

American International University-Bangladesh (AIUB)

# Conceptual Clarity and Terminology of Hate-speech Intensity Scale

full name (xx-xxxxx-x) full name (xx-xxxxx-x) full name (xx-xxxxx-x) full name (xx-xxxxx-x)

*A Thesis submitted for the degree of Bachelor of Science (BSc) in Computer Science and Engineering (CSE) at*

*American International University Bangladesh in December, 2021*

Faculty of Science and Technology (FST)

## Abstract

At the beginning time, emotion resolution or analysis is a method that is computing or expresses feelings, opinions, emotions, sentiments, and text subjectivity deeply from near the person's heart. In modern time’s people are mostly used different type’s social media platforms for expressing themselves in front of others with a variety of techniques, views, pictures, paintings, emoticons, and statements. Therefore the huge researcher gives their attention, and labor to detect the positive side of people’s emotion so that a little bit researcher or persons are giving their interest to detecting Hate or negative side of people’s opinion. Here, the main focusable contribution of this study is the detecting Hate speech and its four categories of hate emotions from English text data which is coming from after converting speech or voice to text using the Linear Support Vector Machine (SVM), the Long-Short Term Memory (LSTM), the Bidirectional Long-Short Term Memory (Bi-LSTM), the Decision Tree, and the Logistic Regression machine learning algorithms, or classifiers with various features such as Part-of-speech (POS) Tagging, Uni-gram, Bi-gram, Tri-gram, Term Frequency-Inverse Document Frequency (TF-IDF), and Word2Vec. So, using a labeled suitable dataset that is a converter from voice to text, the collection of the English text, or comments are curated for categorized four hate emotions before appreciating hate emotion from the text. The detected four hate emotions are Stupidity, Pathetic, Greedy, and Retarded. Applying 20485 training datasets and 6455 testing datasets among 26554 total datasets on five different approaches (e.g. Linear SVM, LSTM, Bi-LSTM, Decision Tree, and Logistic Regression), where it is shown the best accuracy that 87.07% from using the Bi-LSTM classifier whither Linear SVM gives 86.18%, LSTM gives 82.09%, Decision Tree gives 72.23%, and Logistic Regression gives 85.16% accuracy. Comparing other research our outcome is more efficient and noticeable also for future machine learning research, human, and social improvement.

## Declaration by author

This thesis is composed of our original work, and contains no material previously published or written by another person except where due reference has been made in the text. We have clearly stated the contribution of others to our thesis as a whole, including statistical assistance, survey design, data analysis, significant technical procedures, professional editorial advice, financial support and any other original research work used or reported in our thesis. The content of our thesis is the result of work we have carried out since the commencement of Thesis project.

We acknowledge that copyright of all material contained in my thesis resides with the copyright holder(s) of that material. Where appropriate we have obtained copyright permission from the copyright holder to reproduce material in this thesis and have sought permission from co-authors for any jointly authored works included in the thesis.

## Approval

The thesis titled **“Conceptual Clarity and Terminology of Hate-speech Intensity Scale”** has been submitted to the following respected members of the board of examiners of the department of computer science in partial fulfilment of the requirements for the degree of Bachelor of Science in Computer Science on **(date of defence)** and has been accepted as satisfactory.

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

#### MD. Kishor Morol

Lecturer & Supervisor

Department of Computer Science

American International University-Bangladesh

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

#### Dr. M M Manjurul Islam

Assistant Professor & External

Department of Computer Science

American International University-Bangladesh

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

#### Dr. Md. Mahbub Chowdhury Mishu

Associate Professor& Head (Undergraduate)

Department of Computer Science

American International University-Bangladesh

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

#### Professor Dr. Tafazzal Hossain

Dean

Faculty of Science & Information Technology American International University-Bangladesh

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

#### Dr. Carmen Z. Lamagna

Vice Chancellor

American International University-Bangladesh

## Contributions by authors to the thesis

List the significant and substantial inputs made by different authors to this research, work and writing represented and/or reported in the thesis. These could include significant contributions to: the conception and design of the project; non-routine technical work; analysis and interpretation of research data; drafting significant parts of the work or critically revising it so as to contribute to the interpretation.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Full Name** | **Full Name** | **Full Name** | **Full Name** | **Contribution (%)** |
|  | *xx-xxxxx-x* | *xx-xxxxx-x* | *xx-xxxxx-x* | *xx-xxxxx-x* |
| Conceptualization |  |  |  |  | 100 % |
| Data curation |  |  |  |  | 100 % |
| Formal analysis |  |  |  |  | 100 % |
| Investigation |  |  |  |  | 100 % |
| Methodology |  |  |  |  | 100 % |
| Implementation |  |  |  |  | 100 % |
| Validation |  |  |  |  | 100 % |
| Theoretical derivations |  |  |  |  | 100 % |
| Preparation of figures |  |  |  |  | 100 % |
| Writing – original draft |  |  |  |  | 100 % |
| Writing – review & editing |  |  |  |  | 100 % |

If your task breakdown requires further clarification, do so here. Do not exceed a single page.

## Acknowledgments

This respective project thesis is showing the result of our cooperative and sincere efforts, as well as our supervisor's uniqueness and never-ending inspiration. But at first, we would like to express our gratitude to Almighty for giving consent us to work on this project thesis for the past couple of semesters. Special gratitude goes to our honorable supervisor Md. Kishor Morol, Lecturer, American International University-Bangladesh (AIUB) for your kind support, encouragement, and resourceful discipline. Moreover, this project thesis was made possible by his great supervision and massive effort. We respectfully thank him for providing us his valuable time and effort with the opportunity to work with him. We also dispatch our gratitude to our honorable Vice Chancellor, Dr. Carmen Z. Lamagna, our Dean, Prof. Dr. Tafazzal Hossain, our Associate Dean, Mashiour Rahman, our Director, Dr. Dip Nandi, and our Head in charge, Dr. Md. Mahbub Chowdhury Mishu, Dr. M M Manjurul Islam Assistant Professor for their constant inspiration and support.

The last but not least, our sincere greetings is to go out to our respected parents who are provided opportunities for us to get educated and pursue our passions in order to do great things for humanity, and who encouraged us to think outside the box.

## Keywords

hate speech, toxic word; hate emotion, natural language processing ,text classifications, machine learning, social media, tweeter, hate comment, internet.

Table of Contents

[Title of the research i](#_Toc76225891)

[Abstract ii](#_Toc76225892)

[Declaration by author iii](#_Toc76225893)

[Approval iv](#_Toc76225894)

[Contributions by authors to the thesis -----------------------------------------------------------------------------------------------------------v](#_Toc76225901)

[Acknowledgments vi](#_Toc76225902)

[Keywords vi](#_Toc76225903)i

[List of Tables Vii](#_Toc76225904)

List of Abbreviations and Symbols Xii

[Introduction 1](#_Toc76225906)

* 1. [Hate speech -----------------------------------------------------------------------------------------------------------------------------------1](#_Toc76225907)
  2. [classes of hate speech ----------------------------------------------------------------------------------------------------------------------2](#_Toc76225907)
     1. [Stupidity ----------------------------------------------------------------------------------------------------------------------------------2](#_Toc76225907)
     2. [Pathetic ----------------------------------------------------------------------------------------------------------------------------------2](#_Toc76225907)
     3. [Greedy ------------------------------------------------------------------------------------------------------------------------------------3](#_Toc76225907)
     4. [Retarded ---------------------------------------------------------------------------------------------------------------------------------3](#_Toc76225907)
  3. [Objective --------------------------------------------------------------------------------------------------------------------------------------3](#_Toc76225907)
  4. [Aim ---------------------------------------------------------------------------------------------------------------------------------------------4](#_Toc76225907)
  5. [Contribution ---------------------------------------------------------------------------------------------------------------------------------4](#_Toc76225907)
  6. [Historical Background --------------------------------------------------------------------------------------------------------------------5](#_Toc76225907)
  7. [Problem Statement -------------------------------------------------------------------------------------------------------------------------6](#_Toc76225907)
  8. [Motivation -----------------------------------------------------------------------------------------------------------------------------------7](#_Toc76225907)
  9. [Scope 8](#_Toc76225907)

Literature review 10

[2 Related Work -------------------------------------](#_Toc76225909)-----------------------------------------------------------------------------------------10

[Methods 15](#_Toc76225910)

3.A Training Phase -------------------------------------------------------------------------------------------------------------------------16

3.B Testing Phase --------------------------------------------------------------------------------------------------------------------------17

3.1 Data Preparation --------------------------------------------------------------------------------------------------------------18

3.1.1 Data Collection ----------------------------------------------------------------------------------------------------------18

3.1.2 Data Filtering -------------------------------------------------------------------------------------------------------------19

3.2 Pre-processinh ---------------------------------------------------------------------------------------------------------------19

3.2.1 Lower Case Converter -------------------------------------------------------------------------------------------------20

3.2.2 Bad Symbol Remover --------------------------------------------------------------------------------------------------20

3.2.3 Emoticon Remover -----------------------------------------------------------------------------------------------------21

3.2.4 Stop-word Remover --------------------------------------------------------------------------------------------------22

3.2.5 Punctuations Remover ----------------------------------------------------------------------------------------------23

3.2.6 Tokenization -------------------------------------------------------------------------------------------------------------24

3.2.7 Stemming ------------------------------------------------------------------------------------------------------------------24

3.3 Feature Selection & Extraction -------------------------------------------------------------------------------------25

3.3.1 POS Tagging ------------------------------------------------------------------------------------------------------------26

3.3.2 Unigram, Bi-gram and Tri-gram ------------------------------------------------------------------------------27

3.3.3 TF-IDF ----------------------------------------------------------------------------------------------------------------------29

3.3.4 Word2Vec --------------------------------------------------------------------------------------------------------------30

3.4 Classification --------------------------------------------------------------------------------------------------------------31

3.4.1 Linear SVM -------------------------------------------------------------------------------------------------------------31

3.4.2 LSTM ---------------------------------------------------------------------------------------------------------------------33

3.4.3 Bi-LSTM ------------------------------------------------------------------------------------------------------------------34

3.4.4 Decision Tree --------------------------------------------------------------------------------------------------------35

3.4.5 Logistic Regressio ------------------------------------------------------------------------------------------------37

[Results or findings](#_Toc76225912) --------------------------------------------------------------------------------------------------------------------------------39

[4 Results 39](#_Toc76225913)

[Discussion 42](#_Toc76225914)

[5.1 Comparison 42](#_Toc76225915)

[Conclusion 4](#_Toc76225916)5

[Bibliography 46](#_Toc76225917)

**List of Figures**

[Figure 3. 3: System Overview](#_Toc76225927) -----------------------------------------------------------------------------------------------------------------15

[Figure 3. 3-A: Architecture of Training Phase --------------------------------------------------------------------------------------------------17](#_Toc76225928)

[Figure 3. 3-B: Architecture of Testing Phase ----------------------------------------------------------------------------------------------------18](#_Toc76225928)

[Figure 3. 3-1: Lower Case Converter --------------------------------------------------------------------------------------------------------------20](#_Toc76225928)

[Figure 3. 3-2: Bad Symbole Remover ---------------------------------------------------------------------------------------------------------------21](#_Toc76225928)

[Figure 3. 3-3: Emoticon Remover -------------------------------------------------------------------------------------------------------------------22](#_Toc76225928)

[Figure 3. 3-4: Stop-word Remover -----------------------------------------------------------------------------------------------------------------22](#_Toc76225928)

[Figure 3. 3-5: Punctuations Remover --------------------------------------------------------------------------------------------------------------23](#_Toc76225928)

[Figure 3. 3-6: Tokenization ---------------------------------------------------------------------------------------------------------------------------24](#_Toc76225928)

[Figure 3. 3-7: Stemming -------------------------------------------------------------------------------------------------------------------------------25](#_Toc76225928)

[Figure 3. 3-8: POS Tagging ---------------------------------------------------------------------------------------------------------------------------27](#_Toc76225928)

[Figure 3. 3-9: N-gram ---------------------------------------------------------------------------------------------------------------------------------28](#_Toc76225928)

[Figure 3. 3-10: TF-IDF ---------------------------------------------------------------------------------------------------------------------------------30](#_Toc76225928)

[Figure 3. 3-11: Word2Vec ----------------------------------------------------------------------------------------------------------------------------31](#_Toc76225928)

