different features on classification was performed using a variety of models that have been used in prior work, such as Support Vector Classifiers, Naive Bayes and Logistic Regression. In order to pick the most suitable feature, grid search with 5-fold Cross-Validation was performed over all feature set combinations in both the approaches.

Since ML algorithms do not understand text directly, the training data can be transformed into vectors (for e.g. simple but effective methods include bag-of-words and TF-IDF). Next, those vectors were used to train machine learning algorithms (Naive Bayes and Logistic Regression for example). However, the output depends on the dataset size and the extracted information.

Finally, the trained model was used for text classification in new unseen dataset by transforming them into vectors and feeding them to the classifier which then decides whether the tweet data contains potentially hate or offensive speech.

Baseline Method(s): Different representations were experimented for baseline approach:

- TF-IDF (Term Frequency - Inverse Document Frequency): Words are given weight. TF-IDF measures relevance, not frequency (that is, word counts are replaced with TF-IDF scores across the whole dataset).
- Model: Logistic Regression with same configuration as in Davidson (2017).

Proposed Approach: Two different neural net architectures were investigated as explained follows. For each of the two architectures, word embeddings are initialized (Glove embeddings).

- CNN (Convolutional Neural Networks): CNNs with 256 filters, kernel size 2 was used.
- RNN (Recurrent Neural Networks): Specifically, GRU and LSTM were used. Recurrent Neural Network like LSTMs can use their internal memory to process arbitrary sequences of inputs. Hence, long term dependencies can be captured

by LSTMs; which may play a role in hate speech detection. LSTMs with 100 units were used in this experimentation.

Figure 4.5 represents the model summary.

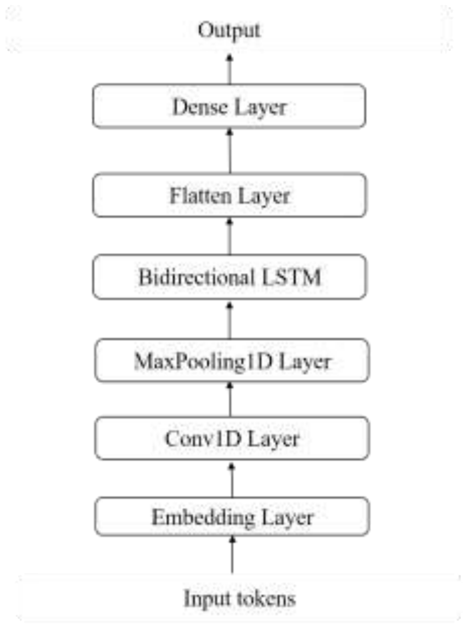


Figure 4.5: Model Block diagram for proposed approach

4.6.1 Optimization and Evaluation

Activation functions: There are many different functions that can be used, the most common ones are: *Sigmoid, tanh, Softmax,* and rectified linear unit (*ReLU*). ReLU and Softmax are the most common functions used. In this classification problem, the softmax activation was chosen.

Loss Function: A few popular loss functions that are currently being used are: mean squared error (MSE), likelihood loss and cross entropy loss. In this case, the categorical cross-entropy loss function is used to obtain the results because it was more effective in previous works.

Callbacks: A callback basically is a set of various functions to be applied in stages of the training procedure. In this case, the EarlyStopping callback is used. Here, the model stops training when certain monitored quantity stops improving over time. For example, the 'validation loss' quantity can be monitored to prevent the model to overfit.

Metric Functions: The four categories: true positive, false positive, false negative and true negative form the basis of the metrics that were used to evaluate the classification model including accuracy, precision, recall and F-Score.

Accuracy measures the percentage of inputs in the test set that the model correctly labelled either as hate speech or non-hate speech. Precision is the ratio of correctly classified documents to the total number of documents classified under a particular category. Recall is defined as the number of correctly classified documents among all documents belonging to that category, whereas F-Score is a harmonized mean of precision and recall.

4.7 Opinion Mining (Sentiment Analysis)

Opinion mining is the automated process of understanding an opinion about a given subject from written or spoken language. From being able to mine opinions from product reviews to being able to forecast stock prices by studying tweets, it has a very wide range of applications. A thorough process of the opinion mining process is described in the next chapter of this work.

The use of a sentiment lexicon designed for social media (Davidson 2017, Hutto and Gilbert 2014) can also help sentiment detection. Table 4.1 shows sentiment analysis of collected data with their polarity (Negative, Neutral and Positive).

