CYBERBULLYING DETECTION USING SENTIMENT ANALYSIS IN SOCIAL MEDIA

Supervisor: Moin Mostakim

Submitted by:

Mifta Sintaha - 13101123

Shahed Bin Satter - 13101258

Niamat Zawad - 13101283

Chaity Swarnaker - 13101290

Ahanaf Hassan - 13101002

Department of Computer Science & Engineering, BRAC University

Submitted on : 18th August, 2016

Declaration

We, hereby declare that this thesis is based on the results found by ourselves. Materials of work found by other researcher are mentioned by reference. This Thesis, neither in whole or in part, has been previously submitted for any degree.

Signature of Supervisor

Moin Mostakim

Signature of Author

Mifta Sintaha

Signature of Author

Shahed Bin Satter

Signature of Author

Niamat Zawad

Signature of Author

Chaity Swarnaker

Signature of Author

Ahanaf Hassan

Acknowledgement

This is the work of Mifta Sintaha, Shahed Bin Satter, Niamat Zawad, Chaity Swarnaker, and Ahnaf Hassan - students of the CSE department of BRAC University. The document has been prepared as an effort to compile the knowledge obtained by us during these four years of education and produce a final thesis which innovatively addresses one of the very urgent issues that's terrorizing our world at the moment.

All thanks to Almighty, the creator and the owner of this universe, the most merciful, beneficent and the most gracious, who provided us guidance, strength and abilities to complete this research. We are especially thankful to Moin Mostakim, our thesis supervisor, for his immense help, guidance and support in completion of our project. We are also thankful to the BRAC University Faculty Staffs of the Computer Science and Engineering, who have been a light of guidance for us in the whole study period at BRAC University, particularly in educating and enhancing our knowledge. Finally, we would like to express our sincere gratefulness to our beloved parents, brothers and sisters for their love and care and our friends for constant support and encouragement.

Table of Contents

List of Figures

Abstract

In this day and age, the usage of Social Media has increased enormously in our daily lives. People like to share their experiences in various social media accounts for their friends to see. Consequently, the possibility and growth of cyber threats have increased as well. To reduce this situation, we try to propose a system that can detect cyber crimes such as fraud, blackmail, spam, impersonation etc. from the social media network Twitter. This type of study can help people to detect early threats and possible criminal activity and the types of accounts to stay alert of in real time thereby creating a more secure social media experience. Our main goal is to compare various sentiment analysis approaches for detecting bullying or threats from social media using three different machine learning algorithms and form a comparison to determine which among the three gives out the highest accuracy in order for us to decide how to detect cyber bullying activity on the Internet and be alert of threats in both the real and virtual world.

Chapter 1: Introduction

1.1 Introduction

In the current era, with the popularity of Web 2.0. people are highly depended on social networking sites. Albeit, these sites offer great opportunities for connecting with other, but they also increase the susceptibility of the young generation to disputable scenarios, such as cyber-victimization. With the recent development of social media, people have adopted new ways of spreading hate speech through sites such as Twitter, Facebook, Myspace which finally lead to *cyber* crime [1]. Day by day the criminals are becoming adhered to technology, often expressing their emotions on the web. Therefore, any form of bullying through the internet can be detected by extracting from microblogs, social media sites and performing sentiment analysis on them.

Social media has become a tool for people to voice their opinions and even to vent their anger. Therefore, internet surveillance can prevent possible life threatening attacks or finding out suspicious profiles or activities. Our main goal was to identify these threatening messages given out by people in the social media using sentiment analysis approach.

Sentiment analysis is generally denoted as techniques used to determine the predisposition of text, usually expressed in free text form. Subjective information in source materials is recognized and extracted by the means of natural language processing, text analysis, and computational linguistics. It is used to determine an author's attitude, with respect to a particular topic or the overall contextual polarity in the text. This is a promising technology which has resulted remarkable interest among academics. The development of research in the field of text analytics has allowed researchers to formulate algorithms and techniques to discover sentiments from free text more efficiently than ever. Detecting cyberbullying from microblogs is an intriguing research topic because, it is important for law enforcement agencies especially the Bangladesh RAB and police which have been lagging behind in terms of technology, to collaborate with the relevant people who are currently working in this area of sentimental analysis so that they can detect the usage of social media in spreading hate messages before crimes are committed. Our work covers all features from mining texts from social media, applying sentiment analysis based on people’s opinions expressed on social media to finally assigning polarity to them as positive, negative or neutral.

We put forward the notion that social networking sites can provide an efficient publish-subscribe messaging pattern to detect expletives in microblogs, and categorize them under different types of threats.

We are working on Twitter as our social network. Twitter is one of the most popular online social networks to date, where users post their opinions in short text called "tweets". Since tweets are typically limited to 140 characters, they are precise and brief, hence public sentiments can be easily explored. Twitter also provides the feature of Retweet (RT), which allows users to share content posted by another user. This aspect will help us know the degree of spread of a hate message.

Twitter is used, primarily, for the following three reasons:

- Users utilize it as a daily chatter, hence are prone to express inner thoughts liberally.
- Type of users range from celebrities, recognized entrepreneurs, politicians to common public, maintaining versatility of source of opinion
- Tweets are precise, hence author's attitude can be straightforwardly investigated.

Also, a filter is designed to extract tweets from countries deemed to be either the most dangerous to the ones considered safest, in order to geographically analyze crime-related tweets and then performing sentiment analysis to identify crime prone zones in nearly real-time.

1.2 Motivation

Bullying, racism, discrimination have the ability to cause depression and sometimes even suicides by deteriorating the victim both mentally and physically. It has adverse impacts not only on the victims but also on those who bully and those who witness bullying. Consequently it increases crimes, mental and physical illness and make the victims isolate themselves. As a response to cyber threats, a number of national and cross-national child protective initiatives (e.g., The Suicide Prevention Centre (http://www.preventiezelfdoding.be/), Child Focus (http://www.childfocus.be/)) have been starting projects over the last few years to increase online child safety. In spite of these efforts, much undesirable or even hurtful content remains online [1].

On an average, 20% to 40% of all teenagers have been mistreated online, as suggested by recent research reports [1]. With appropriate detection of possible harmful messages, successful prevention can be achieved. However, there is a requirement for intelligent systems to identify possible risks automatically, given how the Web is overload with massive information. This is what encouraged us to control bullying by detecting it in different areas so that the people out there can take initiatives to end it. Our system can detect bullying in social media sites like Twitter, Facebook and informs the location where it is occurring. As follows, our system will help people to diminish bullying in the particular locations and help the victims to get rid of it.

1.3 Thesis Outline

It is impossible to find out individuals in the world who are involved in racism or bullying and convey them how much negative results it consumes. Even victims most of the times keep silent out of fear and anxiety. Our system will go through the tweets to learn in which location the amount of bullying is excessive as people express their hatred for a particular group through tweets. As a result people can take initiatives i.e., setting up campaigns against bullying or can communicate with students in different institutions etc. to reduce bullying to some extent. In short, our objective is to make our system efficient enough to generate the desired output and help people to halt violence and crime.

Chapter 2: Problem Statement

2.1 Problem Statement

Formerly, traditional bullying was limited to schools and youth crowds. However, with the growing reputation of social media and its swift embracement into our daily lives cyber bullying has become an emerging problem and it can continue at home. Social media practically exhibits various features that make them a suitable way for cyber-bullies to target their victims. Such as being anonymous, lack of supervision and impact. A recent study revealed that 11% of them had been bullied at least once in the six months among 2,000 Flemish secondary school students preceding the survey [26].

It is evident that online platforms are increasingly used for bullying and that cyberbullying is thus not a rare problem. Moreover, it poses a substantial threat to a teenager's mental and physical health, suggested by studies linking cyber bullying to depression, low self-esteem and school problems [27], [28], [29]. In extreme cases, its consequences have been closely linked to self-harm [27] and suicide [30]. Therefore, it is of key importance to successfully detect cyberbullying in order to prevent them from escalating. Latest research on the expectation of such detection systems found that a major part of the respondents desired automatic monitoring over manual surveillance [31].

Detecting cyberbullying from microblogs is an intriguing research topic. This is because, it is important for law enforcement agencies which have been lagging behind in terms of technology, to collaborate with the relevant people who are currently working in this area of sentimental analysis so that they can detect the usage of social media in spreading hate messages to put an end to the discrimination. Our work covers all features from mining texts from social media, applying sentiment analysis based on people's opinions expressed on social media to finally assigning polarity to them as positive, negative or neutral.

Social media has become a major source of spreading hate messages against a particular group as it lets the group know how much some people hate them. However, blocking public to use social networking mediums will be harmful since our lives are highly involved with participation in social media for the means of socializing, education and business purpose. On the other hand, instead of blocking the sites, if measures are taken to detect racism and bullying through microblogs, then actions could be held against a certain sector of people rather than the mass public.

Racial and religious attacks are the most prominent form of cyber bullying. For instance, during holy artisan attack in Bangladesh, the Paris Attack etc. Muslims were condemned worldwide by hate speeches

through social media. The hashtag #StopIslam was trending across the world as social media users debated the involvement of religion in the Brussels terror attacks. Our research aims in detecting such hateful messages and suggesting that there is a high possibility of crimes rising from the places the hashtag originate from, and in turn informing the police as well as warning the community about how much undesirable results it generates.

Moreover, an agitated Christian named William Celi from California stood outside a mosque in Richmond and shouted "I'm going to kill you all" towards a group of Muslims leaving Friday prayers. This behavior of his, was correlated with his Facebook page that was littered with hate-filled comments about Muslims. However, the only penalty he faced was jail for 90 days [2]. Had the tables been turned, and a Muslim man threatened the Christian community, it would have made news worldwide and much more severe actions taken against him. With the aid of our research, we aim to bring justice within religious communities by detecting racism and hate messages directed towards religious committees.
In addition, on January 24th 2014, when The Grammy's telecasted live wedding ceremonies of 33 gay couples to show support towards LGBT rights, twitter got flooded with hateful messages from homophobic worldwide. People are entitled to have their own opinions, but bullying and putting a community down just because of contradiction of beliefs is intolerable and unjust. Also, it puts threats towards LGBT communist as there are reports of numerous attacks against them. For instance, over 1700 transgender people have been killed since 2008 in Central and South America, reported by Transgender Europe's Trans Murder Monitoring project. Also, research suggests one out of every two LGBT youths are regular victim of cyberbullying. It is also suggested that cyberbullying has much greater impact on LGBT community, producing suicidal thoughts within them who have been repeatedly victimized. The goal of our research is to portray the intensity of such cyber bullying, so as to display how much threat prone LGBT community is.

2.2 Related Works

Many researchers have worked mostly with finding out crime pattern from social media or Internet blogs using sentiment analysis. Very few research has been conducted on the context of cyberbullying. Their approach differs from ours in many ways and so, we will discuss a brief summary of their research and results.