[Figure 3. 3-12: Linear Support Vector Machine (SVM) -----------------------------------------------------------------------------------------33](#_Toc76225928)

[Figure 3. 3-13: Long-Short Term Memory (LSTM) ----------------------------------------------------------------------------------------------34](#_Toc76225928)

[Figure 3. 3-14: Bidirectional Long-Short Term Memory (Bi-LSTM) -------------------------------------------------------------------------35](#_Toc76225928)

[Figure 3. 3-15: Decision Tree ------------------------------------------------------------------------------------------------------------------------37](#_Toc76225928)

[Figure 3. 3-16: Logistic Regression -----------------------------------------------------------------------------------------------------------------38](#_Toc76225928)

[Figure 4. 4-1: Detect Hate Emotion (Stupidity) ---------------------------------------------------------------------------------------------------40](#_Toc76225928)

[Figure 4. 4-2: Detect Hate Emotion (Greedy)](#_Toc76225928) ------------------------------------------------------------------------------------------------------41

# List of Tables

[Table 3. 3-A: Class distribution in the Dataset 1](#_Toc76193928)9

[Table 3. 3-B: Uni/ Bi/ Tri- gram](#_Toc76193929) -----------------------------------------------------------------------------------------------------------------------28

[Table 4. 4-A : Detailed evaluation using best model ---------------------------------------------------------------------------------------------- 39](#_Toc76193930)

[Table 4. 4-B: Sentence split and accuracy ---------------------------------------------------------------------------------------------------------- 40](#_Toc76193930)

[Table 4. 4-C: Classifiers ---------------------------------------------------------------------------------------------------------------------------------41](#_Toc76193930)

[Table 5. 5-A: Comparison with relevant state of art work 43](#_Toc76193930)

# List of Abbreviations and Symbols

Abbreviations

NPL Natural Language Processing

DNN Deep Neural Network

ML Machine Learning

SVM Support Vector Machine

LSTM Long Short Term Memory

Bi-LSTM Bidirectional Long Short Term Memory

POS Tagging Parts-of-speech Tagging

TF-IDF Term Frequency – inverse Document Frequency

e.g. ForExample

Symbols

*%* Percentage

+ Positive

& And

\* Multiplication

= Equal

e.g. For example

**Chapter 1**

# Introduction

The detection of Hate Speech from speech to text is becoming an interesting as well as an important part of Natural Language Processing in the Machine Learning Domain in recent decennia. In the past few years, researchers give their interest in detecting Sentiment or Emotion from the text. Here, emotion categorization is the process where it is indicated that studying a speaker's expression to determine if he or she has a positive or negative attitude about a certain issue. However, this concept has become important for researchers because they realize that the impression of this topic and its bright outcome. Working with this concept is often difficult for everyone when the person must have to reach an actual agreement that exactly to identify which type of speech acts under hate speech topic.

* 1. **Hate Speech:**

Before describing the whole paper, one must have to be familiar with the word Hate Speech. “Hate Speech” is a pathological concept and also a category of Emotions. There has not any universally adopted definition of it in International Human Rights Law. But some people are tried to define it according to their perception. Therefore, the definition of “Hate Speech” it might be a more complex concept which is composed of two basic concepts, they are Hate and Speech. Consequently, it can be split into two main components according to give basic concept:

* **Hate:** Mainly, Hate word indicates, the intense and irrational emotion of opprobrium, enmity, and detestation towards an individual or group, targeted because of their having certain – actual or perceived – protected characteristics. In additionally "Hate" is more than preference, and must be so discriminative. In short, Hate is a gesture of an emotional state, opinion, or thought, and therefore it is separate from any manifests action or act.
* **Speech:** Here,theSpeech word indicates any expression which is imparting opinions, action, or ideas that bring subjective opinions, action, or ideas to an external audience, or listener. It can look, and take many forms such as written, non-verbal, visual, or artistic, and can be propagated through any kind of media, like the internet, print, radio, or television.

After going through these two basic elements, and to put it simply, one can say that “Hate Speech” is a type of emotional expression which indicates hate towards people.

* 1. **Classes of Hate Speech:**

As this paper is built for describing detected hate, contempt, or toxic speech from voice to text which are tested by different methodology and algorithms, one must have to be familiar with what kinds of Hate Speech are detected. In shortly, receivers have to be familiar with the classes of Hate Speech that are detected in this whole research. So, here is a short description of the classes of Hate Speech;

* + 1. **Stupidity:**

At first, the definition of Stupidity is that a person with his or her less than normal intelligence, sagacity, or a boring situation. Mainly, stupidity indicates lacking intelligence, knowledge, or common sense of a person. The example of this like,

* An adult person who does not know or give an answer of 1+2 is equal. Or
* For unnecessary work, a person wants to make conversation ingeminate at any time.
  + 1. **Pathetic:**

The definition of Pathetic is alluding that someone, or something which is brined or is capable of bringing about feeling a pity, sadness, sympathy, or sorrow. In another way, after seeing something a person feels sadness, pity, or miserable, this is called pathetic. Sometimes it occurs for lack of respect. The example of Pathetic is like,

* A cat is sitting with mange in road-site. Or,
* An actor is forgotten these lines while he acts on stage.
  + 1. **Greedy:**

Greedy is an uncontrolled emotion of a person. The definition of Greedy is that it is an uncontrolled desire for an increase in the achievement, or use of material gain, or social value. Here social values such as status, or power. In another way, Greedy can be identifying as undesirable throughout known person history because it is generated behavior conflict between personal, moral, and social goals. The example of Greedy is like,

* For building a modern shopping center, builders cut down a lot of trees.
* To achieve more profit, the shopkeeper mixed adulteration on his products.
  + 1. **Retarded:**

Retarded is a classification of hate word. Definition of Retarded is a situation while someone feels insulting, humiliating, and inappropriate for any kind of action. The person acts like he has less advanced, or ahead mentally than other people of his or her age, or generation. Retarded hate emotion example is like,

* A person asks a question that why garbage smells is so unhealthy.
* You are acting like a fool.
  1. **Objective:**

This paper proposes a process of “Hate” emotion analysis of text data from the voice which is written in the English language. This paper will be shown four types of Hate Speech (stupidity, pathetic, greedy, and retarded) from a text, line, or paragraph with previously used methodology and new suggest algorithms. This process can automate the analysis of a user's reaction towards a specific emotion like news-paper, book, or social media. With more and more people expressing their antithetic opinions openly on social networking sites, analyzing the sentiment of comments made about a specific text, movie and advertisement indicate how they feel. This paper will be accentuation some objectives like:

* This research proposes the detection of the four hate emotions namely stupidity, pathetic, greedy, and retarded for English text or speech.
* Supervised and unsupervised learning methodology can easily be applied to English text for hate or detest emotion analysis or opinion mining for the specific domain of the text.
* Ambiguous and unambiguous despise emotions for English text can be handled using the semi-supervised methodology.
* One more direction is, based on English text, the identification of overlapped comments like user's comments on each other about a subject and dual emotion detection as like the same comment is stupidity for one and greedy for another people.
* The project can score multiple dislike emotions at a time with a percentage.
* It can also work to remove punctuation with emoticons, sentence splits.
  1. **Aim:**

In shortly, the main target of this research is to create, or generate, and transform, or variation data corpus to suitable document establishing a model which can be representing numeric vector, and which can be also analyzing different machine learning techniques to evaluate the performance and accuracy of the classifiers in the context of detecting Hate Speech from the speech to the text. The aim of this study can be pointed such as –

* Create a standard English Hate Speech classification corpus.
* Categorizing documents according to selective human opinions from the text.
* Construct document embedding model representing numeric vectors to work with standard and accurate Machine Learning algorithms.
* Analyzing performance of Deep Learning and Traditional Machine Learning approaches for detecting Hate Speech analysis.
  1. **Contribution:**

For the time being, "Hate Speech" is a type of speech that can act as an invective communication mechanism in all types of environments such as online social media and also realistic society. Hate speech can be target different kinds of social characteristics suchlike gender, religion, race, environment, disability, and so on. Therefore, according to improve previous research and today's position, here some contributions will be described which are completed in this paper or research.

* At first, the noticeable thing is comparing the effectiveness of fully, or absolute character-based models, with word + character or letter embedding models, subWord models, and byte pair encoding for showing which of the techniques are performed better than previously used word-based models.
* Then applied and compare a range or compass of strong or potent classifiers to find a new brilliant accuracy with a multilevel dataset which is around 26,954 user comments from Twitter and validate or make out the results on a different domain.
* Also applied two types of different pertained word embedding for the domain or state of user comments which are taken from Tweets to recompense or compensate errors, and deficiency. For example, Idiosyncratic, personal, intrinsic and misspelled words, and texts.
* Examine or tested how pre-processing documents like used classifiers, and methodology with advanced, best, and right approaches. The output performance is very noticeable rather than the previously acquired result.
  1. **Historical Background:**

This concept is largely motivated by numerous articles that attempted to categorize various groups, such as aggression and emotions. And it is one of the important works that have been following. For highly resourced languages like English, Arabic, and other European languages, emotion analysis or hate-speech detection from text is a well-studied research topic. Recent research has placed a greater emphasis on acknowledgment. A tremendous rise in the internet and social networks usages has been seen for a couple of years. But unfortunately, the increase of the dark side in toxic speech is greater than the absolute side. Toxic or hate speech can be defined as feisty and uncertain, abusive or threatening speech or writing that expresses against a particular group based on gender discrimination, religion, nation, denomination, and so on. The interest of research on hate speech begins in 2014 (from 2013 to 2018) which is revealed in the volume of Web of Science (WoS) – indexed production [1]. In this time another author uses semantic content analysis using Natural Language Processing (NLP) and Deep Neural Networks (DNN) methodologies for detecting non-toxic and toxic words [2]. Using those methodologies, they found their result 88% and 78% (for non-toxic using NPL and DNN) and 6.5% and 10.9% (for toxic using NPL and DNN) with 200 datasets [2]. Observing 104 articles a group of author’s works on racism and hate speech around 500 words as a result they find 40.38% results by using qualitative methods and 35.58% results by using quantitative methods in different geographic regions from social media platforms [3]. Another crew works to discover hate speech and international legislation in four countries (Egypt, Kenya, Serbia, and South Africa) with BERT fine-tuning, fastText embedding, and BERT embedding methodologies. As a result they find accuracy nearly 78.0%, 75.7% and 76.2% [4]. After that, we find some other researchers used BOW and SVM methodologies for clearly outperforming results such as 0.3140% and 0.7941%. Where SVM gives, the highest level of accuracy about 0.4201% [5]. Afterward, another troupe authors are seen to grant interest to find out hate speech from tweeter data with 860 tweets using HASOC (only) and HASOC (offensEval) which gives result 0.7945% and 0.7976% in Macro F1-score. On other hand in Weighted F1-score gives by using HASOC (only) and HASOC (offensEval) which gives results of 0.8426 % and 0.8504% [6]. From social network sites other group searcher works on Recurrent Neural Network named Long Short Term Memory (LSTM) and Support Vector Machine (SVM) where SVM is given strong performance rather than LSTM. In this observation, they find 410 weak hate, 130 strong hate from 540 words data set [7]. From Twitter and Wikipedia, a couple of researchers used word-based, subword-based, joint word & character and embedding methodology for detecting hate speech. As a result they find a good result with 83.0%, 71.2%, 64.1% and 71.5% [8]. On 17 June 2021, a fellowship investigator is used CNN-GRU model, Long Short-Term Memory (LSTM) network, Gradient-Boosted Decision Tree (GBDT) to compare their result from others [9]. After that a few, researchers are continuing this type of research from serval social networks such as Yahoo (950K), Twitter (160K), news (112K), Wikipedia (115K), Facebook (300K). But they only work on detecting three types of emotion such as threat, identity hates, and insult with 78%, 88%, 43% using CNN, LSTM, and GRU methodology [10].