Table 4.1: Sentiment Analysis Example

Tweet	**Classification**	**Polarity**
I love being a dizzy h*e for social media	Offensive	1 (Positive)
I want to retweet it 1000 times.. I just hate this one roommate of mine. F***ing c*nt.	Hate	-1 (Negative)
Shoppee 9.9 sale = time to stock up on basic hoe necessities coz IT'S BRITNEY, B**CH.	Offensive	0 (Neutral)

4.8 Tools Used

Programming Language	Python 3.6 +
IDE	Jupyter Notebook, Notepad++
ML library	Keras, sklearn, tensorflow

The hardware specifications used are Graphics GPU accelerator Nvidia Geforce MX and RAM of 12 GB.

5. RESULT, ANALYSIS AND COMPARISON

5.1 Dataset and Experimental Settings

As explained earlier, TF-IDF was used for baseline and GloVe pre-trained embeddings was used for word embedding based approach. A 5-fold Cross Validation was performed and weighted metrics were used for precision, recall and F1-scores. For deep learning method, the 'Adam' optimizer was used. Batch size of 128 was used. Also, because of the unbalanced number of labelled classes, experimentation was also done with taking the equal number of the class instances.

Word embeddings: GloVe pre-trained word embeddings was experimented. It was trained on a corpus of 6 billion tokens and contains 200 embedding dimensions. On glove embeddings, the training is basically performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space (Source: Stanford GloVe).

5.2 Results and Analysis

The classifier was assigned a batch size of 128, epoch number of 10 and optimizer adam. Table 5.1 shows the results of different approaches applied to the detection of hate speech. The first row shows various results for Baseline Method and the second row shows the implementation of the deep learning models.

Table 5.1: Experimental Results

	Class	Precision	Recall	F1-Score
Baseline Method	Hate	0.48	0.78	0.59
	Offensive	0.88	0.67	0.76
	Neither	0.93	0.86	0.89
Deep learning Methods	Hate	0.57	0.80	0.66
	Offensive	0.87	0.71	0.78
	Neither	0.92	0.86	0.88

The final model shows overall precision of 0.82, recall 0.78 and F1 score of 0.80 and individually, as shown in the table for hate, offensive and neither classes, thus proposed methods prove to be significantly better than the baseline methods. RNNs (particularly

LSTMs) and CNNs performed slightly better than classical methods in the proposed approach. The use of word embeddings also helped in the classification.

The confusion matrix in table 5.2 illustrates the output using various approaches.

Table 5.2 (a): Confusion Matrix using baseline method

		Predicted		
		Hate	Offensive	Neither
Actual	Hate	0.47	0.40	0.11
	Offensive	0.09	0.87	0.03
	Neither	0.04	0.02	0.93

Table 5.2 (b): Confusion Matrix using RNN

		Predicted		
		Hate	Offensive	Neither
Actual	Hate	0.59	0.21	0.20
	Offensive	0.05	0.82	0.13
	Neither	0.01	0.02	0.96

Table 5.2 (c): Confusion Matrix using equal instances of all classes (baseline)

		Predicted		
		Hate	Offensive	Neither
Actual	Hate	0.56	0.32	0.10
	Offensive	0.09	0.86	0.04
	Neither	0.05	0.02	0.92

Table 5.2 (d): Confusion Matrix using equal instances of all classes (using RNN)

		Predicted		
		Hate	Offensive	Neither
Actual	Hate	0.74	0.13	0.12
	Offensive	0.09	0.78	0.12

	Neither	0.07	0.12	0.81

ROC Plots

Figure 5.1 shows ROC curve for the different classes.