Kenedy Dende, in his paper wrote on dealing with data mining to detect crime [7]. Their main objective was to design a model which is capable of categorizing and classifying data from the cyberspace as positive, negative and neutral. The specific objectives of this work are summarized as follows:

I. Design a sentiment analysis model based on Naïve Bayes.

II. Develop a sentiment classifier that is able to classify sentiments into positive, negative and neutral.

III. Embed the classifier developed in a web based application for the purpose of analyzing the efficacy of the designed Naïve Bayes Model.

In this project they tried to enable users to get the advantage when the application is implemented and therefore law enforcement agencies can use it to notice sentiments expressed in the social media. This will then diminish crime that emanate from the social sites before they occur or mature. Their discovery and increased emphasis on cooperation and sharing intelligence means that law enforcement agencies are expected to gain access to sensitive information that are used to commit crime.

H. Bouma et al. present a technique to find abnormal behavior on the internet and give an early warning for threats which is very similar to what we're trying to accomplish. The system was tested on Twitter data [6]. The results show that it can successfully analyze the content of tweets and recognize irregular changes in sentiment and threatening content.

They used Turkish-Kurdish data as their corpus. They used anomaly detection while processing the data. New data that does not lie within the same distribution of the normal 7 / 12 training data, is detected as anomaly. The classification scheme uses some features to classify anomalies and some other as post-processing to filter the output of the classification. They used anomaly detection method and applied it to the Turkish-Kurdish data and the 4daagse data and detected crime.

Raja Ashok Bolla conducted his research on crime pattern detection and collected over 100,000 crime-related tweets over a period of 20 days [8]. Sentiment analysis techniques were conducted on these tweets to analyze the crime intensity of a particular location. For this purpose they collected data from twitter and then did geographic analysis on it. After that they did sentiment analysis where the sentiment classification is divided into two categories: binary sentiment classification and multi-class sentiment classification. Naive Bayes, Maximum Entropy, and Support Vector Machine are the algorithms that they used to detect crime pattern in a particular location.

V.H. Cynthia et al. detected different categories of cyberbullying e.g. blackmail, curse, defamation, insult, sexual talk, defense, harasser encouragement etc. from different sources which include the social

networking site - Ask.fm, campaign donations consisting of victim's messages of cyberbullying and simulations [1]. The data were then classified using only support vector machines since they work well with high-skew text classification tasks. As preprocessing steps, they applied tokenization, PoS tagging and lemmatization to the data. They carried out two classification tasks; cyberbullying event detection and the classification of text categories related to cyberbullying. For evaluation, they used a metric called F-score, which is the weighted average of the classifier's precision and recall.

Chapter 3: Terminology

Before getting started on the methodologies for cyberbullying detection, we will introduce some words to get an idea of what aspects of sentiment analysis we are using for our research and why we chose to use them.

3.1 Sentiment Analysis

Sentiment analysis is the course of computationally detecting and categorizing sentiments expressed in a piece of text or in a whole content, especially in order to decide whether the writer's attitude towards a particular topic is positive, negative, or neutral. It is a combination of natural language processing, text analysis and computational linguistics.

In this process a sentence is considered positive if it has positive keywords and is considered negative if it has negative keyword. The comparison among the number of each type of contents decides the positivity and negativity of the whole content. This study tends to provide an algorithm that may help in analysis of words that may lead to crime detection especially in social sites.

For our research, we are using machine learning sentiment analysis technique. The three algorithms that we're using are Naive Bayes, Support Vector Machine and Neural Network. For Naive Bayes and Support Vector Machine, an initial training data set is required which need to be labelled with positive, negative and neutral sentiment accordingly. For that, we are using lexicon based approach instead of manually classifying 11,500 tweets on our own.

3.2 Corpus

To detect bullying, we need to gather a large amount of data as corpus so that data mining can be done and we can predict the outcome. Hence, we used twitter API to collect a corpus of text posts and formed a dataset of three classes: 1. Positive sentiments, 2. Negative sentiments and 3. Neutral sentiments. Now for this process we needed an authorization key which is provided by twitter when you login.

3.3 Hashtag

To find relevant tweets from the twitter we used Hashtag. A hashtag is a way of categorizing tweets so that they are a part of a narrowed conversation and they are easier to find in twitter search. Users create and use hashtags by inserting the hash character (#) before the topic or theme which is a usually a word or unspaced phrase in the beginning, middle or at the very end of a main text of a message, comment, post or

caption. For example: #StopIslam, #sorrynotsorry, #NewYear2016 etc. Clicking on a Hashtag in any message shows all other tweets marked with that keyword.

3.4 Bag of Words

The bag of words model is a simplifying representation used in natural language processing and information retrieval (IR). In this model, a text or a document is characterized as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity.

Bag of words is created by using one of the following preprocessing techniques:

- Unigrams
- Bigrams
- Stemming
- Lemmatization
- Parts of speech tagging

In our research we used unigrams preprocessing technique. Unigram is a model used in information retrieval can be treated as the combination of several one-state finite automata, i.e. each words are treated independently.

Chapter 4: System Implementation

The complete system at this stage consists of the components listed below. Each of the main components is explained in more detail.

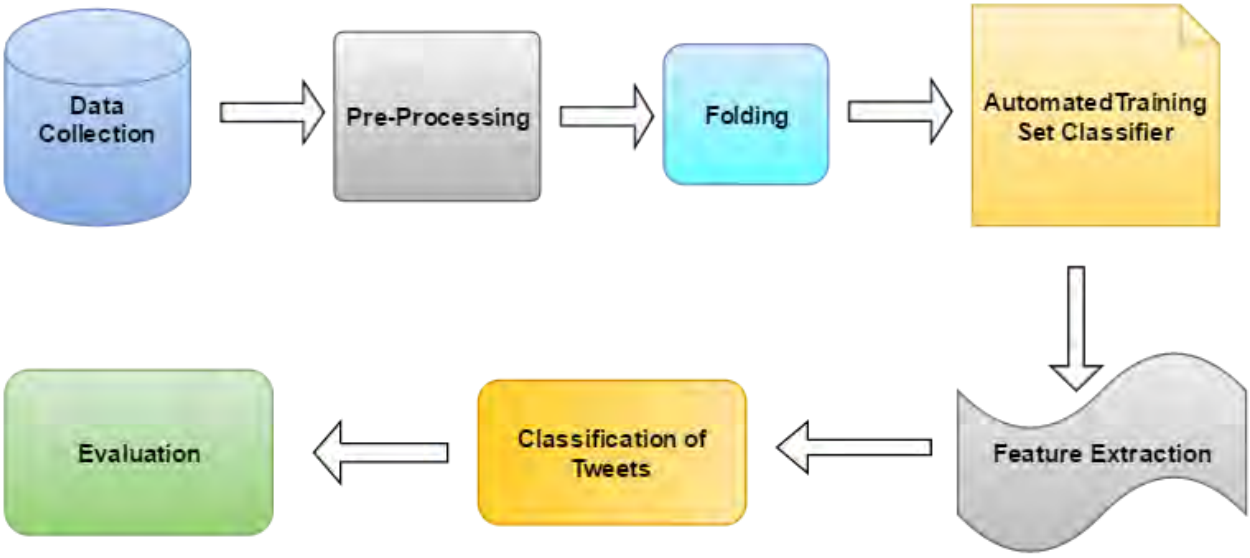

Fig- 4.1.1 Block Diagram of Proposed System

4.1 Data Collection:

As mentioned before, we used twitter to collect relevant data since it is a social networking and microblogging website. To get access and gather our required corpus it was needed to get the Twitter API for python which requires an authorization key. The key provided by twitter can now be used to collect corpus from microblogs. We crawled the twitter API for two days by deploying our code to a herokuapp and directly collecting the tweets in the database. The tweets were tracked using particular keywords that were prone to indicate bullying. The keywords used were - *dyke, gay, black nigger, bitch, StopIslam, ugly, jihad, muslims terrorists, deported, ban islam.* These keywords signify three kinds of bias - racial bias, gender bias and religious bias. Using Twitter API we collected a corpus of text posts and formed a dataset of three classes: positive sentiments, negative sentiments, and a set of objective texts. The two types of collected corpora will be used to train a classifier to recognize positive and negative sentiments. In order to collect a corpus of objective posts, we retrieved text tweets from Twitter.

4.2 Pre-Processing:

Initially for pre-processing we corrected the spelling mistakes in the tweets as many people tend to right English words in short form i.e., “u r nt pretty” is converted to “you are not pretty”. Then we converted the uppercase letters of the tweets to lowercase order. Then we removed all the usernames, URLs and unnecessary white spaces from the tweets.

Stopwords are words that are generally considered *useless*. Most search engines ignore these words because they are so common that including them would greatly increase the size of the index without improving precision or recall. Any group of words can be chosen as the stop words for a given purpose. For some search engines, these are some of the most common, short function words, such as *the*, *is*, *at*, *which*, and *on*. In this case, stop words can cause problems when searching for phrases that include them, particularly in names such as "The Who", "The The", or "Take That".

People love to express their emotions through emoticons in tweets. So we are also detecting the emoticons using regex package of python. We coded the regular expressions of normal eyes, nose area, happy mouths, sad mouths, angry mouths etc. of the emoticons i.e., for happy mouth r'[DvV\)\]]', nose area= r'(|o|O|-)'. Then we split the tweets to find out the emoticons and replaced them with a corresponding word that defines the emoticon. Emoticons that do not help in finding out negativity or positivity of a message are replaced with the word neutral, so that those can be ignored. This process will help to find out polarity of the tweets.

4.3 Folding:

In the dataset, we use 80% of the dataset for training and 20% for testing purpose. It is quite rare to use 50/50, 80/20 is quite a commonly occurring ratio. A better practice is to use: 60% for training, 20% for cross validation, 20% for testing. From the 15,000 tweets and used 11,500 for training the classifier and 3,500 for testing the polarity of the tweets against the classifier.

4.4 Automated Training Set Classifier:

Machine-learning sentiment analysis requires a set of tweets labeled positive, negative or objective, it can be created by hand, by labeling tweets manually. Although this will create a highly reliable lexicon, it requires dozens of hours of work [6]. We opted for a more automatic approach by collecting hundreds of thousands of tweets and running an algorithm through those tweets which compares each individual words with positive list and negative list of words. The pseudocode is outlined below:

1.Initialize positiveWordList, negativeWordList

2.positiveScore = negativeScore = 0

3.for each word in tweet

3.1. if word in positiveWordList:

3.1.1. positiveScore = positiveScore + 1

3.2. If word in negativeWordList:

3.2.2. negativeScore = negativeScore + 1

4.if positiveScore > negativeScore:

4.1. return 'positive'

5.if negativeScore > positiveScore:

5.1. return 'negative'

6.if positiveScore = negativeScore:

6.1. return 'neutral'

After labelling each tweets with their polarities, the training set is stored in a NoSQL non-relational database named *mongodb.* The reason for choosing a non-relational database is because it is more scalable for large datasets than compared to a relational database. As most of the sentiment analysis related research have been conducted using a MySQL database, we tried something different for storing the data. The schema of the twitter data comprised of *tweet, sentiment, geolocation, username, fullname.*

4.5 Feature Extraction:

I. **Stop words -** a, is, the, with etc. These words don't indicate any sentiment and can be removed. We kept a text file with all the possible stopwords and filtered them out for feature extraction.