* 1. **Problem Statement:**

Before all as a human begin, they have six emotions. They are happiness, sadness, surprise, fear, anger, and disgust. After observing related works we can understand that most of the authors work on text analysis in different platforms with different approaches. Some of them are given their labor to discover human emotion (Positive, Negative, Happy, Sad, Surprise, Disgust, Fear, Anger) from text which belong to a different platform (News-paper, Book, Article, Social media). However, most of the authors or researchers are worked on only three emotions (Positive, Negative, Neutral) with a big dataset which gives them a huge success. But to detect hate speech a low number of authors are seen to give interest. The main reason is no one prefers hate speech. Moreover, a group of searchers works on detect hate speech from a text in a different area. But this does not give them too much success. In such type of document, the problem statement does not always remain so clear. It can be classified into various kinds of different problems and get successful analysis of hate emotion depends on a lot of issues including (but not limited to it):

* Sometimes such texts and speeches contain multiple sentiments related to two or more than two issues.
* Sometimes such documents contain both kinds of hate sentiments. i.e., negative and positive both. Here, the identification of the most effective one is a major issue.
* In some cases, the problem can be converted into multi-subjective hate sentiment analysis.
* Once in a while, some words or sentences stupid and disgust works look similar. So, both are difficult to identify the accurate meaning.
* On-time long sentences are created difficult to detect the real emotion that what is it indicates.
* Punctuation marks are created difficult for using algorithms.
* Proper and accurate methods and algorithms are not used properly. The used deficient dataset also create problem to detecting Hate speech.
  1. **Motivation:**

The use of the Internet and social networks or media is increasing day by day tremendously which is must noticeable for all people. Besides that, the growth of the dark side of social networks or media is increased faster. Here the dark side means Hate Speech or Toxic Speech. As already know that Hate Speech has no uniform definition in Scientific Literature, and also have not any clear difference between hate, offensive, toxic, and abusive speech. So, this paper represents Hate emotion analysis from English text from Twitter data. It is the most important area of Natural Language Processing and Deep Learning. Generally, Hate emotion or speech state or rank means the resolution to determine any expression of a user or a speaker where he or she holds, and express a negative opinion on a specific topic, text, or subject. Hate speech resolution can greatly walk out for everyone in their real-life with the quick growth of E-commerce. As an example, for prototype, any types of product reviews on the Social media, and Web is lead an important source of information for patron’s decision in any element when anyone wants to buy somewhat product. As a result, the reviews are hourly to multifarious for patrons to go through that how automatically it classify, detect, and discover the Hate speech emotion from them has driven an important exploration problem. According to this oath, this paper presents Hate speech detection which is recognizes the English Hate emotion or opinion about a subservient from the English Tweets primer. So, here the most noticeable thing is that it is constructing with some idiom patterns, and calculates Hate emotion disclosure. Moreover, Social media or Networks are playing the principal fund to collect information about people’s expressions, passions, and opinions which towards different kinds of questions or queries as they put out hours daily on social media and give, and take their opinion. Twitter data is playing the role of Social media or Networks. Hence, this whole paper shows the employment of Hate emotion detection and how it is connected to kind of Twitter and move hate emotion analysis queries. After running essays on different types of questions from politics to humanity and show the absorption results. Realizing that, the Hate speech emotion or opinion for tweets is the most significant gives a high position which shows the limitations of the present manufactory. Therefore the research is more monumental for

* This presented a manually annotated English Hate Speech emotion corpus, which is formless, the diversity of fine- coarse emotion expressions in social-media or network handbooks.
* To employ classical machine-learning approaches that are normally perform well in classifying the six aforesaid Hate Speech emotion types.
* Compare the machine-learning amiable performance with alpha to identify the best-performing model for identifying hate speech emotion brackets.
* The pre-processing of the data in a way so that it's readily usable by experimenters.
* Employment of deep continual models on an English Hate Speech and Romanized English Hate Speech handbook corpus.
* Then the pre-train dataset of one ticket for another (vice versa) to prove its use.

So, after observing these a decision comes that to do discovered on this research topic. And, it can help a lot with its important area to do piercing.

* 1. **Scope:**

Now a day’s social media plays a very important role in people’s regular life. Through social media, people can directly share their own opinion, thoughts, mind, view, memories, facts, and emotions with others in a short time with their culture, plane, and traditions. Because of dependent social media, and networks, Cyber-crime is arising frequently. Cyber-crime and Cyber-bullying are on the upturning as a result it gives the tumid behaviour of social media and networking sites and found that the majority of the victims are girls and women. For this reason, people are sharing their thought and emotions by texting social media. So, the vertical methods have a lot of strength for detecting text. That is why text analysis is the more difficult commitment in the English Language especially detecting Hate speech text from people's comments. There have huge computational techniques which connected directly to Machine Learning and it put numerical attention into its field in recent times. Therefore Hate Speech emotion analysis is also repeatedly utilized in psychological, deep learning, and machine learning studies. However, it is mostly used to recognize or figure out what a person hates assuming exponent is. So that this paper is trying to help and detected hate opinions of a person which is he or she expressed in his or her text in his or her lifestyles. For this reason, it is further used to residing hate speech emotional marketing. Besides, it indicates which kinds of people are purchased, and which type of product with which hate opinion. But detecting linguistic hate emotion is too much difficult for people, researchers, and machines also. Whatever, detecting linguistic hate emotion is highest significant Natural Learning Mechanisms for develop, and expand the Hate Speech, emotional models.

**Chapter 2**

# Literature review

Now a day, Hate speech concept or detecting has become more an attractive topic within researchers whereas, most of the researcher is giving priority to detecting Sentiment Analysis and Emotion Analysis in different ways like in a variety of linguistic domains, including English, French, Chinese, Arabic, and others languages. However, various works are published on the topic of Hate Speech which is related to detection, identification, and characterization from text which is collected from several social media.

So, this concept was largely motivated by numerous articles that attempted to categorize various groups, such as aggression and emotions. And it's one of the important works that have been following. For highly resourced languages like English, Arabic, and other European languages, emotion analysis or hate-speech detection from text is a well-studied research topic. Recent research has placed a greater emphasis on acknowledgment. A tremendous rise in the internet and social networks usages has been seen for a couple of years. But unfortunately, the increase of the dark side in toxic speech is greater than the absolute side. Toxic or hate speech can be defined as feisty and uncertain, abusive or threatening speech or writing that expresses against a particular group based on gender discrimination, religion, nation, denomination, and so on.

Due to some technical and empirical constraints, however, the extent of its progress in Hate Speech detection is insignificant. The interest of research on hate speech begins in 2014 (from 2013 to 2018). “Maria Antonia Paz, Julio Montero Diaz, Alicia Moreno Delgado” worked on “Hate Speech: A Systematized Review” [1] which is revealed for discussion the importance of interdisciplinary and mapping of hate speech in the volume of Web of Science (WoS) – indexed production in different languages like USA, England, Australia, Canada, Spain, Germany, South-Africa. Mainly, they did not apply any kind of methodology; they focused on the only discussion of the importance of hate speech in different languages.

In this time “Ashwin Geet D’sa, Irina Illina, Dominique Fohr” uses semantic content analysis using different kinds of Deep Neural Networks (DNN) classifiers for detecting non-toxic and toxic words on “Towards Non-Toxic Landscapes: Automatic Toxic Comment Detection Using DNN” [2]. Using those classifiers, they found their result 88% and 34.1% (for non-toxic to toxic using Binary Classification and Regression Model) and 6.5% and 7.6% (for toxic to non-toxic using Binary Classification and Regression Model) with 200 datasets. The best result of their work is BERT fine-tuning which gives an accuracy of about 78.2%.

Observing 104 articles “Ariadna Matamoros Fernandez and Johan Farkas” works on “Racism, Hate Speech and Social Media: A Systematic Review and Critique” [3] around 500 words as a result they find the accuracy about 40.38% by using Deep Neural Networks (DNN) classifiers like Convolutional Neural Network (CNN) and another accuracy 35.58% by using quantitative classifiers like Recurrent Neural Network (RNN) in different geographic regions from social media platforms. So, DNN gives more efficient accuracy than other approaches. The accuracy difference is about 4.8%.

“Sergio Andres, Natalia Suarez, and Luz Magnolia” propose “Internet, social media and online hate speech. Systematic review”[4].Where they work to discover hate speech and international legislation in four countries (Egypt, Kenya, Serbia, and South Africa) with Bidirectional Encoder Representations from Transformers (BERT) fine-tuning, fastText embedding, and BERT embedding classifiers. As a result, they find accuracy nearly 78.0%, 75.7%, and 76.2%. Here BERT fine-tuning gives a suitable accuracy from other presence.

After that, “Salla-Maaria Laaksonen, Jesse-Haapoja, Teemu Kinnunen and Matti Nelimarkka” used BOW and SVM methodologies in “The Datafication of Hate: Expectations and Challenges in Automated Hate Speech Monitoring” [5]. Using Bag-of-words (BOW) and Support Vector Machine (SVM) methodologies on detecting non-hate speech messages and hate speech messages are clearly outperforming accuracies such as 0.3140% and 0.7941%.Where SVM gives, the highest level of accuracy about 0.4201% from BOW.

Afterward, another troupe author named “Pedro Alonso, Rajkumar Saini and Gyorgy Kovacs” propose “Hate Detection using Transformer Ensembles on the HASOC dataset” [6]. Authors are seen to grant interest to find out hate speech from tweeter data with 860 tweets using HASOC (only) and HASOC (offensEval) which gives results 0.7945% and 0.7976% accuracy in Macro F1-score. On other hand in Weighted F1-score gives by using HASOC (only) and HASOC (offensEval) which gives results of 0.8426 % and 0.8504% accuracy. Seeing their obtained accuracy they understand that HASOC (offensEval) gives the best output from other approaches.

Message from social network sites “Fabio Del, Andrea Cimino, Felice Dell, Marinella and Maurizio” searchers are worked on “Hate me, hate me not: Hate speech detection on Facebook” [7]. Using Recurrent Neural Network (RNN) classifier named Long Short Term Memory (LSTM) and Support Vector Machine (SVM) where SVM is given strong performance rather than LSTM. In this observation, they find 410 weak hate words, 130 strong hate words from 540 words data set.

“Sravan Bodopati, Spandana Gella, Kasturi Bhattacharjee and Yaser Al-Onaizan” are used Twitter and Wikipedia for research their paper named “Neural Word Decomposition Models for Abusive Language Detection” [8]. Those researchers used many types of classifiers, and methods like word-based, subword-based, joint word & character and embedding methodology for detecting hate speech from text which is found online. As a result, they find a good accuracy with 83.0%, 71.2%, 64.1%, and 71.5%. After finding that accuracy they fathom that 83.0% is the highest accuracy for them and other researchers.

On 17 June 2021, “Wenjie Yin and Arkaitz Zubiaga” named fellowship investigator who is published a paper named “Towards generalizable hate speech detection: a review on obstracles and solutions” [9]. Here they are detected harmful types of words that are used online or directly attack people and also indicate hate toward with a short dataset of nearly 475 words. For detecting they used Convolutional Neural Network - Gated Recurrent Unit (CNN-GRU model), Long Short-Term Memory (LSTM) network, Gradient-Boosted Decision Tree (GBDT) to compare their result from others. Among those models or method, Gradient-Boosted Decision Tree (GBDT) gives a fine accuracy which is about 65%.

After that, “Julian Riseh and Ralf Krestel” gives their work and published a paper named “Toxic Comment Detection in Online Discussion” [10]. Those researchers are continuing this type of research from serval social networks such as Yahoo (950 words), Twitter (160 words), news (112 words), Wikipedia (115 words), Facebook (300 words). Mainly Julian and Ralf are tried to detect hate speech with its categories. They are the first researcher who researches such kinds of classification and detection. But they only work on detecting three types of emotion such as threat, identity hates, and insult with 78%, 88%, 43% using Convolutional Neural Network (CNN), Long Short Term Memory (LSTM), and Gated Recurrent Unit (GRU) methodology. Long Short Term Memory (LSTM) gives maximum accuracy among other methodologies.

Most recent year “Ashwin Geet d’Sa, Irina Illina and Dominique Fohr” authors are giving their best efforts finding hate words from news-paper. They published a paper named “Classification of Hate Speech Using Deep Neural Networks” [11]. There they use Support Vector Machine (SVM), Convolutional Neural Network (CNN), Bi-Directional Long Short Term Memory (Bi-LSTM), and Convolutional Recurrent Neural Network (CRNN) classifiers with fast-Text for automatic detection of hate speech from online. Mainly they perform classification using two approaches, one is SVM and CNN. Another one is fine-tuning with Bi-LSTM and CRNN. Using those approaches they have achieved an excellent accuracy which is 65.8%, 89.2%, 72.3%, and 62.01%. Those results are better than other research results. Convolutional Neural Network (CNN) gives a very good accuracy from all methods or approaches and differences are more noticeable.