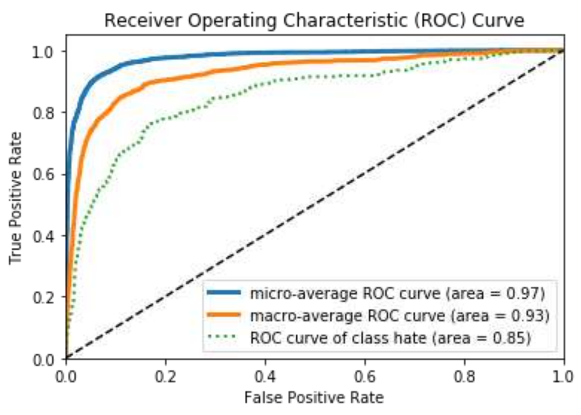


Figure 5.1 (a): ROC of class *hate*

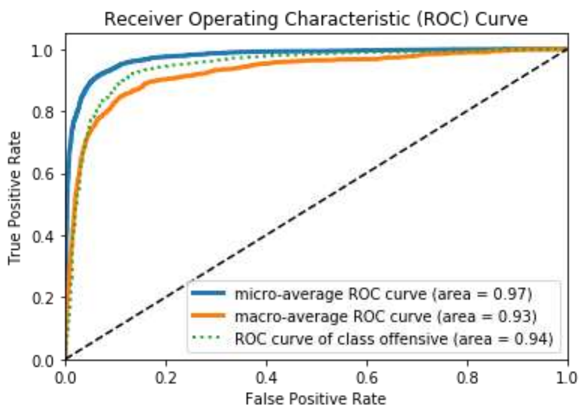


Figure 5.1 (b): ROC of class *offensive*

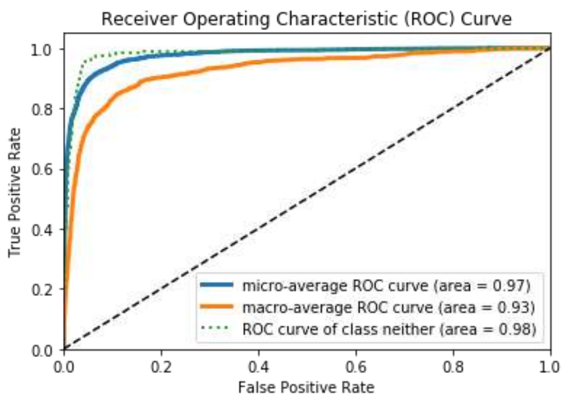


Figure 5.1 (c): ROC of class *neither*

Opinion Mining

After preprocessing and a close analysis of the top positive and negative keywords in the dataset (for simplicity, most frequent ‘hateful’ and non-hateful keywords were identified from the labeled dataset, using the positive and hateful lexicon from hatebase.org), it results in the following list of top 50 keywords as shown in figure 5.2.