II. **Repeating letters -** Looking at the tweets, sometimes people use repetitions to stress emotion. E.g. thirsty, thirssstyyy for 'thirsty'. We can look for 2 or more repetitive letters in words and replace them by 2 of the same.

III. **Punctuation -** We can remove punctuation such as comma, single/double quote, question marks at the start and end of each word.

IV. **Words with alphabets at the beginning –** As we process, each of the tweets, we keep adding words to the feature vector and ignoring other words. The entire feature vector will be a combination of each of these feature words. For each tweet, if a feature word is present, we mark it as 1, else marked as 0. Now, we can think of each tweet as a bunch of 1s and 0s and based on this pattern, a tweet is labeled as positive,

neutral or negative. Given any new tweet, we extract the feature words as above and we get one more pattern of 0s and 1s and based on the model learned, the classifiers predict the tweet sentiment. For Naive Bayes, we used bigram feature words as bigram is considered more accurate compared to unigrams.

4.6 Analysis and Classification:

The training tweets were retrieved from database, classified and run through three machine learning classifying techniques to compare and contrast the performance of those algorithms. After the classifier has been trained, the test tweets, which were pre-processed and its' features extracted, were run through the classifier to detect the polarity. After the polarity has been detected, it was used to compare the accuracy of the classifiers.

4.7 Evaluation:

After implementing the three machine learning approaches to determine which of the classifiers used for detecting cyberbullying gave the most accurate results and is preferable over the other, we determined the accuracy of the classifier, precision, recall and f-score of the positive, negative and neutral tweets. Precision and recall are the metrics used to determine classifier output quality. Precision is the measure of how relevant the results are and recall is the measure of how many relevant results are returned. F-score is the average of both the precision and recall [5]. A system with high precision but low recall means that most of its predicted labels are correct when compared to the training labels whereas a system with low precision but high recall means most of its predicted labels are incorrect when compared to the training labels.

Chapter 5: Comparison of Machine Learning Techniques

For detecting cyberbullying from twitter, we implemented various machine learning techniques to find out the most efficient one with the highest accuracy. All of these techniques required a training set which were collected, preprocessed and had been run through the classifier techniques. We implemented two supervised learning technique, namely Naive Bayes and Support Vector Machines and compared the results with a reinforcement based learning technique, Convolutional Neural Network to check how much the results differ from one another.

5.1 Naive Bayes Classifier

5.1.1 Background

Naive Bayes classifier is based on the Bayes' Theorem and is a supervised learning approach. Specifically, a supervised learning algorithm takes a known set of input data and known responses to the data, and trains a model to generate reasonable predictions for the response to new data. Naive Bayes classifier is used for sentiment analysis purposes due to its high accuracy. Although it is a simple theorem, it performs almost as well as many other complicated approaches. It is essentially a set of supervised learning algorithms based on the application of Bayes' theorem with the "naive" assumption of independence between every pair of features [3]. Given a class variable and a dependent feature vector through , Bayes' theorem states the following relationship:

$$P(y|x_1,...,x_n) = \frac{P(y)P(x_1,...,x_n|1)}{P(x_1,...,x_n)}$$

For all *i,* this relationship is further simplified to:

$$P(y \mid x_1,\ldots,x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i \mid y)}{P(x_1,\ldots,x_n)}$$

This means that - the probability that classification y is correct, given the features x_1, x_2, and so on equals the probability of y times the product of each x feature given y, divided by the probability of the x features.

5.1.2 Experimental Setup

We implemented the python nltk package for Naive Bayes classification. The training sets need to be labelled in order to recognize the category a corpus is classified upon. For example, if we are trying to find the gender mentioned in a particular corpus, the labels would be male and female. For our case, we are trying to detect bullying and hence we need to find out if a particular tweet is positive or negative or is opinionated/neutral. The negative tweets are regarded as cyberbullying related tweets. The labelled tweets were then stored in MongoDB. If it is positive or neutral, then there is no harm done and we leave it at that. However, if a tweet is negative, we can successfully identify cyberbullying. Now the question remained, how accurate was the detection of this negative tweet. We collected a large data set as mentioned before for training the classifier in order to increase the accuracy.

By iterating through the training set, Naive Bayes classifier finds out the number of occurrences of each bigram word and checks if the test sentence has the same feature words as the training data. After the preprocessing of training set was complete, the bigram feature vectors were extracted from every tweets. We maintained two frequency distributions, one for counting the number of occurrences of individual words and another for counting the number of occurrences of bigram frequencies. By maintaining these frequency distributions, it can score bigrams individually using a scoring function called *chi-square* [4]. The scoring function was used to measure the collocation correlation of two words to check whether the bigram occurs as frequently as each individual word. A collocation is an expression consisting of two or more words that correspond to some conventional way of saying things. In order to find the highest informative words, we used a chi-square of 200 most informative words since it produced the best results. This was done to find the information gain for classification which is a measure of how common a feature is in a particular class compared to how it is in all other classes.

After the bigram features were extracted and added to the feature vector, we trained the Naive Bayes classifier using the built in package function with the 11,500 tweets that were collected and annotated. Then we moved on to testing the polarity of the test data. In order to determine how much precise and accurate our classifier was we also found out some metrics like precision, accuracy, recall and f-score. These will be further discussed in the Results chapter.

5.2. Support Vector Machines (SVM)

5.2.1 Background

The Support Vector Machines is a classifier that finds best hyperplane between two classes of data, by separating positive and negative examples through solid line in the middle called decision line. The goal of svm is to maximize the margin between these separate classes as shown in Fig 5.2.1 [9]. To make svm work, we must feed data to this algorithm in order to train the system. The graph in figure 1, as provided in the scikit-learn documentation (collected from scikit-learn documentation), shows the optimal separating hyperplane between two different classes of data. We must note that svm is not only effective for just 2 classes of data, but for multiple classes, however, in which case it gets difficult represent for our visual needs. Our implementation expects to view 3 separate clusters of data classified under the categories 'positive', 'negative' and 'neutral' [11]. The mathematical formulation, as per Scikit-Learn documentation [10], where given is training vectors $x_i \in \mathbb{R}^p$, i=1,..., n, in two classes, and a vector $y \in \{1, -1\}^n$, the problem solution is

$$\min_{w,b,\zeta} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \zeta_i$$
$$\text{subject to } y_i(w^T \phi(x_i) + b) \geq 1 - \zeta_i,$$
$$\zeta_i \geq 0, i = 1, ..., n$$

whose dual is

$$\min_{\alpha} \frac{1}{2} \alpha^T Q \alpha - e^T \alpha$$
$$\text{subject to } y^T \alpha = 0$$
$$0 \leq \alpha_i \leq C, i = 1, ..., n$$

where e is the vector of all ones, $C > 0$ is the upper bound, Q is an n by n positive semidefinite matrix, $Q_{ij} \equiv y_i y_j K(x_i, x_j)$ Where $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$ is the kernel. Here training vectors are implicitly mapped into a higher (maybe infinite) dimensional space by the function ϕ.The decision function is:

$$\operatorname{sgn}(\sum_{i=1}^{n} y_i \alpha_i K(x_i, x) + \rho)$$

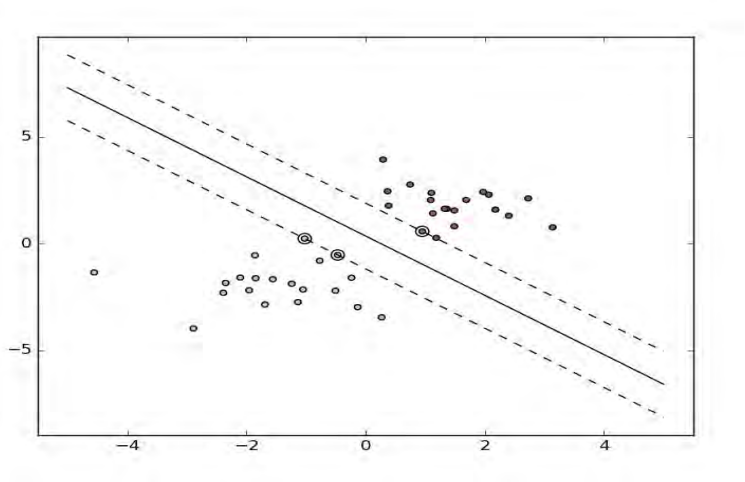

Fig 5.2.1 Graph highlighting the optimal separating hyperplane for two classes of data [10]

For our purposes of classification, the Support Vector Classifiers (SVC) that we selected are as follows:

- (Default) SVC with RBF (Radial Basis Function, or Gaussian) kernel [14].
- SVC with linear kernel [10]
- LinearSVC [12]

Although the latter two of the aforementioned classifiers work best when the data is linear, the default SVC kernel works slightly better with non-linear data [14]