Recent century “Radu Meza, Hanna Orsolya and Andreea Mogos” works to identify detest words from different languages such as Romania, French, Korean and Nepali with multi-layer perceptron. They proposed a paper which is “Targets of Online Hate Speech in Context. A Comparative Digital Social Science Analysis of Comments on Public Facebook Pages from Romania and Hongary” [12]. Gated Recurrent Unit (GRU), Long Short Term Memory (LSTM), and Fast-Text are used in this paper. From Romania, they find 48%, 55%, 32%, and 78% accuracy using those methods. From the French language 88%, 67%, 45%, and 88% results are finding from this method. In Korean language 23%, 76%, 55% and 45% are found. At last from Nepali 89%, 65%, 48% and 67% accuracy found.

Most recently on 30 October 2020 four researchers named “Alice Tontodilmamma, Eugenia Nissi, Annalina Sarra, and Lara Fontanella” give their exertion for finding hate speech from user comments which indicates online text in the “Thirty years of research into hate speech: topics of interest and their evolution” [13] paper. The detecting accuracy of hate speech from social media based on Support Vector Machine (SVM), Convolutional Neural Network (CNN), and Uni-gram methodology are about 77%, 56%, and 58%. Here Support Vector Machine (SVM) gives an outstanding accuracy.

“Betty Van Aken, Jolian Risch, Ralf Krestel and Alexander Loser” are almost the first authors who have used Corpus and Uni-gram methodology for their research named “Challenges for Toxic Comment Classification: An In-Depth Error Analysis” [14]. From Yahoo, Twitter, news, Wikipedia, Facebook, Whatsapp, Instagram a group of people are shown interest to identify hate speech by using Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), Logistic regression, recurrent neural networks. Observing and calculating those method they find a favorable result with an excellent accuracy such as .73%, .86%, .776%, .981% for Yahoo, .70%, .85%, .748%, .979% for Twitter, .71%, .85%, .752%, .978% for news, .74%, .84%, .777%, .980% for Wikipedia, .71%, .86%, .755%, .979% for Face-book. 72%, .86%, .765%, .981% for Whatsapp and last .70%, .88%, .791, .983% for Instagram . Those authors are work on several social media which are not worked before and finds a strong result using those algorithms.

With multiple algorithms, four researchers are “Andrea Cimino, Felice Dell, Marinella, Fabio Del, and Maurizio” spend their knowledge on finding hate speech in “Hate me, hate me not: Hate speech detection on Facebook, Twitter, News” [15]. Their using methods are lexical resources, logistic regression, LSTM classifier they achieved 90%, 76%, and 88% accuracy.

Using the Tweet dataset, “Thomas Davidson, Dana Warmsley, Michael Macy and Ingmar Weber” are proposed a paper, named “Automatic Hate Speech Detection and the Problem of Offensive Language” [16]. There they find 40% of hate speech is misclassified from using Support Vector Machine (SVM), 0.91% is found from N-gram, 0.90% is observed from matrix results are found in research papers.

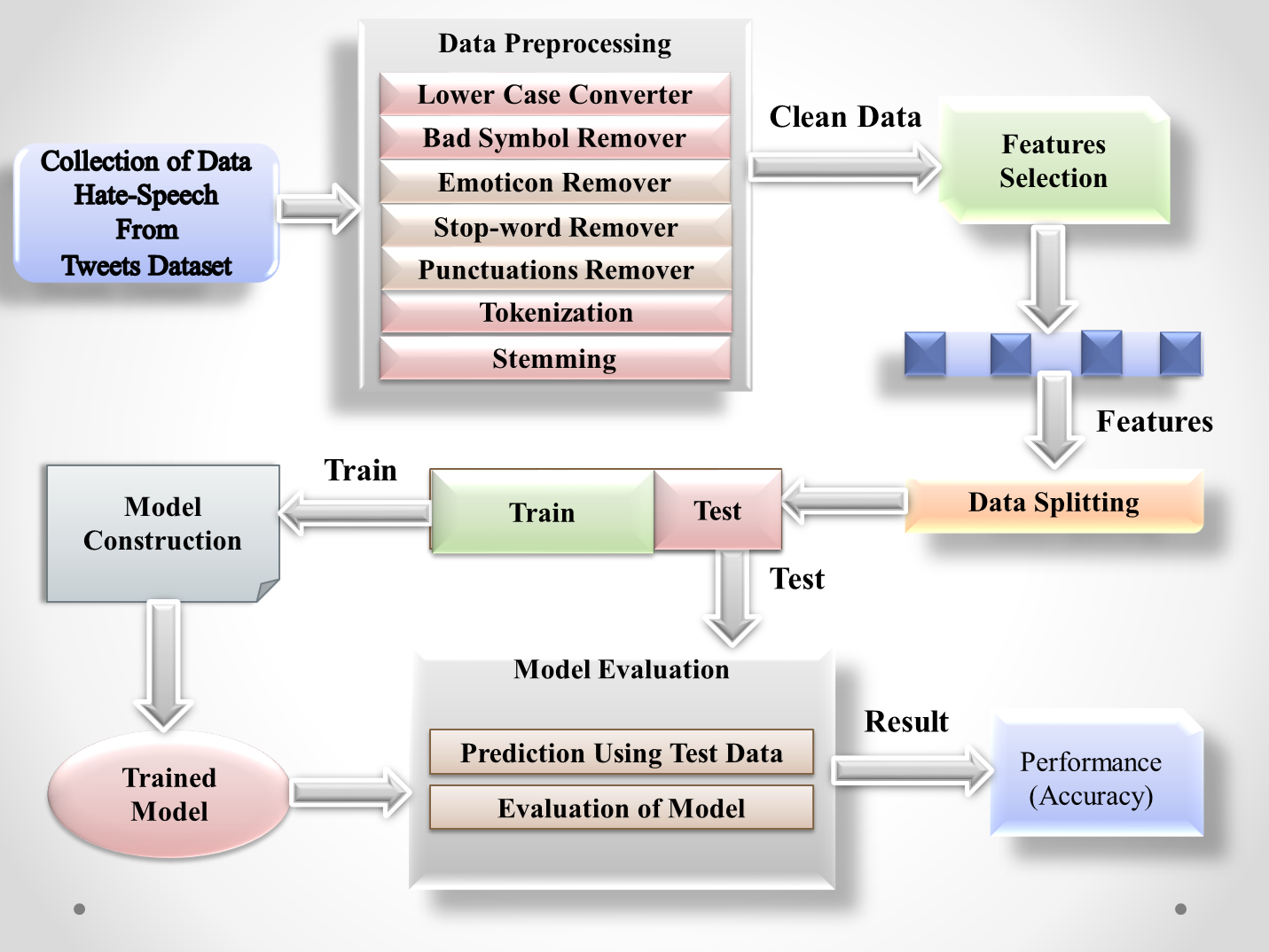
Afterward, “Elizaveta Zinovyeva, Wolfgang Karl Hardle and Stefam Lessmann” are working to elaborate on the detection of antisocial behavior in their paper named “Antisocial Online Behavior Detection Using Deep Learning” [17]. They used Support Vector Machine (SVM), Logistic Regression (LR), decision tree, and lexicon-based approaches and take a dataset from an online newspapers. As a result, they find a good accuracy which is 75%, 48%, 62%, and 45%. Here Support Vector Machine (SVM) gives high accuracy than other methods.

Around 2020 year, the Pakistani researchers named “Sindhu Abro, Sarang Shaik, Zafar Ali, Sajid Khan, and Ghulam Mujtaba” are worked on detecting Hate Speech from text using different types of methods and classifiers and proposed a paper named, “Automatic Hate Speech Detection Using Machine Learning: A Comparative Study” [18]. Here they used Support Vector Machine (SVM), Logistic Regression (LR), Random Forest (RF) method, and classifiers. Using those methods and classifiers they find 79%, 75%, and 75% accuracy as output from different datasets which are taken from different social media. However, Support Vector Machine (SVM) gives they’re the best result from other approaches.

**Chapter 3**

# Methods

The methodology is an essential section that explains the proposed system of “Conceptual Clarity and Terminology of Hate-Speech Intensity Scale” where finds the detection and classification of Hate-Speech from text or comments which are taken from Tweeter and voice speech. After finding the suitable and desired dataset for “Conceptual Clarity and Terminology of Hate-Speech Intensity Scale”, some advanced models are selected for detecting the Hate-Speech from text or comments and divides those comments or text into four Hate emotion categories (e.g. stupidity, pathetic, greedy, and retarded). To detect the Hate-Speech from text or comments, so many steps have to across. The whole process is shortly shown in figure: 1. each of the steps is discussed with details in the approximate sections.



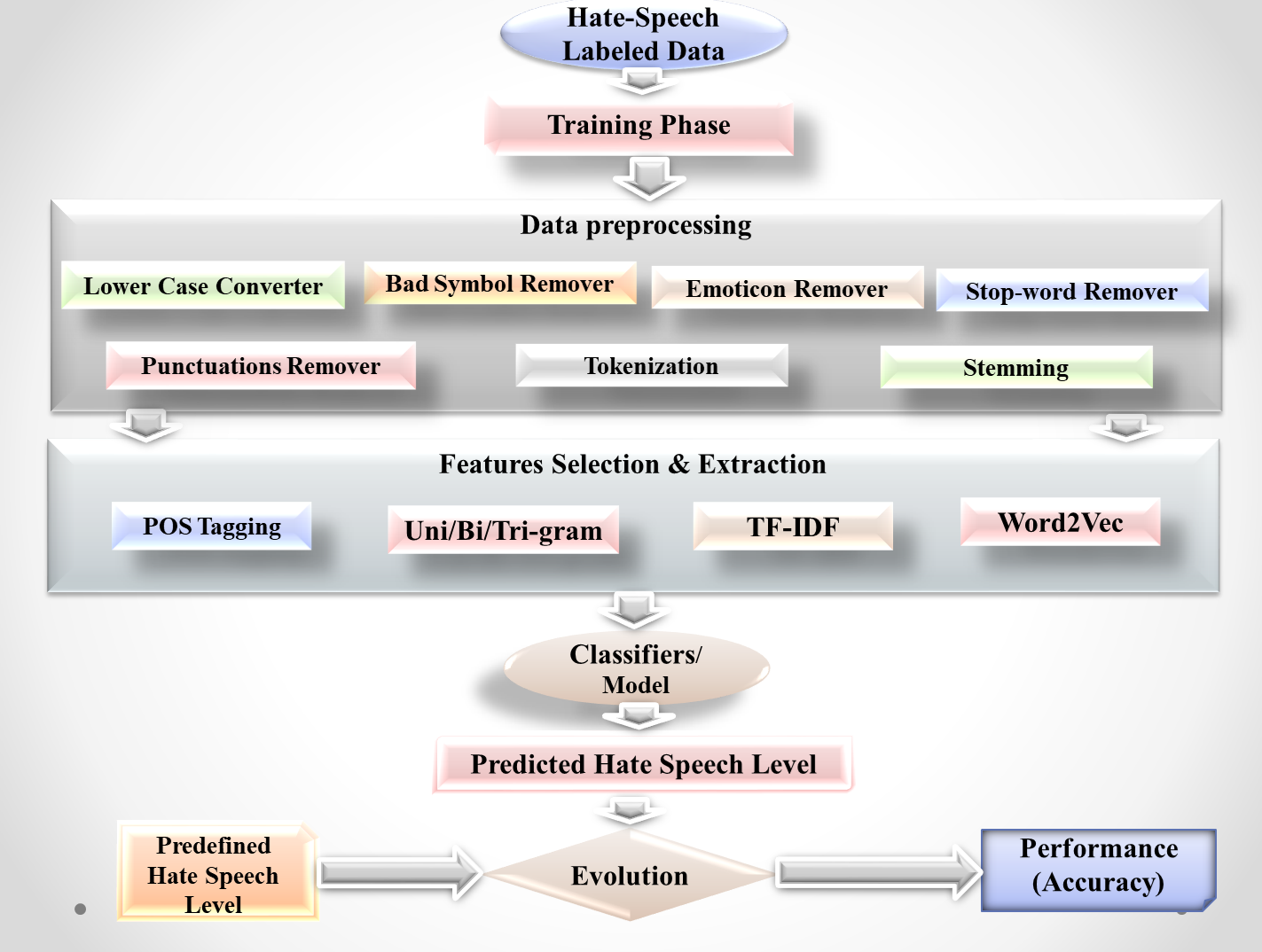
**Fig. 3** System Overview

Moreover, the prime key objective of this thesis was to use a supervised machine learning method called the Naive Bayes classifier to detected only hate comments from the text. But this paper represents the detection of hate speech with the proper four emotion labels like stupidity, pathetic, greedy, and retarded. So, before classifying all the data, it went through several pre-processing procedures to remove any unnecessary information and symbols. That is why increasing classification performance in so many different feature selection techniques is studied. Each of these strategies and techniques was vital because they could impact the overall outcome of this supervised method or model. As a result, in this section, a little architecture has been given which is an overview of approaches for understanding all the processes clearly.