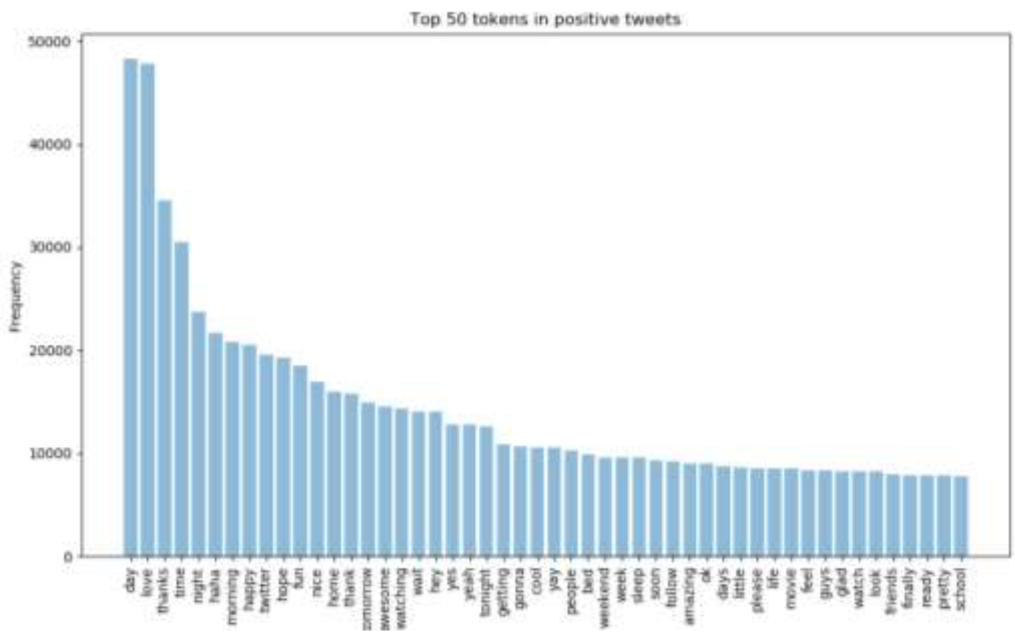


Figure 5.2 (a): Top 50 positive tokens

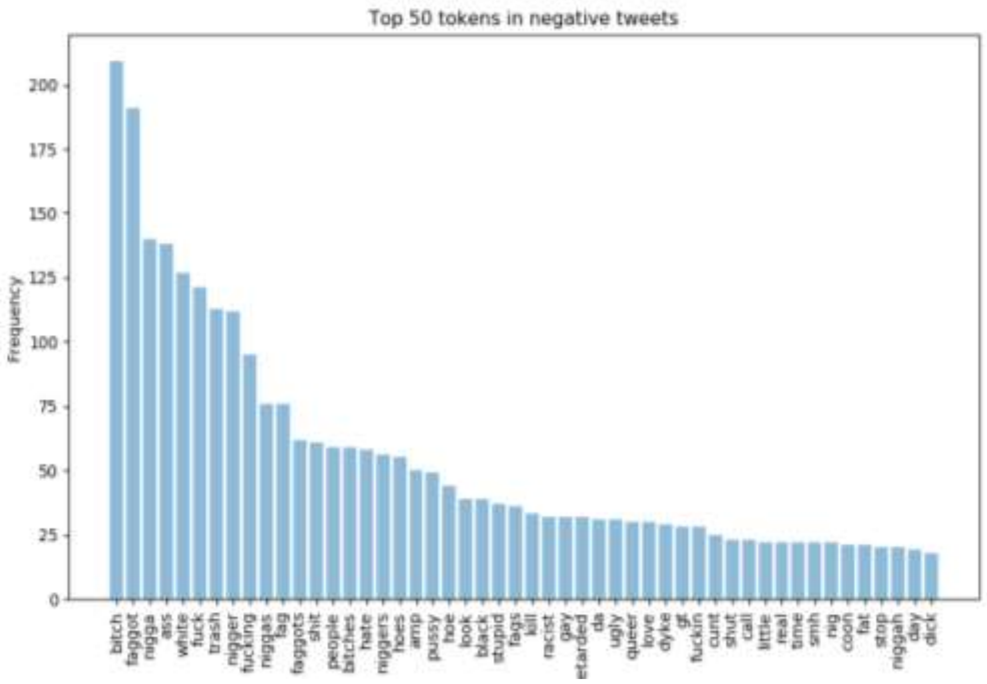
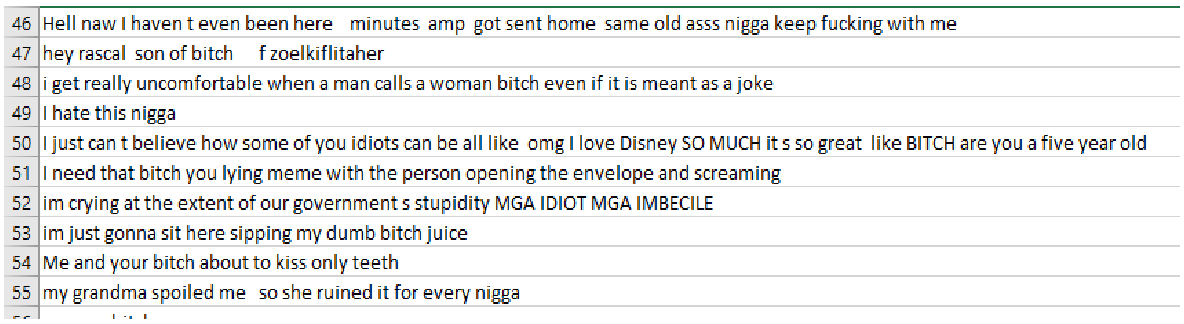


Figure 5.2 (b): Top 50 negative tokens

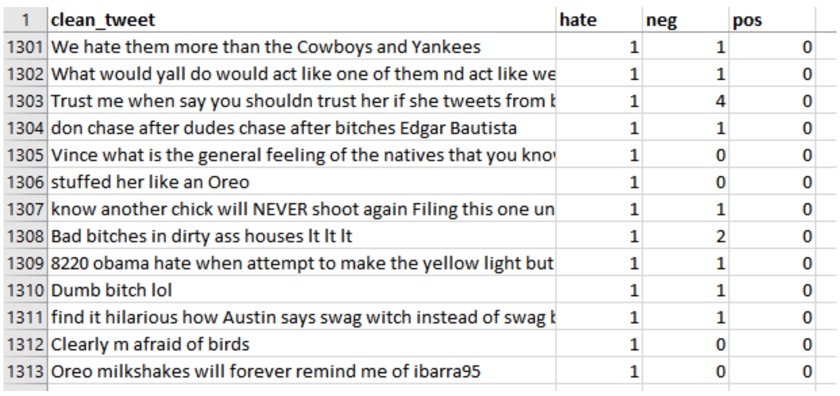
In the next step, tweets were collected over different time containing these particular terms and apply the sentiment analysis model upon the collected data (The streaming twitter API could also be used for data collection in real-time). For this, tweets were randomly collected over a period of two months. Figure 5.3 shows a sample of tweets containing particular terms.