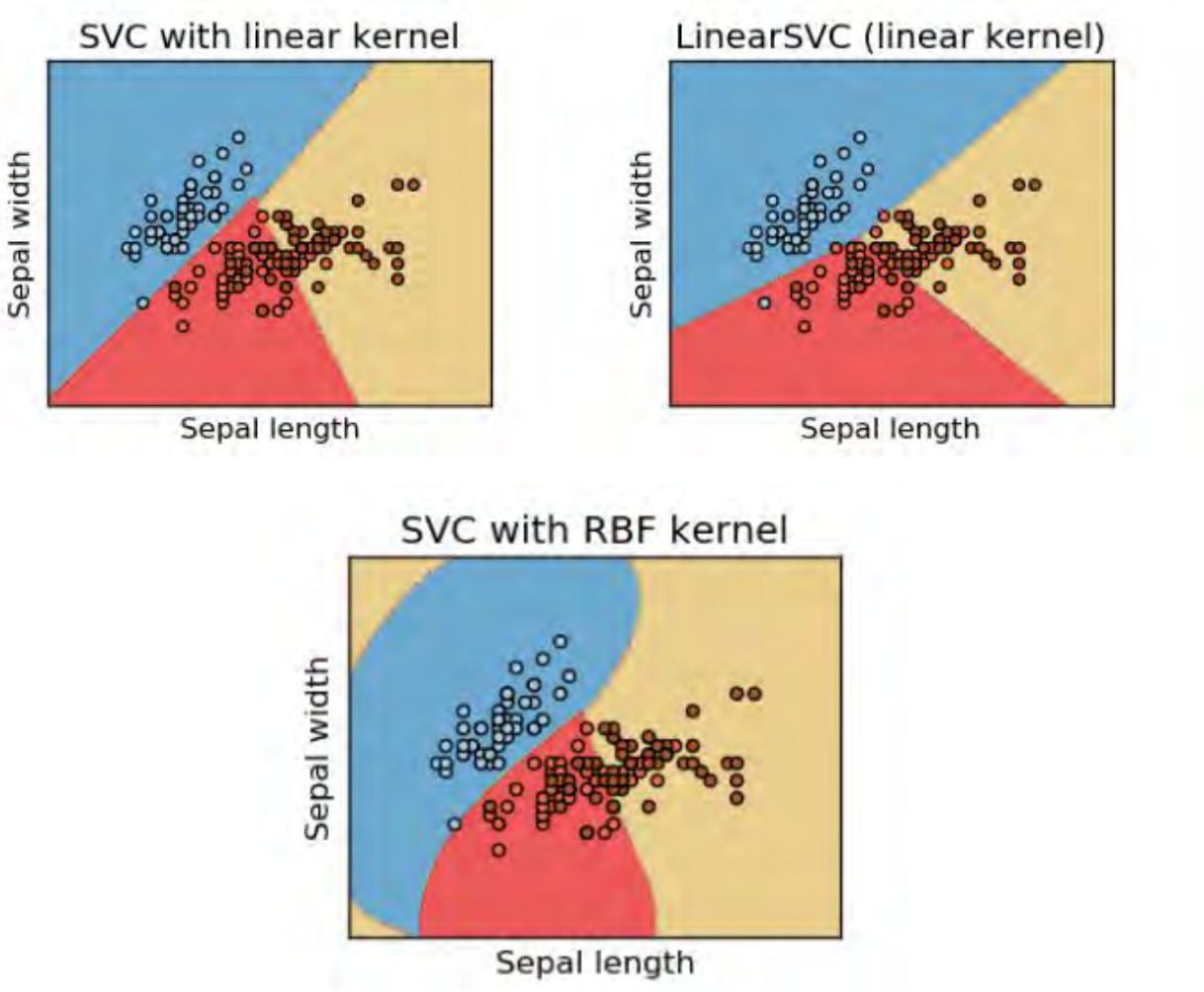

Fig - 5.2.2 Demo classification of the selected classifiers [10]

5.2.1 Experimental Setup

Since our system is implemented in python, we have opted to use the popular machine learning library scikit-learn, which provides a handful of algorithm and resources for data scientists. 11,500 tweets were collected using various keywords and then classified as per the above categories mentioned with the help of an automated classifier which is also designed by us. These 11,500 tweets serve as our training data that we opted for this experiment. Before proceeding with classification, the tweets were preprocessed to remove any unwanted element from the tweets, for instance, usernames, any URL, etc. It is also essential for these tweets to be in lowercase for the experiment.

In classification, the items are represented by their features, and these features need to be extracted from sample data sets. The data sets here consist of a database of tweets and hence, nothing serves better as the

features other than the words in this document. Using TF-IDF (Term Frequency- Inverse Document Frequency) vectorizer, we vectorized the input tweets into a format that the machine can identify. TF-IDF vectorizer also provides an efficient weighting scheme that makes it ideal for our situation [11]. The feature weights for the training data is obtained, and this vocabulary is used in determining the feature weights of the test data as well, which is nothing but training our system for the test set of data. Scikit-Learn also provides for efficient calculation of accuracy and precision measurements [15]. The training set is significantly larger than the test set and this helps to boost our accuracy and precision.

In addition, the classifiers were adjusted using several parameters such as the gamma value, C value, etc. Gamma value reiterates the importance and influence of a single entry from the training set whereas C is the penalty value of error term [14]. Some feature vectors were intentionally ignored such that their nature/frequency of of appearance did not cause unwanted influence in our system, as done by M. Bonazini in his research on a very similar topic. Too common words were ignored as it should not be classified as a feature of course and words too rarely appearing in our data sets are also ignored for they may appear only under special circumstances and might not affect polarity of the tweets by any means [11].

5.3 Convolutional Neural Network

5.3.1 Background

Neural Network:

Neural networks are models exhibited after biological neural structures[25]. A fair understanding of how an individual neuron works gives a vivid picture of the way this model functions.

Structure of Real Neurons:

A typical neuron consists of an *axon*, and a *cell body* surrounded with *dendrites*. Signals generated by neurons are passed along the axon on to the dendrites of other neurons which act like receptors.. In surplus of a certain *threshold* or *activation* level, the combination of these signals will result in the neuron *firing*, that is sending an impulse on to other neurons connected to it. All human actions are believed to be the united effect of the firings in the sequence of synaptic connections between neurons.

Neural Network Structure

A model neuron is the starting point for most neural networks, as in Figure 5.3.1.

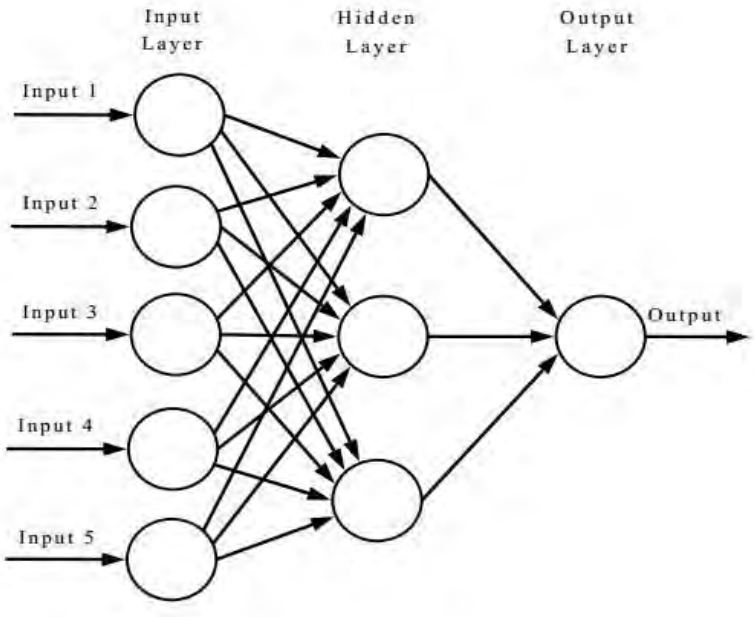

Fig - 5.3.1 General Structure of a Neural Network

Each neuron consists of multiple inputs and a single output to several other neurons in the next layer. All the layers are then arranged one following the other so that there is an input layer, various intermediate layers and finally an output layer, as in Figure 5.3.1.

Neural Network Operation

Equation 5 shows the output of each neuron as a function of its inputs. wj and xj represent as vector matrices of weight and input of jth neuron respectively[24]

$$Output = \begin{cases} 0 & \text{if } \sum_j w_j x_j \leq threshold \\ 1 & \text{if } \sum_j w_j x_j \geq threshold \end{cases} \text{ - - - - (5)}$$

For every neuron, j, in a layer, a previously established weight, wij is multiplied with each of the i inputs, xi The output achieved by summing all these values and then biasing it by a previously established threshold value, b, as shown in equation 6.

$$Output = \begin{cases} 0 & \text{if } w.x + b \leq 0 \\ 1 & \text{if } w.x + b \geq 0 \end{cases} \text{ - - - - (6)}$$

This is then sent through an activation function, Fth. This activation function is usually the sigmoid function, defined by equation 7.

$$\sigma(z) = \frac{1}{1 + \exp(-\sum_j w_j x_j - b)} \text{ - - - - (7)}$$

The resulting output, is an input to the next layer or it is a response of the neural network if it is the last layer. Equation 1 computes the combination operation of the neuron and Equation 3 decides the firing of the neuron. The description of the network structure which comprises of the number of layers and the number of neurons in each layer are used along with a predetermined set of weights, a predetermined set of threshold values, to implement response of the neural network to any set of inputs.

Convolutional Neural Networks:

A convolutional neural network (CNN) is a type of artificial neural network which consists of a number of convolutional and subsampling layers, succeeded by fully connected layers[21]. The figure below illustrates a full layer in a CNN**.**

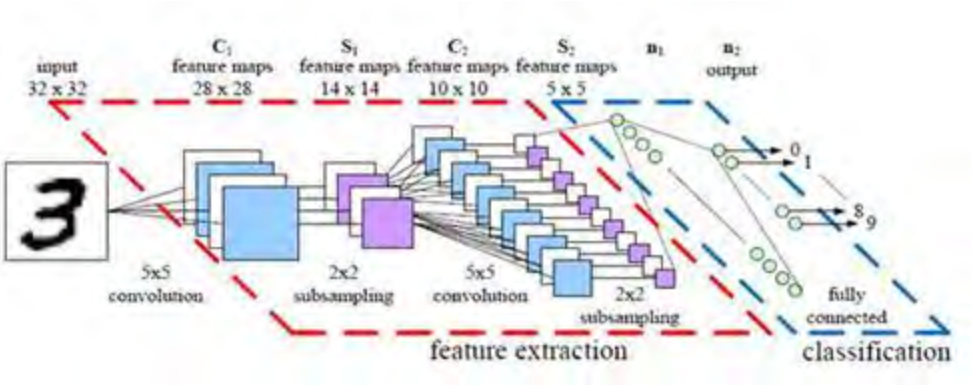

Fig - 5.3.2 Full layer in a CNN[23]

Convolutional layer:

The first and one of the most important layers is the convolution layer. The convolutional layer will have k filters, which look for parts of the input sentence that are similar to the filter. The one-dimensional convolution is an operation between a vector of weights m ∈ R m and a vector of inputs viewed as a sequence s ∈ R s where m is the filter of the convolution. Convolution is the amount of overlap of one function when shifted over to another. The filter here moves across the input sentence and finds the similarity between the filter and words in the sentence. Filters are also learnt during training, so as we keep on training more data, we get better filters that suit our network needs. There are some parameters (hyper) that are fixed by us and do not depend on training i.e number of filters, the stride (the offset of how much to move the filter over the sentence).

Maxpooling layer:

Pooling layers typically comes after the convolutional layers. The input received by pooling layers is subsampled. The most common way to do pooling it to perform a max operation to the result of each filter. For example, the following shows max pooling for a 2×2 window (pooling is usually applied over the complete output in NLP, yielding just a single value for each filter)[20]. Pooling reduces the amount of storage and overfitting.