Therefore, the system can be separated into two phases: training and testing to understand the whole process spontaneously and clearly. For this reason, at first, the entire dataset was first sorted into training and test data. Then before feeding the training set to the classifier, it is gone through so many pre-processing phases, and then several feature selections approaches are performed. After that, the test data run through the same process and predicting possible hate speech with emotion labels for each document. So, the system's efficiency is then evaluated and can compare with the predefined emotion labels to the predicted ones.

1. **Training Phase:**

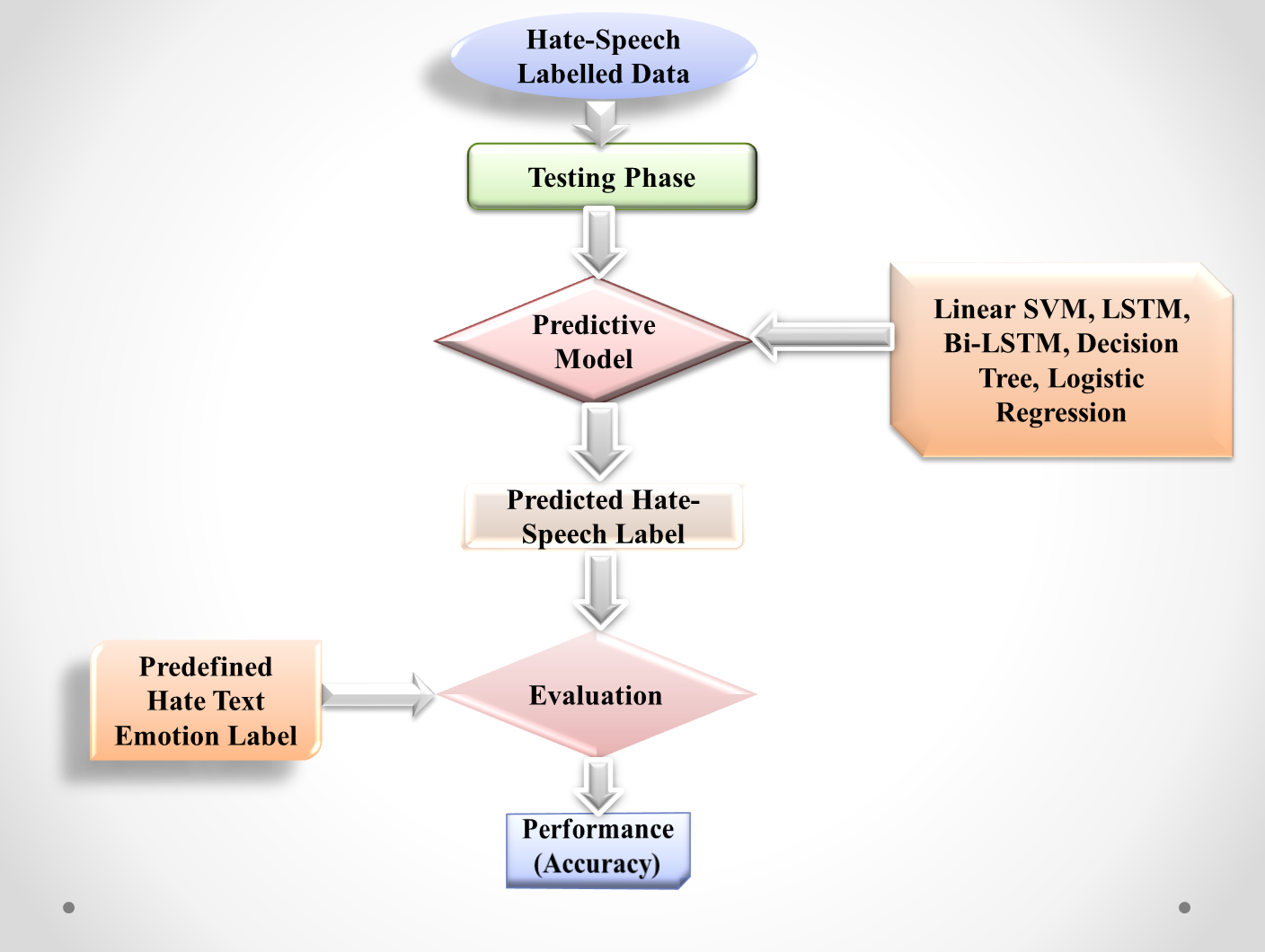
At first, the Training phase is the most significant and essential phase in machine learning or deep learning study or implementation where it is used to train all the algorithms and classifiers for creating the right and accurate output. In extensively, the training phase is a model that simply means learning or determining excellent values for all the gravity and the bias from labeled data, models, or examples. However, the main goal of the training phase is to find an excellent set of the weights and the biases that have low loss, on average, across all models and examples. Essentially, the training phase is a learning phase that has the input parameters and configured the models which have the input data. To train the model, it is used the training set and worked 80% of the whole process and other are worked testing phase. In brief, the training phase means creating the model.



**Fig. 3-A** Architecture of Training Phase

1. **Testing Phase:**

Among machine learning or deep learning study or implement testing phase lead another vital role which is a method of measuring the gain accuracy of the used model. In generally, the testing phase or model testing is referred that it is a process where the performance of a fully trained model is evaluated on a testing set and gives an accuracy of used models. To assess testing results, there are numerous numbers of statistical metrics that are used and also including mean manage or squared errors and receiver operating or functional characteristics curves. Testing phase worked after evaluating training phase The model tested using testing set and worked 20% the whole process other are worked training phase.

****

**Fig. 3-B** Architecture of Testing Phase

However, the proposed method can be divided into four sequential phases:

1. Dataset Preparation

2. Pre-processing

3. Feature Selection and Extraction

4. Classifier / Model

* 1. **Dataset Preparation:**
     1. **Data Collection:**

For the progress of “Conceptual Clarity and Terminology of Hate-Speech Intensity Scale” work, need a suitable dataset. Lacks the labeled and standard dataset, previous researchers are not reached to gain near the good accuracy to detecting Hate speech or comments from the text. Remembering this thing, there the research begins with detecting a Hate-speech from text which is taken from Twitter comments. So, for the purpose of classification and detection of the Hate-speech, around 26554 line comments or tweets are collected from Twitter which is converted from voice and used on this paper as a suitable dataset. The dataset is more suitable than other datasets because it contains around 26554 tweets which indicate a huge number of comments that are not used in another research before this research. This dataset is taken from online Twitter which is users' feedback or comments on a variety of different topics socio and political issues are among those topics. The dataset can be also used to separate or classify four hate emotions such as stupidity, pathetic, greedy, and retarded.

* + 1. **Data Filtering:**

A dataset is played a significant preface for the progression of any kinds of research and to do implementation and get strong or prosperous outcome or result. Around 26554 line comments as a dataset used in this study which are taken from Twitter. Therefore, it become extremely challenging to handle this huge dataset and also separate or classify four hate emotions such as stupidity, pathetic, greedy, and retarded terms and phrases. The entire dataset is split into two part which named training and testing phase. From 26554 line comments, 20485 comments or lines are used for training phase and 6455 comments or lines are used for testing phase.

|  |  |  |
| --- | --- | --- |
| **Label** | **Train Dataset** | **Test Dataset** |
| Stupidity | 10033 | 3202 |
| Pathetic | 3825 | 1210 |
| Greedy | 5412 | 1238 |
| Retarded | 1215 | 805 |
| Total | 20485 | 6455 |

**Table 3-A:** Class distribution in the Dataset

* 1. **Pre-processing:**

Pre-processing or Data pre-processing is a process that is prepared the raw data and make it suitable for Machine Learning models. Another way in Machine Learning, Pre-processing or Data pre-processing mainly indicate the technique which is helped to prepare (cleaning, manufacturing, and organizing) the raw data to make suitable, and competent for a building or creating, and training the Machine Learning models and also transfer or posterior processing procedure. Traditionally, it was used for the preliminary step or leap for a Data Mining process. Moreover, it is the first and crucial step for creating a Machine Learning model. It is mandatory that while it is doing any operation with given data; it has to clean data and also put it in a properly formatted way for the model. It also can do remove punctuations marks, numbers, stop words, duplicate data, and unnecessary symbols which are worthless from the comments or datasets. For this reason, the next step will be more and easier to process.

* + 1. **Lower Case Converter:**

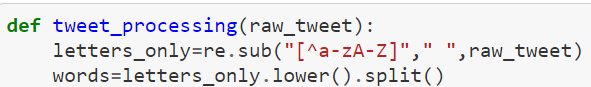
In the Pre-processing or Data pre-processing or Text pre-processing, upper to lower case converting is a most necessary step because in Machine Learning upper case create problem to identify actual word or text and more comfortable to identify word or text in lower case. For example, during the compelling or investigate type if any word like “Us” has a place in a text, it will be confused by two things. First one, it thought that this word might be a pronoun representing “We” on the text or sentence. The second one, it thought that this word indicates the country “USA”. For this confusion, the machine can not avail to give the right output. That is why lower case letters or text must have to be used in the dataset. To convert lower case an example code likes,

input\_str = “cant you see these hoes wont change”

input\_str = input\_str .lower()

print(input\_str )

From the implementation part of this study,



**Fig. 3-1** Lower Case Converter

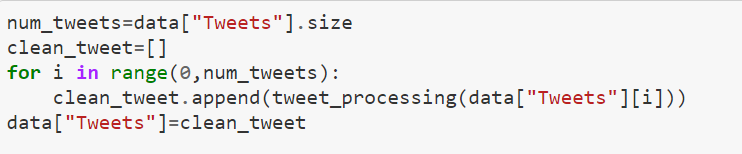
* + 1. **Bad Symbol Remover:**

Every line, comment, or text has some bad or special symbol and numbers. Here bad or special symbols are ‘$’, ‘!’, ‘+’, ‘-‘, ‘\*’, ‘\_’, ‘`’, ‘~’, ‘@’, ‘#’, ‘^’, ‘&’, ‘?’, ‘/’, ‘.’, ‘<’, ‘>’, ‘%’, ‘alpha, ‘beta’, ‘gama’, and so on. The numbers are ‘0’, ‘1’ to infinity. To earn a good output and accurate accuracy, this symbol must have to remove from lines, texts, or comments. Otherwise, it must create a big problem for detecting the text and the machine became confused seen those symbols. To discard those symbols the code looks like,

extra\_pun = [‘$’, ‘!’, ‘+’, ‘-‘, ‘\*’, ‘\_’, ‘`’, ‘~’, ‘@’, ‘#’, ‘^’, ‘&’, ‘?’, ‘/’, ‘.’, ‘<’, ‘>’, ‘%’, ‘0’, ‘1’], or

clean\_num = [‘$’, ‘!’, ‘+’, ‘-‘, ‘\*’, ‘\_’, ‘`’, ‘~’, ‘@’, ‘#’, ‘^’, ‘&’, ‘?’, ‘/’, ‘.’, ‘<’, ‘>’, ‘%’, ‘0’, ‘1’]