46	Hell naw I haven t even been here minutes amp got sent home same old asss nigga keep fucking with me
47	hey rascal son of bitch f zoelkiflitaher
48	i get really uncomfortable when a man calls a woman bitch even if it is meant as a joke
49	I hate this nigga
50	I just can t believe how some of you idiots can be all like omg I love Disney SO MUCH it s so great like BITCH are you a five year old
51	I need that bitch you lying meme with the person opening the envelope and screaming
52	im crying at the extent of our government s stupidity MGA IDIOT MGA IMBECILE
53	im just gonna sit here sipping my dumb bitch juice
54	Me and your bitch about to kiss only teeth
55	my grandma spoiled me so she ruined it for every nigga

Figure 5.3: A sample of tweets with particular terms

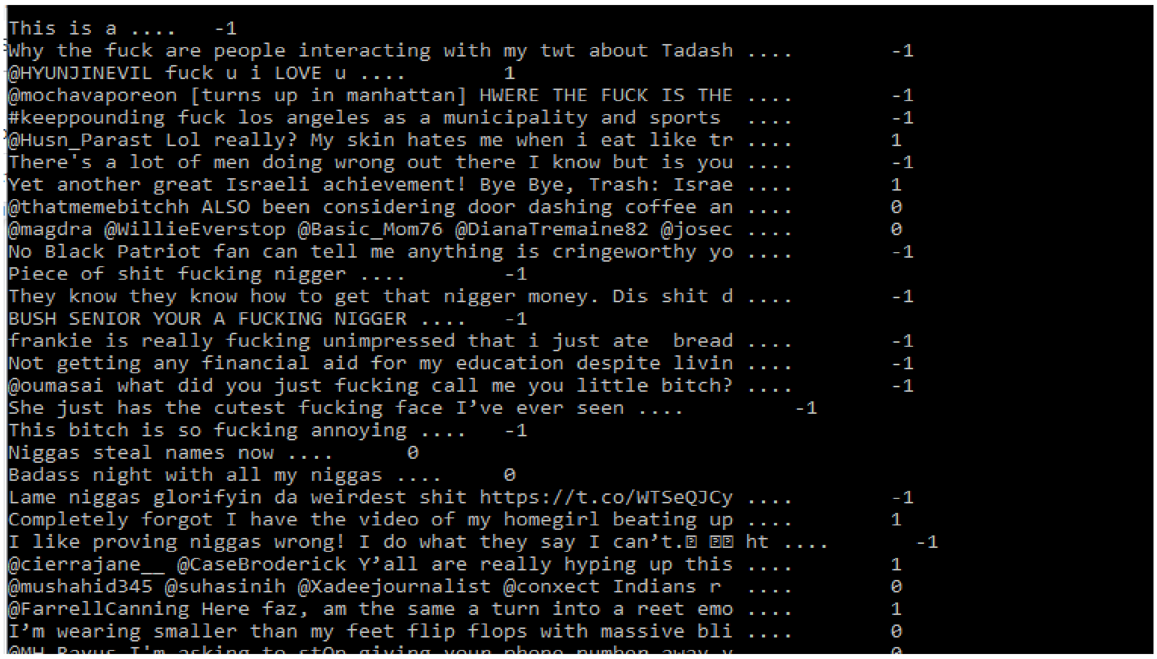
Moreover, figure 5.4 shows the use of sentiment weighing for the tweets collected and preprocessed earlier.



1	clean_tweet	hate	neg	pos
1301	We hate them more than the Cowboys and Yankees	1	1	0
1302	What would yall do would act like one of them nd act like we	1	1	0
1303	Trust me when say you shouldn trust her if she tweets from t	1	4	0
1304	don chase after dudes chase after bitches Edgar Bautista	1	1	0
1305	Vince what is the general feeling of the natives that you kno	1	0	0
1306	stuffed her like an Oreo	1	0	0
1307	know another chick will NEVER shoot again Filing this one un	1	1	0
1308	Bad bitches in dirty ass houses lt lt lt	1	2	0
1309	8220 obama hate when attempt to make the yellow light but	1	1	0
1310	Dumb bitch lol	1	1	0
1311	find it hilarious how Austin says swag witch instead of swag b	1	1	0
1312	Clearly m afraid of birds	1	0	0
1313	Oreo milkshakes will forever remind me of ibarra95	1	0	0