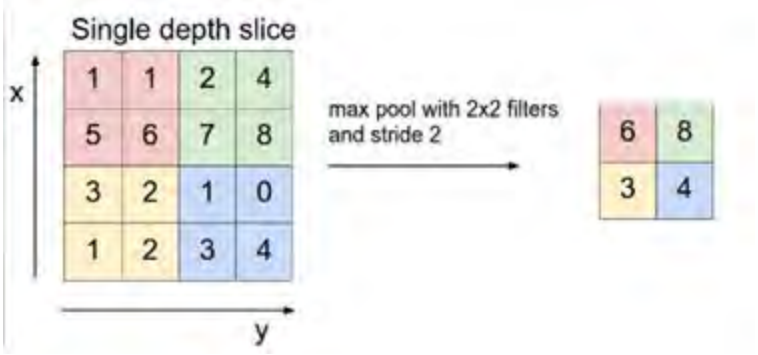

Fig - 5.3.3 CNN Max Pooling layer

Convolutional Neural Network Learning by Backpropagation

What has intrigued the most interest in neural networks is the possibility of learning. Learning in a neural network is called training, and it is done by backpropagation. Backpropagation is a procedure that is applied repetitively for each set of inputs and corresponding set of outputs produced in response to the inputs.. This is the most predominant and generalized neural network currently in use. From the variance between the actual response and the desired response, the error is computed and a portion of it is propagated backward through the network.The error is used to adjust the weights and threshold values of each the neuron, to make sure that during the next iteration, the error in the network response will be less for the same inputs. This method proceeds as long as the individual or total errors in the output exceed a threshold value or until there are no considerable errors. This point onwards, the neural network has learned the training dataset and training process can be halted and the neural network can be used to produce responses to new input data.

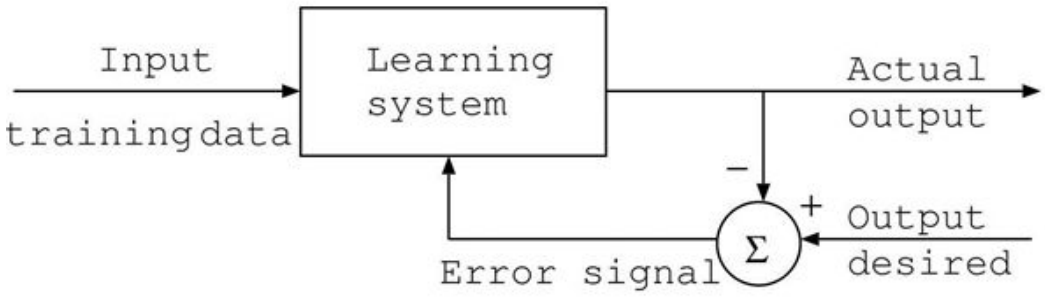

Fig - 5.3.4 Backpropagation process

5.3.2 Experimental setup:

The network consists of 2 convolution layers with filters of widths 8 and 6.[17] The parameters also consist of word embeddings and weights of the fully connected network at the top. One dimensional convolutions are calculated at each layer by matrix multiplying the weights with the input matrix.

Training:

The network is at first trained using an existing corpus. In our case it was the twitter dataset we collected. Theano framework is used by the algorithm. Theano is a numerical computation library specifically made for Python. In Theano, computations are expressed using a NumPy-like syntax and compiled to run efficiently on either CPU or GPU architectures. The readymade packages nltk and numpy were also used to handle the complex vector multiplications and calculating the non-linear results between layers.

Since the neural network is optimized to work with numbers, each sentence that is input is converted into a vector form. This is done using word2vec, which is a neural network in itself. Each row of the matrix corresponds to one token, typically a word. These words are then mapped to vectors of real numbers. These words are known as word embeddings.

Each word in the input matrix can be represented by more than one feature.

In addition to the feature of the word itself, it can also consist of substring features. For example for the words “eating”, there can be two features which are the prefix “eat” and the suffix “ing”. The matrix may also consist of gazeteer features i.e. determining whether the word in the sentence belongs to a list of know persons, objects etc. All these features are represented in the following form:

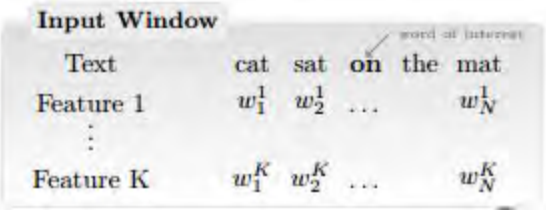

Fig - 5.3.5 Input matrix representation of a convoluted neural network [18]

Convolution:

A convolution matrix is obtained by convolving a matrix of weights **m** with the matrix of activations at the layer below. In the case of the second layer, the output is obtained by matrix multiplying the weights with the sentence matrix. At first a filter(or kernel) is passed over the input matrix. The filter extracts n-grams from the matrix in sequential order. These subsequences of word representation help to

keep the sequences of words intact, maintaining order and positions of the words with respect to each other. The n-grams have to be equal or less than the width of the filters.

The dot product between vector m with each m-gram in the sentence results in the following sequence:

$$\mathbf{c}_j = \mathbf{m}^\intercal \mathbf{s}_{j-m+1:j}$$

There are two types of convolutions: narrow and wide.

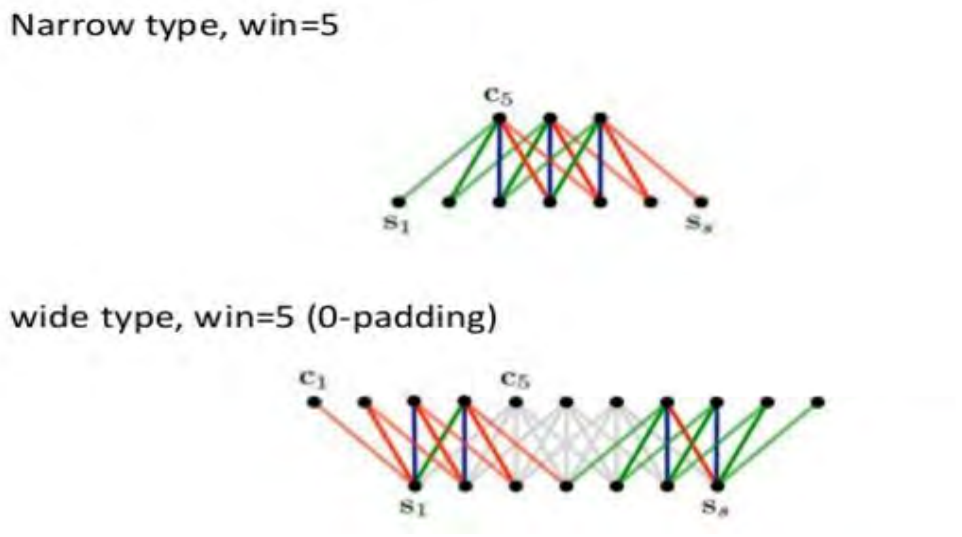

Fig - 5.3.6 Narrow vs wide convolution

In the case of narrow convolution, the dimensionality of the input matrix needs to be less than the dimensions of the vector of weights. A wide convolution, however, has no such requirement. Out of range values are taken to be zero. By using a wide convolution network, all the words in the sentence are guaranteed to be taken into account. Because of its advantages, the neural network used implements wide convolution.

K-max pooling:

The neural network implements a k-max pooling function

$$k_l = \max(\, k_{top}, \lceil \frac{L-l}{L} s \rceil \,)$$

The ktop symbol implies that for the input layer, the max pooling should always have a fixed value fot the topmost layer. L signifies the total number of convolutional layers while l denotes the layer under consideration at the moment. An example would make this formula clear. If we use an input sentence of

length 12 and convolutional neural network with 3 layers and ktop=3, the max pooling at the first layer would be 8 and 4 at the second layer. The third layer will have the constant pooling parameter ktop=3.

After each convolutional layer, there may be a pooling layer. The pooling layer takes small rectangular blocks from the convolutional layer and subsamples it to produce a single output from that block. There are several ways to do this pooling, such as taking the average or the maximum, or a learned linear combination of the neurons in the block.

The k-max pooling operation makes it possible to pool the k most active features in p that may be a number of positions apart; it preserves the order of the features, but is insensitive to their specific positions. It can also discern more finely the number of times the feature is highly activated in p and the progression by which the high activations of the feature change across p. Because of max-pooling, no matter what the length of the sentence is, the fully connected layer will always receive an input matrix of fixed dimensions.

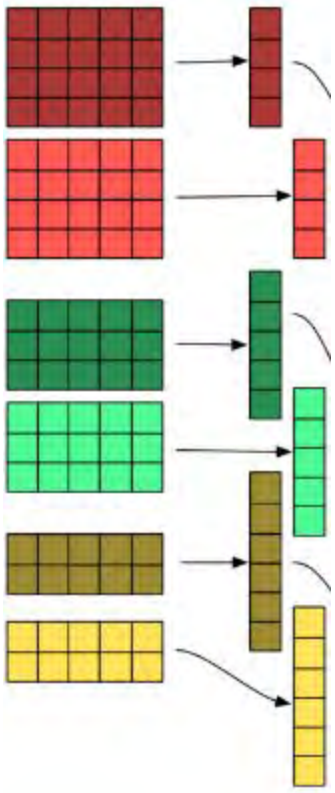

Fig - 5.3.7 Max pooling [20]

Folding:

After a convolutional layer and before (dynamic) k-max pooling, one just sums every two rows in a feature map component-wise. For a map of d rows, folding returns a map of d/2 rows, thus halving the size of the representation. This helps to avoid complex dependencies across the same rows in multiple feature maps.

Non-linear function:

After (dynamic) k-max pooling is applied to the result of a convolution,n, a bias and a nonlinear function are applied component-wise to the pooled matrix. M is defined to be the matrix of diagonals: M = [diag(m:,1), . . ,diag(m:,m)] where m are the weights of the d filters of the wide convolution. Then after the first pair of a convolutional and a non-linear layer, each column a in the matrix a is obtained as follows

$$a = g\left(\mathbf{M}\begin{bmatrix}\mathbf{w}_j \\ \vdots \\ \mathbf{w}_{j+m-1}\end{bmatrix} + \mathbf{b}\right)$$

Here a is a column of first order features. Second order features are similarly obtained by applying the above equation to a sequence of first order features aj , ..., aj+m0−1 with another weight matrix M0 . In this way, the other features are calculated. Together with pooling, the feature function induces position invariance and makes the range of higher-order features variable.