From the implementation part of this study,



**Fig. 3-2** Bad Symbol Remover

* + 1. **Emoticon Remover:**

People are often used emoji’s and emoticons in a sentence, comment, or text for expressing their feeling or narrate purpose instead of writing on a social media platform. So, before removing this symbol, the first one has to understand the difference between emoji and emoticon. Here, both emoji and emoticon sustain the emotional expression of a text message or comment. However, the main difference between emoji and emoticon is, an emoji is a small actual image that is used for expressing emotion or any idea of a text message, or comment. On the other hand, an emoticon is a facial expression or representation which is used from keyboard characters and punctuations. To understand, there have an example,

For representing happy face,

Emoji: C:\Users\Hp\Downloads\download.jpg

Emoticon: **‘:)’**

To remove emoji’s and emoticons, the sample code likes,

import re

def remove\_emoji (string) :

emoji\_pattern = re.compile (“[“

u”\U0001F600-\ U0001F64F” # emoticons

u”\U0001F300-\ U0001F5FF” # symbol & pictographs

u”\U0001F680-\ U0001F6FF” # transport & map symbols

u”\U0001F1E0-\ U0001F1FF” # flags (ios)

u”\U00002702-\ U000027B0”

u”\U000024C2-\ U0001F251”

”]+”, flags=re.UNICODE)

return emoji\_pattern.sub (r’’, string)

remove\_emoji (“cancel that bitch like Nino C:\Users\Hp\Downloads\1_CNwn1bMLvg08iHyvPpxAXQ.jpeg .” ”)

**Output:** cancel that bitch like Nino.

From the implementation part of this study,



**Fig. 3-3** Emoticon Remover

* + 1. **Stop-word Remover:**

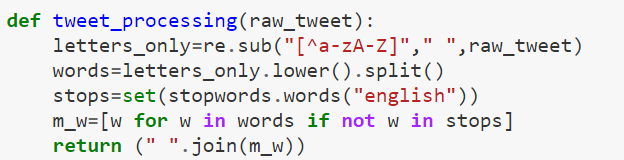
One of the most commonly used pre-processing; text pre-processing or data pre-processing steps is Stop-word removal in the different Natural Language Processing (NLP) applications and Text Mining to eliminate words that are mostly, commonly used in a text that carries very little useful information. Another way Stop-word is the set of words that might be from any language which does not add much meaning to a sentence, comment, or text. For this reason, those words can be safely ignored and also abreast use the sentence without sacrificing its real meaning. So, in short, Stop-word is a simple process or idea that simply removes the words which are usually conducted in all the documents, comments, or texts in the corpus. Such as the, is, at, which, on, a, as, so, in, or, are, the, and so on are the examples of Stop-word. For Stop-word removal code might be like,

from nltk.corpus import stopwords

stop\_words = stopwords.words(‘engilish’)

print(stop\_words)

From the implementation part of this study,



**Fig. 3-4** Stop-word Remover

* + 1. **Punctuations Remover:**

Punctuations are mostly and freely used in all sentences, texts, comments, and words. Without using punctuation, no sentences, texts, comments, and words are finished. Sometimes without any reason, people or users used different category punctuations in their texts or comments. Mainly it helps to get rid of ineffective or unhelpful parts of the data, text, noise, or comments by removing punctuations marks. The most common punctuations marks are period (.), question mark (?), exclamation point (!), comma (,), semicolon (;), colon (:), dash (\_), hyphen (-), parentheses (()), brackets ([]), braces ({}), apostrophe (‘), quotation marks (““), and ellipsis (…). Finding an excellent output must have to remove punctuations marks from every line of a used dataset. Otherwise, it creates huge problems while the code is compiled. To remove Punctuations marks, the possible and common used sample code is,

import string

regular\_punct = list (string.punctuation)

def remove\_ punctuation (text, punct\_list) :

for punc in punct\_list:

if punc in text :

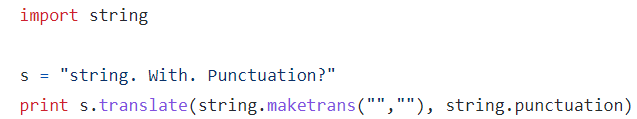
text = text.replace (punc,‘ ‘)

return text.strip ()

remove\_ punctuation (“So hoes that smoke are losers ? "" yea ... go on IG”)

**Output:** So hoes that smoke are losers yea go on IG

From the implementation part of this study,



**Fig. 3-5** Punctuations Remover

* + 1. **Tokenization:**

Tokenization is introduced to the process that is turned a meaningful piece of data, such as an account name, number, phrase, and sentence into an indiscriminate or random string of each character which must have meaningful value. Shortly, Tokenization is a process which is breaking up or split a phrase, sentence, paragraph, keyword, symbol, or an exhaustive text document, comment into smaller units. Here, smaller units are indicated as individual words or terms. Each of those smaller units or terms is called a token. In addition, Tokens can be individual words, phrases, or even a whole sentence of any paragraph. However, in the Tokenization process, some characters like punctuation marks are also discarded. Mainly it needs for implementation to make easier the work of the compiler and also get excellent output. The possible sample code is,

Word tokenizer:

from nltk.tokenize import word\_ tokenize

text = “So hoes that smoke are losers ? "" yea ... go on IG”

print (word\_ tokenize (text) )

**Output:** [‘So’, ‘hoes’, ‘that’, ‘smoke’, ‘are’, ‘losers’, ‘?’, ‘yea’, ‘...’, ‘go’, ‘on’, IG’]

Sentence tokenizer:

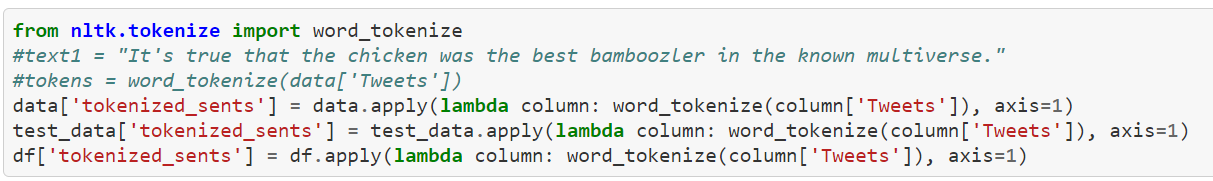
from nltk.tokenize import sent\_ tokenize

text = “So hoes that smoke are losers ? "" yea ... go on IG”

print (word\_ tokenize (text) )

**Output:** [‘So hoes that smoke are losers?’, ‘yea ... go on IG’]

From the implementation part of this study,



**Fig. 3-6** Tokenization

* + 1. **Stemming:**

Stemming is a well-known process of Machine Learning, Deep Learning, Natural Language Processing (NLP), and Natural Language Understanding (NLU). In short, Stemming normally refers that a method of chopping off the last few words, or character. Stemming is a process that can reduce a word to its word stem which converts words, affixes to suffixes and prefixes or the roots of words. So, Stemming is a procedure that can remove the suffix from a word and also reduce it into its root word. Whenever it finds any new word, it can be present new research opportunities. Stemming can operate on a single word or character without any knowledge of the main context. Mainly, the uses of suffixes are to create or find a new word from an original or real stem word. But Stemming is not a well-defined method or process, and it often suffers from incorrect meaning problems and also spelling errors. The stems were then removed from the left or right section of the word.

To better understand, there has an example:

“Crying” is a simple word. Here the suffix from of Crying is “ing”. So, if the “ing” word removes from “Crying”, then the base or root word comes which is “Cry”.

For implementations, the sample code will be like,

from nltk.stem import PorterStemmer

from nltk.tokenize import word\_tokenize

stemmer= PorterStemmer()

input\_str= “bad bitches is the only thing that i like”

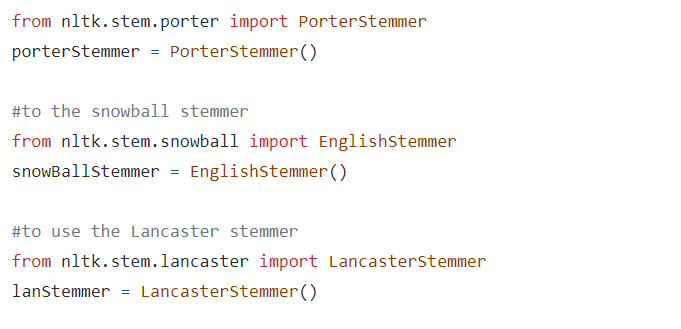
input\_str=word\_ tokenize(input\_str)

for word in input\_str:

print (stemmer.stem (word) )

**Output:** bad bitch is the only thing that I like

From the implementation part of this study,



**Fig. 3-7** Stemming

* 1. **Feature Selection and Extraction:**

After the Pre-processing, Text pre-processing, or Data pre-processing Feature Selection and Extraction step come. Feature Selection and Extraction is one of the most significant parts to analyze the “Conceptual Clarity and Terminology of Hate-speech Intensity Scale” because it will affect the whole result of this study or research that is why must have to complete this section carefully. To earn a better and excellent result or outcome must have to select advance and proper features for this kind of research. For this reason, selecting a luscious feature is more and more essential for acquiring better consequences. After taking counsel from the antecedent research used feature selections, here used various perspectives for evaluating this study with its processed data after passing on the pre-processing period which is just completed and also here used a combinational strategy for achieving the best possible result. The selecting dataset is also used to create this work expanded Hate emotional dictionary, which is employed a strong sequel. Feature Selection and Extraction of this study have consisted of four different phases.

* + 1. **POS Tagging:**

Part-of-speech (POS) Tagging consider as one of the primaries, or basic but most necessary phase which is required for Natural Language Processing (NLP), and Deep Learning appeals where the word sense or knowledge is disambiguating, information is processed, parsing, answering from the question, and also can machine translation. So, Part-of-speech (POS) Tagging, or tag is a particular label which is assigned for each token, or word in a word, or text corpus and also indicates the part of speech, and other grammatical approaches, or categories such as tense, number (singular/plural), case, adjective, and so on. Moreover, the Part-of-speech (POS) Tagging is aimed to put in parts of speech into each of the given words, or text-based on their definition and context. Part-of-speech (POS) Tagging are most useful to build parse trees which are used to build NERs (most commonly used name of Nouns) and extract the relation between words or texts. To build, Lemmatize (used to reduce a word or text from its root word, or text), POS Tag is also played an essential role. Mostly POS Tag is used in corpus searching and algorithms as a text analysis tool. For better understanding and example of POS Tagging is,

**Input:** hoe what its hitting for.

**Output:** [(‘hoe’, u’NN), (‘what’, u’WDT), (‘its’, u’NNS), (‘hitting’, u’VBG), (‘for’, u’IN)]

The probable sample code is,

input\_str=” cancel that bitch like Nino”

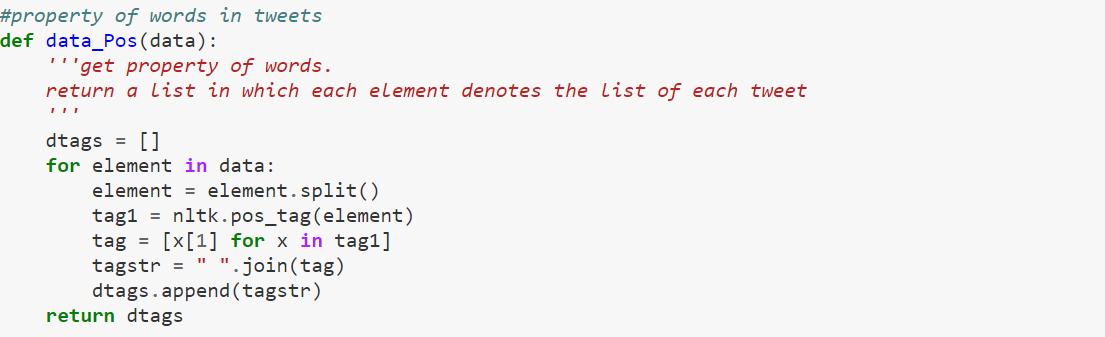
from textbob import TextBlob

result = TextBlob(input\_str)

print(result.tags)