Figure 5.4: Sentiment weighing for opinion mining

These tweets then undergo all the preprocessing steps mentioned earlier. After this, the opinion mining model is applied on to these tweets with their target class prediction. The polarity sentiment is denoted by positive and negative values (for example, 1 for positive and -1 for negative polarity). A summary of the prediction is shown in figure 5.5. Although the polarity shows only negative or positive instances, it does not mean that an offensive speech is negative. As some tweets with negative words could also express positive meaning (e.g. f##ing awesome), the positive lexicon is also considered for more reliable prediction.



```
This is a ....   -1
Why the fuck are people interacting with my twt about Tadash ....        -1
@HYUNJINEVIL fuck u i LOVE u ....        1
@mochavaporeon [turns up in manhattan] HWERE THE FUCK IS THE ....        -1
#keeppounding fuck los angeles as a municipality and sports  ....        -1
@Husn_Parast Lol really? My skin hates me when i eat like tr ....        1
There's a lot of men doing wrong out there I know but is you ....        -1
Yet another great Israeli achievement! Bye Bye, Trash: Israe ....        1
@thatmemebitchh ALSO been considering door dashing coffee an ....        0
@magdra @WillieEverstop @Basic_Mom76 @DianaTremaine82 @josec ....        0
No Black Patriot fan can tell me anything is cringeworthy yo ....        -1
Piece of shit fucking nigger ....        -1
They know they know how to get that nigger money. Dis shit d ....        -1
BUSH SENIOR YOUR A FUCKING NIGGER ....   -1
frankie is really fucking unimpressed that i just ate  bread ....        -1
Not getting any financial aid for my education despite livin ....        -1
@oumasai what did you just fucking call me you little bitch? ....        -1
She just has the cutest fucking face I’ve ever seen ....        -1
This bitch is so fucking annoying ....   -1
Niggas steal names now ....      0
Badass night with all my niggas ....     0
Lame niggas glorifyin da weirdest shit https://t.co/WTSeQJCy ....        -1
Completely forgot I have the video of my homegirl beating up ....        1
I like proving niggas wrong! I do what they say I can’t.� �� ht ....        -1
@cierrajane__ @CaseBroderick Y’all are really hyping up this ....        1
@mushahid345 @suhasinih @Xadeejournalist @conxect Indians r  ....        0
@FarrellCanning Here faz, am the same a turn into a reet emo ....        1
I’m wearing smaller than my feet flip flops with massive bli ....        0
@MH_Rayus I'm asking to stOp giving your phone number away y             0
```

Figure 5.5: Opinion Mining for the tweets

However, assuming that both the lexicons have same frequency, a tweet with short length could show more sentiment than one with a longer length. This is why the normalization of positive and negative term frequency is to be considered for more reliable sentiment analysis, as shown by the scores in Table 5.3. The sentiment scores are on the scale of 1-10.

Table 5.3: Sentiment scores of tweets

Tweet	**Classification**	**Sentiment Scores**
Yes you do r*tard	Hate	Negative: 0 Positive: 0
love when girls pass up the guys who want Something real and to treat them right for the guy who just uses em cuz then they b*tch about it	Offensive	Negative: 3 Positive: 2
Roach happy birthday ni**a do coon shit in responsible way	Hate	Negative: 2 Positive: 1

Table 5.4 illustrates the use of normalization for given tweets. Sentiment scores are on 1-10 Scale and the normalized score is on the 0-1 scale.

Table 5.4: Sentiment scores with normalization

Tweet	**Classification**	**Sentiment Scores**	**Scores with Normalization**
All h*es lie	Offensive	Negative: 1 Positive: 0	Negative: 0.33 Positive: 0
These h*e a** ni**as talkn shyt about ni**a make ni**a wanna go and snatch the b**ch up outta ni**a	Offensive	Negative: 3 Positive: 0	Negative: 0.15 Positive: 0
haha yes ni**ah but yesterday night Idk man it hit me at night like around in the morning didn go to sleep till like	Hate	Negative: 6 Positive: 1	Negative: 0.25 Positive: 0.04
happy birthday you dirty little bird tear it up	Neither	Negative: 1 Positive: 4	Negative: 0.07 Positive: 0.14

5.3 Comparison

This thesis work uses the word embedding models for creating word vectors and deep learning architectures such as long short term memory for text classification of the tweet data. Different standard datasets are used for experiment of the classification procedure. The precision, recall and F1 scores of the proposed method in classifying the text are 0.82, 0.78 and 0.80 in case of classes hate, offensive and none, which suggests a noticeable improvement over the baseline method. The abusive language was taken as the combination of hateful as well as offensive tweets. The ROC curve also suggests an acceptable degree of classification among the classes. Moreover, the accuracy of hate speech can be increased if the instances of hateful languages are increased.