Chapter 6: Experimental Results & Analysis

The results obtained from the three machine learning approaches are based on their precision, recall, accuracy and f-score. These are described below:

6.1 Naive Bayes Results

For the Naive Bayes classifier, the results obtained are illustrated below:

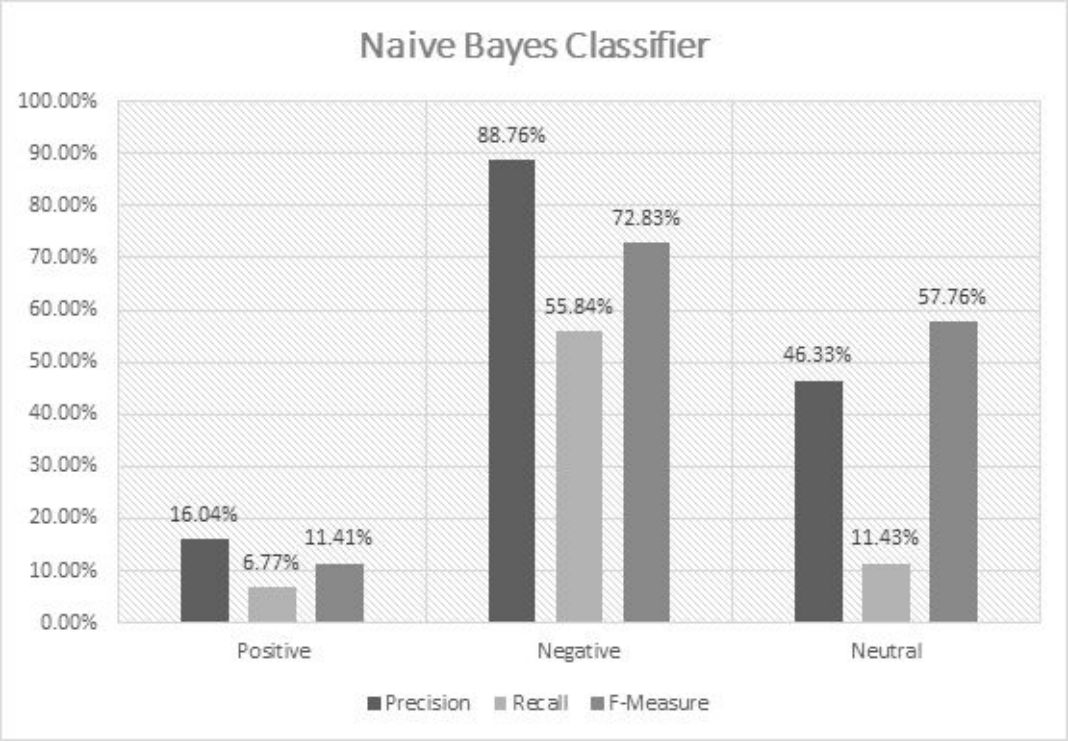

Fig - 6.1.1 Results of Naive Bayes Classifier

As shown above, Naive Bayes Classifier provided a positive precision of 16.04% and a recall of 6.77%. These are very low values for precision and recall. It means that only 16.04% of the positive tweets retrieved by the classifier were relevant and only 6.77% of the relevant positive tweets were retrieved. One reason for such low positive precision and recall may be because the context of training tweets were mostly cyberbullying related, which means they had a lot of slang and hate words compared to nice and positive words.

We will focus more on the negative results since our context is cyberbullying. The negative precision came out to be 88.76% and recall was 55.84%. This means that most of its predicted sentiment was accurate when compared to its training set by a small amount. Hence 88.76% of the negative tweets were

relevant and 55.84% of the relevant negative tweets were retrieved. This means very few *false positives* were found for the negative class. However, many tweets that are negative are incorrectly classified. Low recall causes 44.16% false negatives for the negative label.

Comparatively, the neutral precision reading came out to be 46.33% and recall is 11.43% which shows that the neutral precision is much higher than the recall. All in all, since our context is cyberbullying and we identify it from the negative tweets, our average accuracy of the cyberbullying detection results came out to be 72.83% for negative tweets.

6.2 SVM Results

As mentioned previously, three SVM classifiers were used and they all performed quite successfully. Their success is portrayed in the following charts.

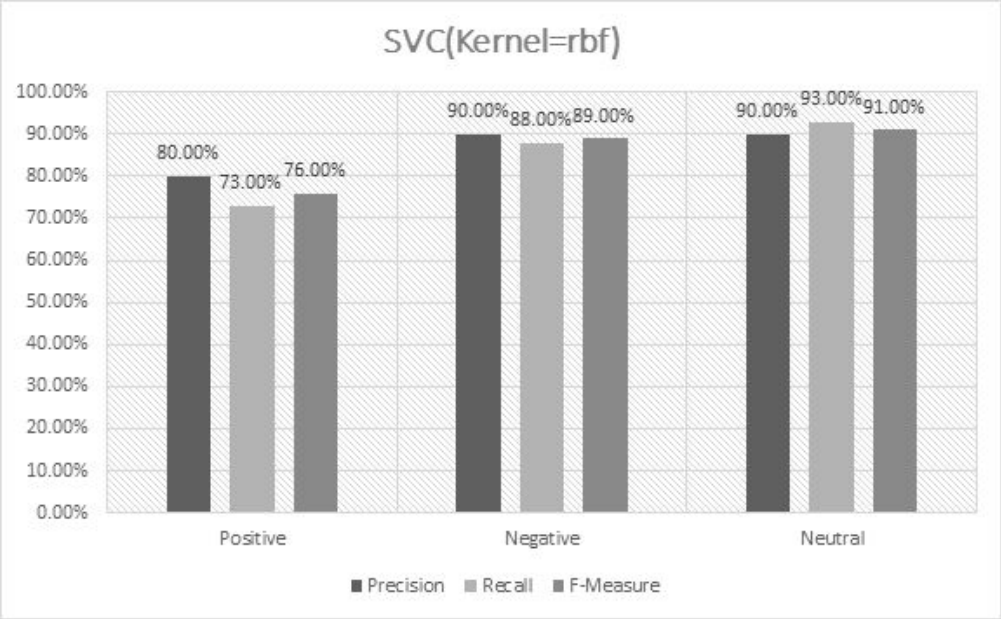

Fig - 6.2.1 Results of SVC(Kernel=rbf)

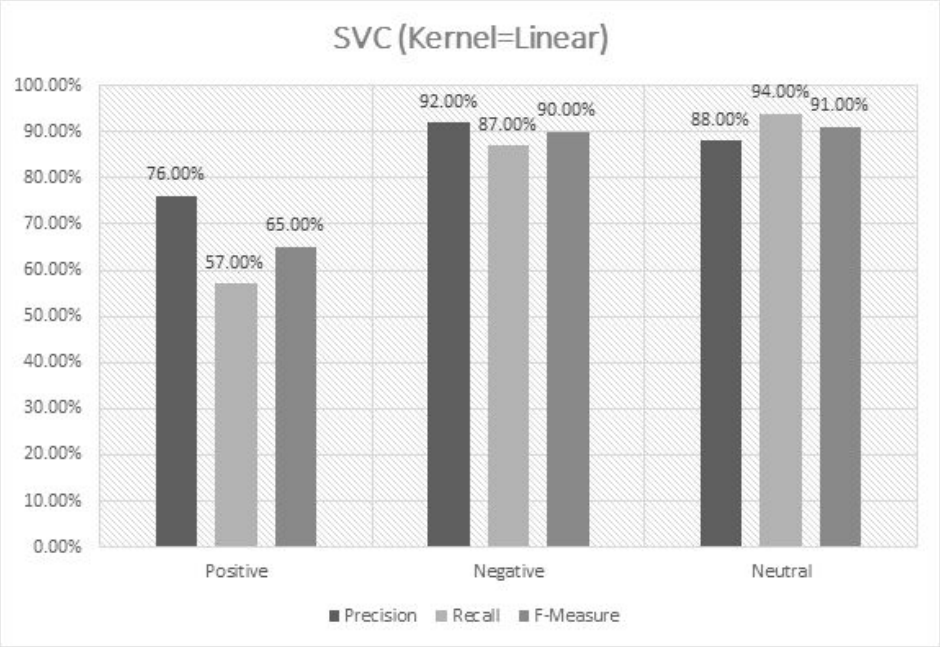

Fig - 6.2.2 Results of SVC(Kernel=linear)

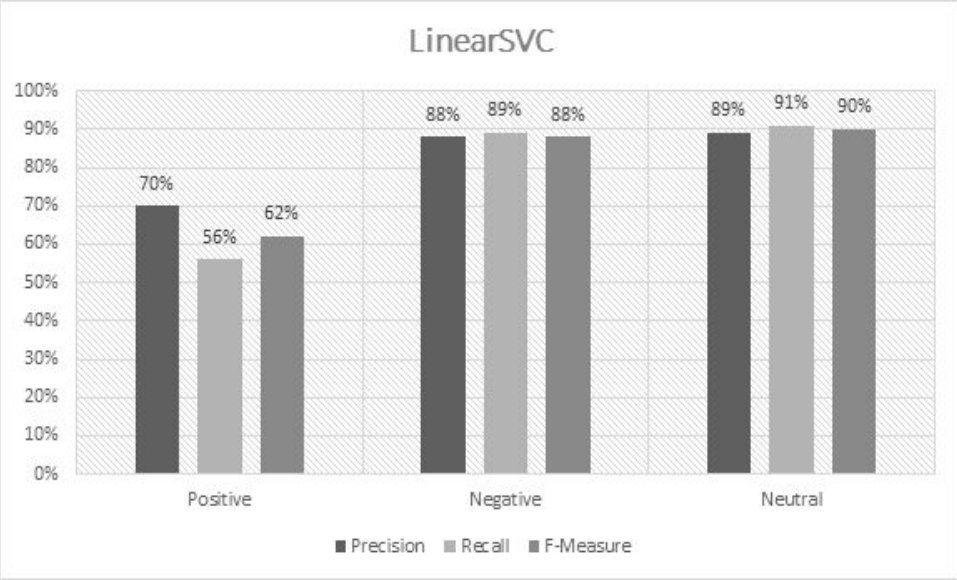

Fig - 6.2.3 Results of LinearSVC

6.2.1 RBF-kernel

Gaussian Kernel performs the best out of the other two SVM classifiers by a margin of accuracy of just 0.1% and 0.2%. This can be explained by the non-linearity of the data and the fact that they were

classified into more than two samples. However, the classification process was slow and took up to 1.5 times the time required to train and predict than that of the linear model of SVC. The performance was adjusted using a gamma value of 0.5 and and the default C value of 1.0 so as to provide the optimal output.

RBF kernel registered an accuracy of 89.39%. The precision results as shown above in the chart indicate that the kernel was successful in predicting 80% of the positive, 90% of the negative and the neutral tweets were relevant among the ones retrieved. The recall values of 73%, 93% and 88% indicate that the retrieved positive, neutral and negative (respective) tweets were relevant. F-score is the measurement of the test's accuracy and the f-score 88% for negative tweets.

6.2.2 Linear kernel

The linear kernel took considerably smaller time than RBF kernel [19], however was still about 100 times slower than LinearSVC. The linear kernel is a wrapper of the python library libsvm. Precision values read 76%, 88% and 92% for positive, neutral and negative tweets respectively whereas the values for recall stands at 57%, 94% and 87% respectively. The low recall value for positive tweets indicates the retrieved positive tweets thought to be relevant by the classifier was significantly smaller than the overall set of relevant positive tweets. Positive tweets also have the lowest precision among the other two classifiers primarily because the training data was more dense towards the negative polarity. The accuracy of this classifier is found to be 88%.