**Output:** [(‘cancel’, u’VB), (‘that’, u’WDT), (‘bitch’, u’NN), (‘like’, u’JJR), (‘Nino’, u’NN)]

From the implementation part of this study,



**Fig. 3-8** POS Tagging

* + 1. **Uni/ Bi/ Tri- gram:**

N-gram plays an important role in text analysis which is probably the easiest concept in Machine Learning, and Natural Language Processing (NLP). An N-gram is a contiguous sequence or series that has n items sequence from the given sequence of text, speech, or comment in the fields of Machine Learning, Computational Linguistics, and Probability. Here, the items are letters, words, syllables, comments, phonemes, or base pairs according to appeal. Typically, the N-gram is collected from a text, comment, or speech corpus. N-gram is language independence that can be avail to apply in a new language without any additional effort. There, N-gram can be reflecting the information which is consisting of the content and context of the typed character, and word. For this reason, N-gram of text, comment, and speech is widely used in Text Mining tasks. In addition, an N-gram also means a sequence of N words. Sometimes a single word, comment, or text is not sufficient for observing the whole context of any text. So, N-gram can be dividing into three parts. After observing, the performance of uni-gram, bi-gram, and tri-gram to achieve the best model for detecting exact hate emotion from the English text. There Uni-gram is defined to indicate a one-word dilution where Bi-gram is consisting of two-word dilution, and the last one Tri-gram is consisting of a three-word dilution of a word. Howsoever, in this study, a noticeable thing is a combinational feature of the N-gram features which is easy to understand. Normally, Tri-gram provides a relatively better result than Bi-gram, and Uni-gram and also Bi-gram gives better result than Uni-gram. There is an example for better understand those grams.

|  |  |
| --- | --- |
| **Sample Text** | it aint nothing to cut a bitch off |
| Uni-gram Feature | ‘it’, ‘aint’, ‘nothing’, ‘to’, ‘cut’, ‘a’, ‘bitch’, ‘off’ |
| Bi-gram Feature | ‘it aint’, ‘aint nothing’, ‘nothing to’, ‘to cut’, ‘cut a’, ‘a bitch’, ‘bitch off’ |
| Tri-gram Feature | ‘it aint nothing’, ‘aint nothing to’, ‘nothing to cut’, ‘to cut a’, ‘cut a bitch’, ‘a bitch off’ |

**Table 3-B:** Uni/ Bi/ Tri- gram

The possible sample codes of N gram like,

from nltk.util import ngrams

def extract\_ngrams (data, num) :

n\_grams = ngrams (nltk.word\_tokenize (data), num)

return [‘ ’.join (grams) for grams in n\_grams]

data = ‘ Murda Gang bitch its Gang Land ’

print (“1-gram: “,extract\_ngrams (data, 1))

print (“2-gram: “,extract\_ngrams (data, 2))

print (“3-gram: “,extract\_ngrams (data, 3))

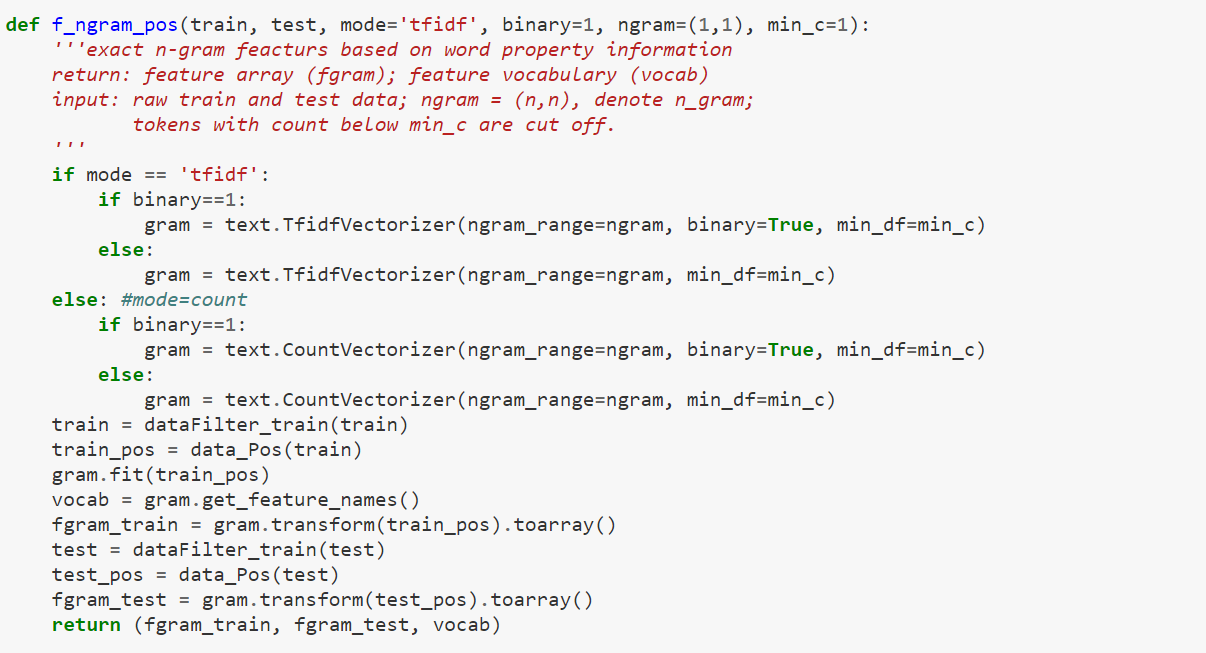
**Output:**

1-gram: [‘Murda’, ‘Gang’, ‘bitch’, ‘its’, ‘Gang’, ‘Land’]

2-gram: [‘Murda Gang’, ‘Gang bitch’, ‘bitch its’, ‘its Gang’, ‘Gang Land’]

3-gram: [‘Murda Gang bitch’, ‘Gang bitch its’, ‘bitch its Gang’,’ its Gang Land’]

From the implementation part of this study,



**Fig. 3-9** N-gram

* + 1. **TF-IDF:**

Firstly, TF is the short form of the Term Frequency, and IDF is the short form of the Inverse Document Frequency. TF-IDF is indicating a matric that is taken into estimate word, or term frequency which can be both in a single document and in all corpuses (Collection of the document). Moreover, TF-IDF arrives when a word or term is became high frequency in a single record, or document and low frequency on the whole corpus. Shortly, TF-IDF is a method that is delegated the reflection of any particular word, or term in a corpus document of a numerical weightage of words. So, for Natural Language Processing (NLP) task, TF-IDF is the most popular used accession to weight terms. It can assign a value into a term, or word in according to its necessary in a document scaly which its importance crosswise the whole documents of is given corpus that is always mathematically, or exactly removes occurring words in the English language sentence, comment, or word, and can be the select word that is more and more. Here, TF is the frequency for the term t in document d. So, this is a way that is used for the metric which is applied by logarithm normalization to give the counts. After that, IDF is tried to give more and more weight for less frequent terms once that are more preferential. Because of that, it is achieved by looking at the logarithm which base will be 10 and also the inverse of the proportion of the given documents in the whole corpus and which can contain the actual term. To identify or apply TF-IDF, there has a formula that is used for information recovery and also in text mining and successfully used for the stop-words filtering from the text document. The formula is,

TF (w) = (Number of times word w, arrived in the letter) / (Total number of words in the whole letter) (1)

DF (w) = log (e) (Total numbers / number of letters with word w) (2)

TF-IDF (w) = TF (w) \* IDF (w) (3)

Now, for better understand there is an example:

Assume that a document is built with 200 words. Whereas, the word “sad” arrived 3 times. After that assume that there have 10 million documents and the word “sad” arrived one thousand times in there. So, there

For “sad”,

The Term Frequency (TF) is (3 / 200) = 0.02.

The Inverse Document Frequency (IDF) is (10,000,000 / 1,000) = 4.

Therefore, TF-IDF = 0.02 \* 4 = 0.08.

The sample code of TF-IDF is,

from sklearn.feature\_extraction.text import TfidfVectorizer

text = [‘cant you see these hoes wont change’,

‘hoe what its hitting for’]

vectorizer = TfidfVectorizer()

x = vectorizer.fit\_transform(text)

print(vectorizer.get\_feature\_names())

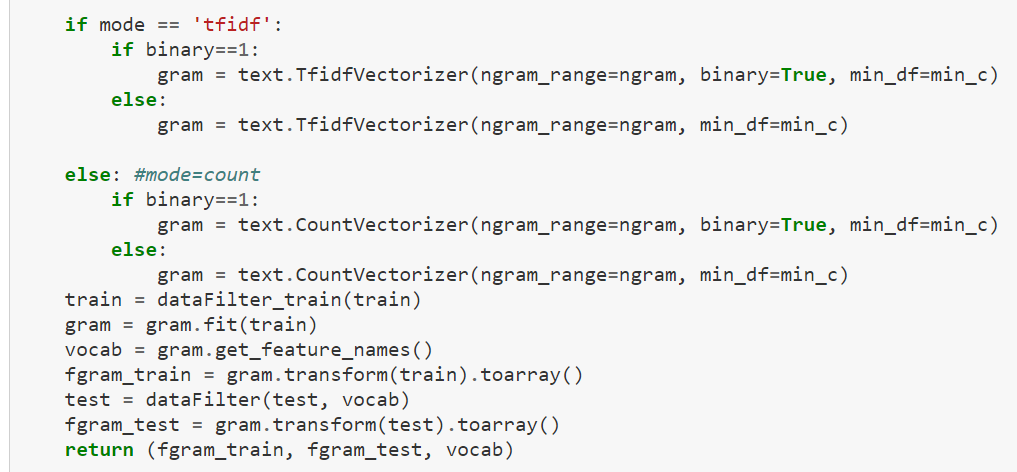
print(x.shape)

**Output:**

[‘hitting’, ‘hoe’, ‘wont’, ‘change’]

(2, 4)

From the implementation part of this study,

****

**Fig. 3-10** TF-IDF

* + 1. **Word2Vec:**

Nowadays, Natural Language Processing (NLP), Deep Learning, and Text Mining Word2Vec is another popular and essential technique which is published in 2013, and it is mainly a language modeling technique that is used for mapping words into vectors of the real numbers. In short, Word2Vec has consisted of models that are used for generating word embedding and these models have two-layer Neural Networks that have one input layer, one hidden layer, and only one output layer which are processed text, sentence, or comment by “Vectorizing” the words. Here, the input layer is a text corpus, and the output layer is a set of vectors. So, in another word, the Word2Vec is a technique or algorithm which is used in a Neural Network model for learning word confederations from a broad corpus of text, sentence, or comment, and also used for extracting the concept of the relatedness across words, or manufactured like semantic relatedness, detect synonym, or suggest additional words, categorization concept, and analogy. But while Word2Vec is not a Deep Neural Network, it branches text, comment, or sentence into a numerical form which is Deep Neural Networks can understand properly. The sample code of Word2Vec such as,

from genism.models import Word2Vec

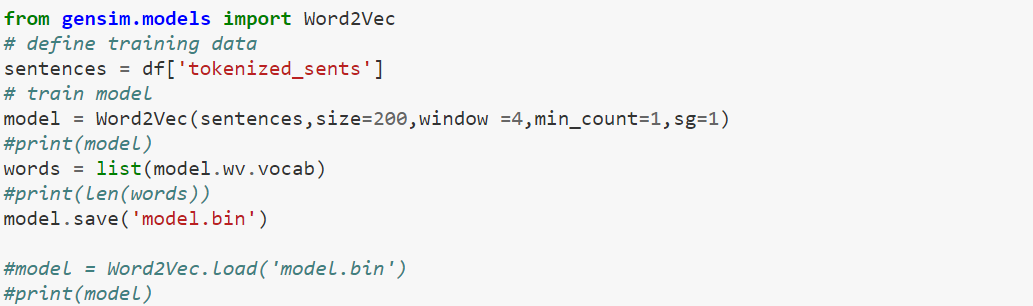
word\_vectors = model.wv

word\_vectors.save (“Word2Vec.wordvectors”)

wv = Word2Vec.load (“Word2Vec.wordvectors”, mmap = ‘r’)

vector = wv [‘c’]