6. CONCLUSION AND RECOMMENDATION

6.1 Conclusion

The application of deep learning models has helped better identify hate speech from the given data. A model for recognizing the hate speech has been developed and the performance of the proposed method was demonstrated on different number of various classes. However, considering the fact that offensive language sometimes contributes to the negative sentiment, the prediction of offensive language as hate does improve the sentiment analysis model. Also, through the use of various feature of tweets, such as TF-IDF, Part-of-Speech, and other features such as average syllables, sentiment scores, number of characters, etc. as proposed by authors of [11] also helps in the distinction of hate speech from offensive language, however that is not always the case. For sentiment analysis, detection of offensive language as hate is ignored as both the class contribute to negative sentiment in most cases. The use of deep learning approaches such as Recurrent Neural Networks has proven to improve the classification.

6.2 Limitations

From the experimental results, the proposed method proves to have a significant performance over classic methods, however this method is limited to English language only and since hate speech is a subjective matter, the classification might be a challenging problem because what might be offensive to a person, might be hateful to others. This is true especially with people from different communities and different age groups. Also, due to class imbalance on the available dataset, there is no particular distinction on some instances of hate and offensive languages.

6.3 Recommendation

The use of deep learning methods has shown to increase the detection of hate speech and abusive language on the datasets however, given sufficient number of minority class, the precision can be increased even further. Also, hate speech being subjective matter, there is no particular distinction on hate and offensive languages. The use of deep learning architectures such as LSTMs and GRUs have helped on overall abusive language identification, however the differentiation on such features depends on the actual dataset size, class instances and the features used.

7. REFERENCES

[1] Kherwa, P., Sachdeva, A., Mahajan, D., Pande, N., & Singh, P. K. (2014). An approach towards comprehensive sentimental data analysis and opinion mining.

[2] Wafula, & W, G. (2016). Social Media Forensics for Hate Speech Opinion Mining.

[3] Schmidt, A., & Wiegand, M. (2017). A Survey on Hate Speech Detection using Natural Language Processing. Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media.

[4] Mugambi, S. K. (2017). Sentiment analysis for hate speech detection on social media: TF-IDF weighted N-Grams based approach.

[5] Davidson, T., Warmsley, D., Macy, M.W., & Weber, I. (2017). Automated Hate Speech Detection and the Problem of Offensive Language. ICWSM.

[6] Sharma, S., Chakraverty, S., Sharma, A., & Kaur, J. (2017). A context-based algorithm for sentiment analysis. International Journal of Computational Vision and Robotics, 7(5), 558.

[7] Ceron, A., Curini, L., & Iacus, S. M. (2016). ISA: A fast, scalable and accurate algorithm for sentiment analysis of social media content. Information Sciences, 367-368, 105-124.

[8] Yoo, S., Song, J., & Jeong, O. (2018). Social media contents based sentiment analysis and prediction system. Expert Systems with Applications, 105, 102-111.

[9] Thakor, P., & Sasi, S. (2015). Ontology-based Sentiment Analysis Process for Social Media Content. Procedia Computer Science, 53, 199-207.

[10] Firdaus, M., Bhatnagar, S., Ekbal, A., & Bhattacharyya, P. (2018). Intent Detection for Spoken Language Understanding Using a Deep Ensemble Model. Lecture Notes in Computer Science PRICAI 2018: Trends in Artificial Intelligence, 629-642.

[11] Ghosal, D., Akhtar, M. S., Ekbal, A., & Bhattacharyya, P. (2018). Deep Ensemble Model with the Fusion of Character, Word and Lexicon Level

Information for Emotion and Sentiment Prediction. Neural Information Processing Lecture Notes in Computer Science, 162-174.

[12] Medhat, W., Hassan, A., & Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. Ain Shams Engineering Journal, 5(4), 1093-1113.

[13] Go, A. (2009). Twitter Sentiment Classification using Distant Supervision.

[14] K. Paramesha (2016). A Perspective on Sentiment Analysis. Proceedings of 'Second International Conference on Emerging Research in Computing, Information, Communication and Applications.'