6.2.3 LinearSVC

LinearSVC is an implementation based on a different python library, Liblinear [12]. The basic advantage of LinearSVC is its speed of training and prediction compared to those of the Linear kernel and Gaussian kernel, both of which are implemented based on Libsvm [19]. LinearSVC performed at least 100 times faster than them, compromising only 1-2% of accuracy in its predictions. The precision values for LinearSVC reads 70%, 89% and 88% for positive, neutral and negative tweets. Once again, the skewness towards the negative polarity of the data set accounts for the low precision and recall reading of positive tweets. The results of LinearSVC and linear kernel are very consistent and similar. Undoubtedly, LinearSVC is the ideal classifier to implement in any given system owing to its high speed and high accuracy in predictions.

6.3 Convolutional Neural Network Results:

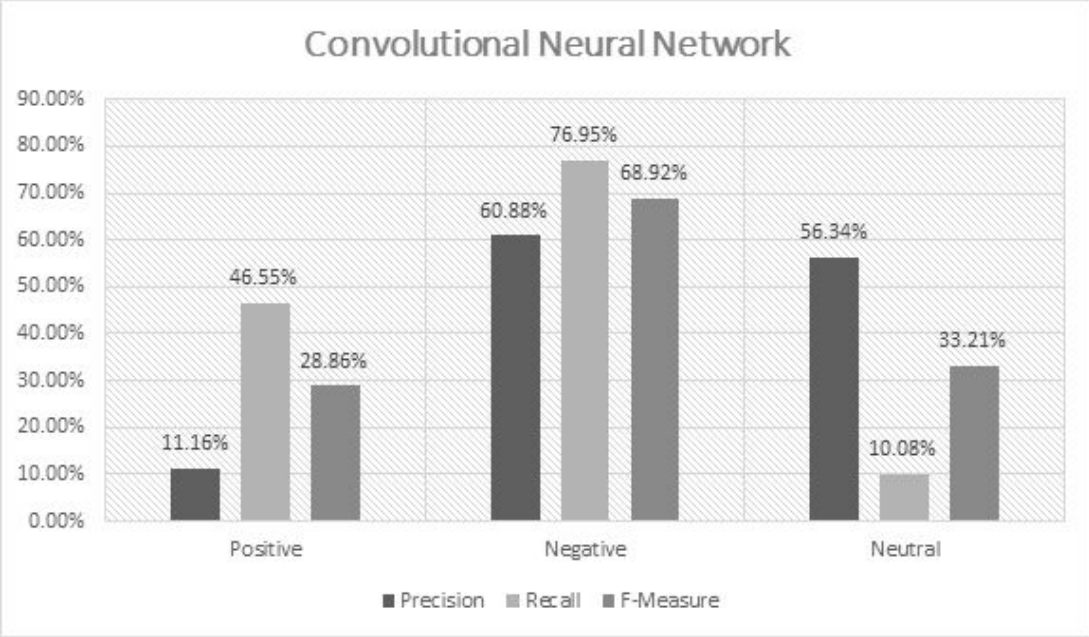

Fig - 6.3.1 Results of Convolutional Neural Network

After training, a dataset consisting of a large number of tweets were used to measure the metrics of the algorithm. After testing over a dataset of 1000 tweets an accuracy of 48.64% was obtained. The reason for the suboptimal performance is that the data we trained the classifier upon does not accurately represent the test data. As a result about 50% sentiments of the tweets are not predicted correctly. The algorithm has achieved a positive precision of 11.16%, negative precision of 60.88% and 56.34% neutral precision. The algorithm is more optimized to correctly predict the negative sentiments, while keeping the number of false positives low. While predicting for positive sentiment, it incorrectly assumes a lot of tweets to be positive when in fact they are negative, resulting in a very low positive precision. By observing the graph, it can be seen that low recall causes 54% false positives and 90% false neutrals. Recall score for negative sentiments is relatively better, with just 23% false negatives.

The main reason for the poor results is mainly because the training dataset does not properly represent the test data. We have chosen 0-0.4 as the range for negative sentiment, 0.4-0.6 for neutral and the remaining scores as an indicator for positive sentiment. Changing the parameters would hopefully give us a better result.

6.4 Comparative Analysis:

Tweets	Automated classifier	SVM			NB	DCNN
		SVC(kernel=rbf)	SVC(kernel=linear)	LinearSVC		
when terrorist sympathisers go basically unchallenged you know that islam really is the root of the problem.	neg	neg	neg	neg	neg	neg
do you see how beautiful this girl is??? like damn look at that highlight. you own that crown	pos	pos	pos	pos	pos	neg
yato is a faggot	neg	neg	neg	neg	neg	neg
you faggot head why dont you just die??	neg	neg	neg	neg	neg	neg
i'm sorry, i don't speak faggot tongue gayass bitch	neg	neg	neg	neg	neg	neg
i don't care what anyone says. no one knows justin personally. no one knows if he genuinely loves us, but i know i love him ripbeliebers	pos	pos	pos	pos	pos	pos
well, i guess we all are dead to him ripbeliebers beliebersnotfans	neu	neu	neu	neu	pos	pos
why u so fucking ugly?!	neg	neg	neg	neg	neg	neg
go home you fucking terrorist! banislam	neg	neg	neg	neg	neg	neg
shut your mouth, faggot. god emperor trump is speaking.	neg	neg	neg	neg	neg	neg
all you muslims should be deported. fucking jihadists banislam	neg	neu	neu	neu	pos	neu
let me make it simple for you; islam + muslims = jihad terrorism	neu	pos	neu	neu	neu	pos

"black people are ugly" now look at him	neg	neg	neg	neg	neg	neg
let the indictments begin	neu	neu	neu	neu	pos	neg
media has reincarnated itself into public court and has started interferring into court proceedings !!	neu	neu	neu	neu	neu	neu
Correct Prediction		13	14	14	12	10
Incorrect Prediction		2	1	1	3	5

Fig - 6.4.1 Sample prediction analysis of three machine learning approaches

From the table above, it is evident that SVM performed the best compared to Naive Bayes and Neural Network. Although we tested 3,500 tweets, we took a small sample of the test data to illustrate the differences in the classification results of the three algorithm approaches. The best performance was by LinearSVC and SVC(kernel=linear) as it was able to predict 14 tweets correctly out of 15. Even SVC(kernel=rbf) closely predicted the sentiment tweets compared to the bigram Naive Bayes approach. Naive Bayes was able to predict 11 tweets correctly out of 15. The lowest performance was by Convolutional Neural Network when compared to Naive Bayes and SVM as it predicted 10 correct tweets out of 15.

The accuracy of a classifier is vital to understand how effective it is in the prediction of sentiment to detect cyberbullying related tweets. Accuracy gives us the number of correct predictions made divided by the total number of predictions made, multiplied by 100 to turn it into a percentage. We measured the accuracy of each of the machine learning techniques in order to identify the most effective one of them all. For Naive Bayes and SVM we used NLTK's nltk.classify.accuracy and sci-kit's sklearn.metrics.accuracy_score to determine the classifier accuracy. For Neural Network we used the formula mentioned previously. The results of the percentage of accuracy is illustrated in the graph below:

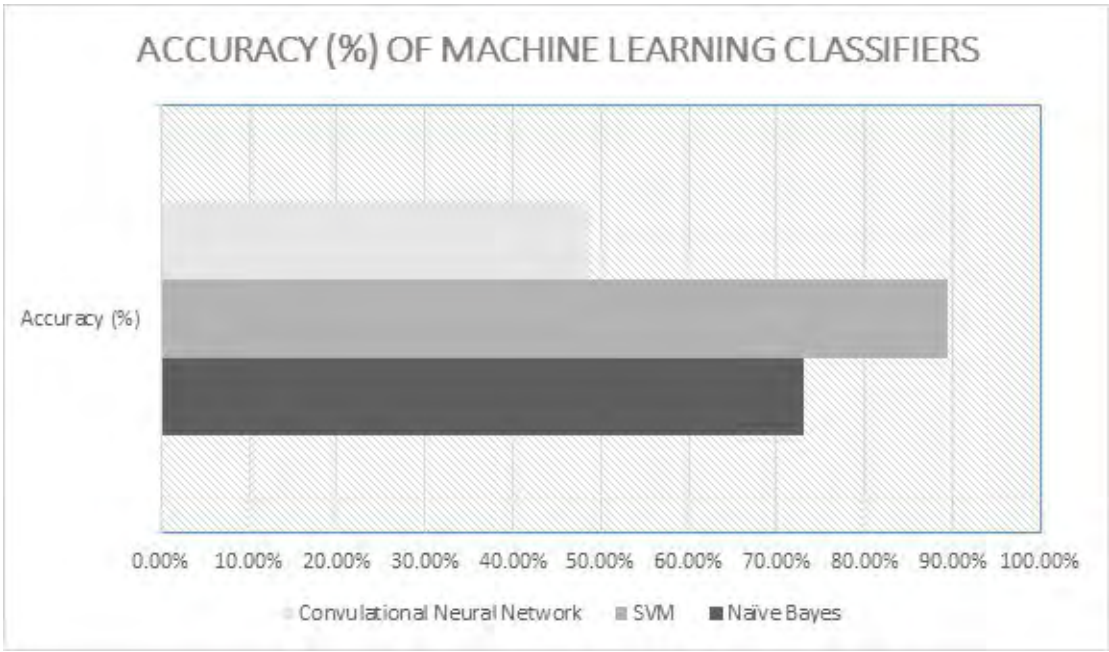

Fig - 6.4.2 Accuracy comparison of the three machine learning approaches

As shown above, the accuracy of SVM was the best when compared to other approaches - Naive Bayes being the second and Neural Network being the third. SVM was correctly able to predict sentiment with an accuracy of 89.39% and Naive Bayes having an accuracy of 73.0328%. The lowest performance was 48.6404% accuracy which was by Convolutional Neural Network Approach.

The reason for Convolutional Neural Network performing less could be because Neural Network usually requires a large amount of data when compared to Naive Bayes or SVM. This is why many don't prefer Neural Nets as it is very time consuming, take up lots of memory and also requires a huge dataset to actually provide relevant results. Another reason is that the algorithm has used only two convolutional layers which means that not enough features are being incorporated when calculating the sentiment

The Naive Bayes method provided satisfactory preliminary results compared to SVM. We believe that the average performance by Naive Bayes may be due to the kind of tweets that were being collected. On twitter, the tweet limit is 140 characters, so people have a very limited capacity to write grammatically correct tweets. People tend to use short forms and accidentally put whitespace in between words. Also the kind of tweets that were filtered where bullying related so bullies don't tend to proofread their sentences as they are insulting others out of anger, detestment, jealousy or hatred.

Furthermore, it has also been proved by many experimental results that SVM performs better than Naive Bayes in simple classification techniques and since we are also classifying based on positive, negative and neutral, SVM performed the best among all.

Naive Bayes seems to perform very well when the dataset is a corpus rather than tweets because the corpuses have better grammatical sentence structures when compared to the tweets retrieved, especially cyberbullying related tweets since these tweets contained lots of slang words. When our bigram classifier was tested with movie reviews corpus, it had an accuracy of 82.063%. Movie reviews were collected from nltk's existing corpus and were run through our classifier to check how other datasets performed.