[15] Liu, B., & Zhang, L. (2012). A Survey of Opinion Mining and Sentiment Analysis. Mining Text Data, 415-463.

[16] Soni, V., & Patel, M. R. (2014). Unsupervised Opinion Mining From Text Reviews Using SentiWordNet. International Journal of Computer Trends and Technology, 11(5), 234-238.

[17] Ghosal, D., Akhtar, M.S., Chauhan, D., Poria, S., Ekbal, A., & Bhattacharyya, P. (2018). Contextual Inter-modal Attention for Multi-modal Sentiment Analysis. EMNLP.

[18] Joshi, M., Prajapati, P., Shaikh, A., & Vala, V. (2017). A Survey on Sentiment Analysis. International Journal of Computer Applications, 163(6), 34-38.

[19] Pinkesh B., Shashank G., Manish G., and Vasudeva V. (2017). Deep learning for hate speech detection in tweets. In Proceedings of the 26th International Conference on World Wide Web Companion. pages 759–760.

[20] Gitari, Njagi Dennis, et al. "A Lexicon-Based Approach for Hate Speech Detection (2015)." *International Journal of Multimedia and Ubiquitous Engineering*, vol. 10.

[21] Ghosal, Deepanway, et al. "Deep Ensemble Model with the Fusion of Character, Word and Lexicon Level Information for Emotion and Sentiment Prediction (2018)." *Neural Information Processing Lecture Notes in Computer Science*.

APPENDIX

Extract (TF-IDF) features from text

```
# Load file
embedding_file = "data/cleaned_tweets.csv"

cols = ['tweet', 'classifi','classification']
df = pd.read_csv(embedding_file, names=cols, encoding='ut
f-8')

try:
    df=df.dropna()
except:
    df=df.replace(np.nan, '', regex=True)

X = np.array(df.tweet)
y = np.array(df['classifi'].astype(int))


# Feature Extraction
from sklearn.model_selection import StratifiedKFold
stratified_split = StratifiedKFold(n_splits=5,
random_state=42)
for train_index, test_index in stratified_split.split(X,
y):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]

from sklearn.feature_extraction.text import TfidfVectoriz
er
# build TFIDF features on train and test data
tv = TfidfVectorizer(min_df=5, max_df=0.75, ngram_range=(
        1,3), use_idf=True,stop_words='english',
        smooth_idf=False, decode_error='replace',
        sublinear_tf=True)
tv_train_features = tv.fit_transform(X_train)
tv_test_features = tv.transform(X_test)

print(tv_train_features.shape, y_train.shape)
print(tv_test_features.shape, y_test.shape)
```

Convert text data to Word Embeddings

```
# convert text to word embedding (Using GloVe):
def loadData_Tokenizer(X_train, X_test,
        MAX_NB_WORDS=75000, MAX_SEQUENCE_LENGTH=1000):
    np.random.seed(42)
    text = np.concatenate((X_train, X_test), axis=0)
    text = np.array(text)
    tokenizer = Tokenizer(num_words=MAX_NB_WORDS)
    tokenizer.fit_on_texts(text)
    sequences = tokenizer.texts_to_sequences(text)
    word_index = tokenizer.word_index
    text = pad_sequences(sequences,
maxlen=MAX_SEQUENCE_LENGTH)
    print ('Found %s unique tokens.' % len(word_index))
    indices = np.arange(text.shape[0])
    # np.random.shuffle(indices)
    text = text[indices]
    #print(text.shape)
    X_train = text[0:len(X_train), ]
    X_test = text[len(X_train):, ]
    embeddings_index = {}
    f = open("data/glove.6B.200d.txt", encoding="utf8")
    for line in f:
        values = line.split()
        word = values[0]
        try:
            coefs = np.asarray(values[1:],
dtype='float32')
        except:
            pass
        embeddings_index[word] = coefs
    f.close()
    print ('Total %s word vectors.' %
len(embeddings_index))
    return (X_train, X_test, word_index,
embeddings_index)
```

Build Model

```
# run Model and see our results
model_RNN = build_Model(64, 100)

from keras.callbacks import EarlyStopping
early_stopping = EarlyStopping(monitor='val_loss', mode='
        min',patience=5, baseline=0.9, restore_best_weight
        s=True)
history = model_RNN.fit(tv_train, Y_train,
                    validation_data=(tv_test, Y_test),
                    epochs=15,batch_size=64,
                    verbose=2,callbacks=[early_stopping])
predicted = model_RNN.predict(tv_test)
predicted = np.argmax(predicted, axis=1)
```