Chapter 7: Limitations

A number of technical limitations have been observed in sentiment analysis. Some of the techniques we tried to achieve to make our algorithms give more accurate data could not be carried out due to lack of time. The main limitations of our experiment are listed below:

1. **Sentence Construction**

The way sentences are assembled in Tweets do not fully constitute formal language. They involve acronyms, emoticons, slangs. Since there is an increase of such slang in the social media, extraction of keywords become difficult.

2. **Separation of relevant objects and irrelevant objects.**

Object identification is vital and tricky. A blog or a tweet may have sentiments articulated revolving various matters. In such a case the problem lies in identifying the object on which a sentiment has been expressed without which the opinion is of little use

3. **Synonym grouping**

As people often use different words or phrases to describe the same feature, grouping synonyms is a challenging task.

4. **Opinion orientation classification**

There are an unlimited number of expressions that people use to express opinions, and in different domains they can be significantly different. Even in the same domain, the same word may indicate different opinions in different contexts. For example, in the sentence, “The battery life is long” “long”indicates a positive opinion on the “battery life” feature. However, in the sentence, “This camera takes a long time to focus”, “long” indicates a negative opinion.

5. **Sentences have implied meaning**

a sentence may not explicitly mention some pieces of information, but they are implied due to pronouns, language conventions, and the context. To deal with these problems, we need to apply NLP techniques such as parsing, word sense disambiguation and coreference resolution in the opinion mining context. Other challenges include domain dependency, irony and sarcasm

6. Privacy Concerns

Sentiment analysis or opinion mining involves the use of personal data of some kind and can lead to the disruption of some important normative values.

7. Unicodes to detect Emoticons

To detect emoticons in a tweet we did not use unicodes of emoticons but used regular expressions. It could not cover all the emoticons since some of the emoticons cannot be detected with regular expressions.

8. Pos tagging

We did not use part-of-speech tagged features for supervised analysis due to lack of time. Pos tagging is important because adjectives, adverbs and nouns in combination performs better than in individual features.

Chapter 8: Conclusions

8.1 Future Plan

Our ultimate goal of cyberbullying detection on social media is to flag as many online threats as possible so as to reduce manual patrolling efforts on social media. We aim to do this by focusing on further optimizing our precision and recall.

We also plan on further categorizing our polarity of positive, negative by assigning degrees to them i.e very positive, positive, very negative and negative. This will improve our research in terms of accurate detecting cyberbullying.

In addition to that, apart from assigning polarities, we want to classify our tweets according to the type of cyberbullying i.e threat, insult, curse etc. This will give us insight into which kind of bullying is more prominent over social media.

Furthermore, since we have not been able to implement PoS tagging, We want to combine this feature with machine learning by taking a lexical based approach to identify words and positive negative, and then feed to it our sentiment classifier will will classify our tweets [22].

Moreover, as implicit indication of cyberbullying is difficult to identify, we plan on exploring use of advanced features such as syntactic patterns and semantic information in addition to PoS tagging.

8.2 Conclusion

In our research, we have shown how each of the machine learning approaches perform in the detection of cyberbullying from social media using sentiments. The negative tweets indicated bullying tweets and hence we were able to detect successfully with the three approaches - among them, SVM performed the best and Naive Bayes performed second best. In this paper, we constructed a dataset of tweets containing cyberbullying and proposed and evaluated a methodology for adequate classification of this data. Additionally, we explored the feasibility of this automatic cyberbullying detection. We have fine tuned our simulation by running tests again and again to show the best results and our analysis showed the reason for each of their performance. With further work, we believe we can bring our research to perfection.

References

[1] W. D. Daelemans *et al.*, "Automatic Detection and Prevention of Cyberbullying," *The First International Conference on Human and Social Analytics*, 2015.

[2] T. Peacock, "Trump supporter arrested for Anti-Muslim terrorism plot," in *News*, Peacock Panache, 2015. [Online]. Available: http://www.peacock-panache.com/2015/12/william-celi-anti-muslim-terrorism-plot20914.html. Accessed: Aug. 16, 2016.

[3] 2014, "1.9. Naive Bayes — scikit-learn 0.17.1 documentation," 2010. [Online]. Available: http://scikit-learn.org/stable/modules/naive_bayes.html. Accessed: Aug. 16, 2016.

[4] Jacob, "Text classification for sentiment analysis – Stopwords and Collocations," in *python*, StreamHacker, 2010. [Online]. Available: http://streamhacker.com/2010/05/24/text-classification-sentiment-analysis-stopwords-collocations/. Accessed: Aug. 16, 2016.

[5] T. Peacock, "rump Supporter Arrested for Anti-Muslim Terrorism Plot," in *Peacock Panache*, Peacock Panache, 2015. [Online]. Available: http://www.peacock-panache.com/2015/12/william-celi-anti-muslim-terrorism-plot20914.htmlstream. Accessed: Aug. 16, 2016.

[6] H. Bouma, O. Rajadell, D. Worm, C. Versloot, H. Wedemeijer, “On the early detection of threats in the real world based on open-source information on the internet”, Int. Conf. Information Technologies and Security ITSEC, (2012).

[7] K. D. Dende, "Sentimental Analysis in crime detection: A case study of Kenya law enforcement agencies". [Online]. Available: http://erepository.uonbi.ac.ke/handle/11295/76651

[8] Bolla, Raja Ashok, "Crime pattern detection using online social media" (2014). Masters Theses. Paper 7321.

[9] 1.4. Support vector machines — scikit-learn 0.17.1 documentation, 2010 URL: http://scikit-learn.org/stable/modules/svm.html

[10] Scikit-Learn, "Sklearn.svm.SVC — scikit-learn 0.17.1 documentation," 2010. [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html.

[11] M. Bonzanini, "Sentiment analysis with python and scikit-learn," Marco Bonzanini, 2015. [Online]. Available: https://marcobonzanini.com/2015/01/19/sentiment-analysis-with-python-and-scikit-learn/.

[12] Scikit-Learn, "Sklearn.svm.LinearSVC — scikit-learn 0.17.1 documentation," 2010. [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html.

[13] B. Cheneke, "Regression and classification with Scikit-learn," 2016. [Online]. Available: http://beletecheneke.org/2016/03/regression-and-classification-with-scikit-learn/.
[14] Scikit-Learn, 2014, "RBF SVM parameters — scikit-learn 0.17.1 documentation," 2010. [Online]. Available: http://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html#example-svm-plot-rbf-parameters-py.
[15] Scikit-Learn, "Sklearn.metrics.precision_recall_fscore_support - scikit-learn 0.17.1 documentation," 2010. [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_recall_fscore_support.html.
[16] S. Ray, K. Jain, G. Blog, and M. Saraswat, "Understanding support vector machine algorithm from examples (along with code)," Analytics Vidhya, 2016. [Online]. Available: https://www.analyticsvidhya.com/blog/2015/10/understaing-support-vector-machine-example-code/
[17] Nal Kalchbrenner, Edward Grefenstette, Phil Blunsom, “A Convolutional Neural Network for Modelling Sentences”, 2014. [Online].
[18] Ronan Collobert *et al.* “Natural Language Processing (Almost) from Scratch” , 2011. [Online]. Available: http://jmlr.org/papers/volume12/collobert11a/collobert11a.pdf
[19] A. KOWALCZYK, "Linear kernel: Why is it recommended for text classification?," in Text classification, SVM Tutorial, 2014. [Online]. Available: http://www.svm-tutorial.com/2014/10/svm-linear-kernel-good-text-classification/.
[20] D. Britz, "Understanding Convolutional neural networks for NLP," in *WILDML*, WildML, 2015. [Online].Available:http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/. Accessed: Aug. 16, 2016.
[21] "Unsupervised feature learning and deep learning Tutorial," in UFDL. [Online]. Available: http://ufldl.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/. Accessed: Aug. 16, 2016.
[22] L. Liu, M. Lei, and H. Wang, "Combining domain-specific sentiment lexicon with Hownet for Chinese sentiment analysis," *Journal of Computers*, vol. 8, no. 4, Apr. 2013.
[23] "Efficient mapping of the training of Convolutional neural networks to a CUDA-based cluster," in*Parallel Architecture Research Eindhoven*. [Online]. Available: http://parse.ele.tue.nl/education/cluster2. Accessed: Aug. 16, 2016.
[24] M. A. Nielsen, "Using neural nets to recognize handwritten digits," Determination Press, 2015. [Online]. Available: http://neuralnetworksanddeeplearning.com/chap1.html. Accessed: Aug. 16, 2016.
[25] C. E. Corporation, "Basic concepts for neural networks," 1995. [Online]. Available: https://www.cheshireeng.com/Neuralyst/nnbg.htm. Accessed: Aug. 16, 2016.

[26] K. Van Cleemput, S. Bastiaensens, H. Vandebosch, K. Poels, G. Deboutte, A. DeSmet, and I. De Bourdeaudhuij, “Zes jaar onderzoek naar cyberpesten in Vlaanderen, Belgie en daarbuiten: een overzicht van ¨ de bevindingen. (Six years of research on cyberbullying in Flanders, Belgium and beyond: an overview of the findings.) (White Paper),” University of Antwerp & Ghent University, Tech. Rep., 2013.
[27] M. Price and J. Dalgleish, “Cyberbullying: Experiences, Impacts and Coping Strategies as Described by Australian Young People.” Youth Studies Australia, vol. 29, 2010, pp. 51–59, ISSN: 1038-2569
[28] V. Sl ˇ eglov ´ a and A. ´ Cern ˇ a, “Cyberbullying in Adolescent Victims: ´ Perception and Coping,” Cyberpsychology: Journal of Psychosocial Research on Cyberspace, vol. 5, 2011, ISSN: 1802-7962. [Online]. Available: http://cyberpsychology.eu/view.php?cisloclanku= 2011121901&article=4
[29] H. Vandebosch, K. Van Cleemput, D. Mortelmans, and M. Walrave, “Cyberpesten bij jongeren in Vlaanderen: Een studie in opdracht van het viWTA (Cyberbullying among youngsters in Flanders: a study commissoned by the viWTA). Brussels: viWTA,” 2006, URL: http:// ist.vito.be/nl/publicaties/rapporten/rapport cyberpesten.html [accessed: 2015-07-30].
[30] S. Hinduja and J. W. Patchin, “Bullying, cyberbullying, and suicide,” Archives of Suicide Research, vol. 14, 2010, pp. 206–221, ISSN: 1381- 1118.
[31] K. Van Royen, K. Poels, W. Daelemans, and H. Vandebosch, “Automatic monitoring of cyberbullying on social networking sites: From technological feasibility to desirability,” Telematics and Informatics, vol. 32, 2015, pp. 89–97, ISSN: 0736-5853.