From the implementation part of this study,



**Fig. 3-11** Word2Vec

* 1. **Classifier / Model:**
     1. **Linear SVM:**

The Support Vector Machine (SVM) is known as a supervised Machine Learning algorithm, which is mostly used for the challenge of both classification, and regression and can simply be the coordinates of the individual regard. Support Vector Machine (SVM) classifier is also known as a frontier which is the best separates method of the two classes; two classes are Hyper-plane and Line. However, for classification, the Support Vector Machine (SVM) classifier is also known as a frontier which is the best separates method of the two classes; two classes are Hyper-plane and Line. However, for classification, Support Vector Machine (SVM) is the linear models which can regression problems. So, it can solve both linear and non-linear problems and also give excellent work for many of the practical problems. So, in short, Linear Support Vector Machine (SVM) is mainly used for separating data linearly, which indicates the meaning that if a dataset can be classified into two different classes by using a single straight line and after that, the user data is termed as like linearly separate data, and classifier then it called Linear Support Vector Machine (SVM) classifier that creates an SVM model in a CPU time which can scales data linearly with the size of the used training data set. So, in short, Linear Support Vector Machine (SVM) is mainly used for separating data linearly, which indicates the meaning that if a dataset can be classified into two different classes by using a single straight line and after that, the user data is termed as like linearly separate data, and classifier then it called Linear Support Vector Machine (SVM) classifier that creates an SVM model in a CPU time which can scales data linearly with the size of the used training data set. For this reason, the idea of the Support Vector Machine (SVM) is too simple that the algorithms can create a line or a hyperplane that can separate the user data into classes. Hate Emotion detection from English text data is a new and most challenging topic in the Artificial Intelligence field. For implementing SVM some standard libraries have to import. Those libraries are;

import numpy as np

import matplotlib.pyplot as plt

from scipy import stats

import seaborn as sns; sns.set()

Code can be written in many ways a sample way can be such as

from sklearn.dataset.samples\_generator import make\_blobs

P, q = m\_blobs (n\_sam=200, cen=2, r\_st=0, c\_std=0.50)

plt.scatter(P[:, 0], P[:, 1], c=y, s=50, cmap=’sr’);

pfit = np.linspace(-1, 2.5)

plt. scatter(P[:, 0], P[:, 1], c=y, s=50, cmap=’sr’);

plt.plot ([0.5], [2.1], ‘p’, color=’black’, mwidth=5, size=14)

for m, d in [(1, 0.65), (0.5, 1.6), (-0.2, 2.9)] :

plt.plot(pfit, m \* pfit + d, ‘-k’)

plt.plim(-1, 3.5);

From the implementation part of this study,



**Fig. 3-12** Linear Support Vector Machine (SVM)

* + 1. **LSTM:**

Long-Short Term Memory (LSTM) is known as an artificial Recurrent Neural Network (RNN) architecture, which is mostly used in the Deep Learning complex field. So, in recurrent structures for sequence modeling, Long-Short Term Memory (LSTM) is one of the widely used structured. For this reason, the LSTM network, or structure is very well suited for classifying, processing, and making predictions that are based on the time series data which can be crawls of the unfamiliar duration between essential occurrences in the time series. So, in short, Long-Short Term Memory (LSTM) network, or structure is a type of Recurrent Neural Network (RNN) that is capable of learning, or wisdom order that is dependent on many sequences of prediction problems, or matters and give a better result, or solution than traditional Recurrent Neural Network (RNN) in the terms of memory. LSTM performs fairly better because it has a stronghold over memorizing certain patterns. Moreover, LSTM was developed for dealing with the vanishing gradient issue which is encountered while the training traditional RNN comes. But LSTM is the most complex area of the Deep Learning study. So, LSTM has indicated a behavior that is required in complex matters such as Machine Translation, Speech Recognition, and so on. However, LSTM is mostly used for getting control given information like data set to flow in the recurrent computations. Farther, LSTM network or structure is quietly very strong or good to hold long term memories which are may be or maybe not retained by the Network depending on the used or given data. For further procedure of implementation, some standard libraries have to import for LSTM. Those libraries are,

import numpy as np

from keras.preprocessing import sequence

from keras.models import Sequential

from keras.layers import Dense, Embedding, LSTM, Bidirectional

from keras.datasets import imdb

Code can be written in many ways a sample way can be such as

n\_uni\_words = 10000

maxlen = 200

bat\_size = 128

(x\_train, y\_train), (x\_test, y\_test) = imbd.load\_da(num\_wor=uni\_wor)

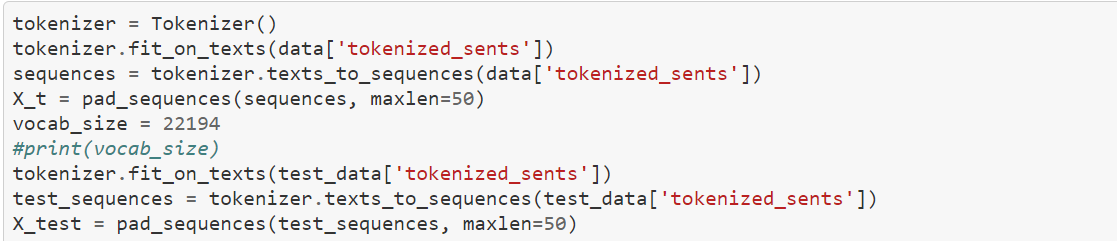
x\_train = seq.pad\_seqs(x\_train, maxlen)

x\_test = seq.pad\_seqs(x\_test, maxlen)

y\_train = np.arr(y\_train)

y\_test = np.arr(y\_test)

Some parts from the implementation part of this study,



**Fig. 3-13** Long-Short Term Memory (LSTM)

* + 1. **Bi-LSTM:**

First of all, Bidirectional Long-Short Term Memory (Bi-LSTM) is indicated a sequence which is a processing model that is consisted of two different LSTM, such as one is taking the input in the forward direction, and second, on is the backward direction. So, Bidirectional Long-Short Term Memory (Bi-LSTM) is expressed as a process that can make any Neural Network which covered the whole sequential information that is a builder by both directions, they are forward and backward direction. There, forward direction means past to future and backward direction means future to past information. In shortly, Bidirectional Long-Short Term Memory (Bi-LSTM) has alluded to an extension of the traditional LSTM which can improve the performance of the model with sequential classification issues. Those issues can be all-time steps of the given input sequence that are available, and Bidirectional Long-Short Term Memory (Bi-LSTM) can train two instead of the one LSTM in the input sequence. Usually, Bi-LSTM is employed where the used sequence has a need sequence to sequence task. But the main difference between LSTM and Bi-LSTM is, LSTM gate Recurrent Neural Network (RNN), and Bi-LSTM is just only an extension to that model. Moreover, Bi-LSTM can flow input into two directions (forward and backward) for making it different from regular LSTM. Regular LSTM can make flow in only one direction which can be either forward or backward. Mainly the directions of Bi-LSTM are used to preserve the future and the past information. For implementation, used libraries are as same as LSTM which are,

import numpy as np

from keras.preprocessing import sequence

from keras.models import Sequential

from keras.layers import Dense, Embedding, LSTM, Bidirectional

from keras.datasets import imdb

Code can be written in many ways a sample way can be such as

 model = seq()

model.add(Emb(n\_uni\_words, 128, input\_len=maxlen))

model.add(Bidirectional (LSTM (64) ) )

model.add(Drop (0.5) )

model.add(Den (1, act=’sig’) )

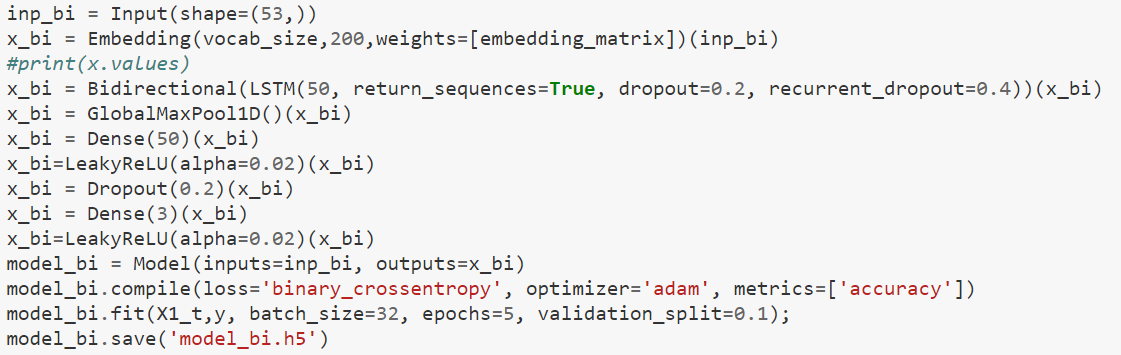
model.compile(loss=’bin\_cross’, optimize=’adam’, metrics=[‘accuracy’] )

history=model.fit (a\_train, y\_train, bat\_size=bat\_size, epo=12, valid\_data=[x\_test, y\_test] )

print (history.history [‘los’] )

print (history.history [‘accuracy’] )

Some parts from the implementation part of this study,



**Fig. 3-14** Bidirectional Long-Short Term Memory (Bi-LSTM)

* + 1. **Decision Tree:**

The Decision Tree (DT) is one kind of Machine Learning algorithm which can be a non-parametric Supervised Learning method or technique that is mostly used for classification and regression where the data is always tried to continuously split that are accordingly like to a certain parameter on the training data phase. So, Decision Tree (DT) classification has two-step processes which are The Learning step and the Prediction step. The learning step is built with the model which is a development based on the given training data set and the prediction step is built with the model which is used to predict the coming response from the given data set. However, the Decision tree (DT) can act like a flowchart structure that can represent a test on the used features of each of the internal nodes, each of the leaf nodes is represented the class label, and the branch is represented conjunction from of the used features which can lead all of those class. To better understand there has an example of the internal node that is whenever a coin flip then there might become up either heads or tails. After that, the leaf node can decide to compute all features or give the outcomes. The main goal of the Decision tree (DT) is wanted to create a model which can predict the value of the target variable such as given data, or a dataset that helps to learn simple decision rules that are inferred from the given, or applied data features. So, a Decision tree (DT) can be seen as in the piecewise constant that could be approximate. Moreover, the Decision tree (DT) can be classified given input by segmenting the whole input space into their regions. To do the implementation, some libraries have to import the Decision tree (DT). So, they are such as,

import pandas as pd

from sklearn.tree import DecisionTreeClassifier

from sklearn.model\_selection import train\_test\_split

from sklearn import metrics

Code can be written in many ways a sample way can be such as

featu\_col = [‘a’, ‘b’]

x = pim[featu\_col]

y = pim.label

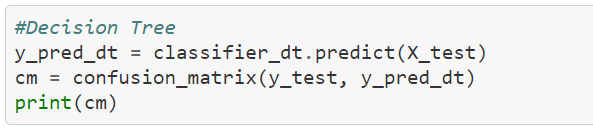
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_siz= 0.2, ran\_sta=1)

clf = DecisionTreeClassifier()

clf = clf.fi(x\_train, y\_train)

print(“Output: “ metrics.output\_sc(y\_test, y\_pre) )

From the implementation part of this study,



**Fig. 3-15** Decision Tree

* + 1. **Logistic Regression:**

One of the most popular Machine Learning algorithms is Logistic Regression that is coming from the Supervised Learning technique or deftness which is mainly used to predict the whole categorical dependent variable that is used from a given data set of the independent variables. Logistic Regression is also used in a logistic function that can map the input data or variables which is must be a categorical response or dependent input data or variables. However, generally, Regression analysis explains that it can be used to identify the explanatory or interpretative data or variable which is must be related to the response data or variable that is described the original form of the relationship-related and must have to provide an equation that is predicting the whole response data or variable from the explanatory data or variable. Moreover, Logistic Regression is applied when 70/30 train and test split of the given dataset is created that indicates a classification algorithm which is used to solve the binary classification problem or issues. So, the Logistic Regression classifier is normally used for the weighted combination of the given input feature that is passing all of them through a sigmoid function. The output of the Logistic Regression is the most probability between 0 and 1 and must be dependent on categorical data or variables. For this reason, the outcome must be the categorical or discrete value or standard. So, the value can be either Yes or No, 0 or 1, True or False but the probabilistic value is between 0 and 1. However, the Logistic Regression is quite similar to Linear Regression but there has a minor difference between those classifications on their used process. In there, Linear Regression is mainly used to solve Regression problems or issues whereas Logistic Regression is mostly used to solve classification problems or issues. In short, the Logistic Regression is indicating a significant Machine Learning algorithm because it has the most important ability that provides the probabilities and can classify the newly arrived or given data which is using the continuous and discrete datasets or variables. For the further procedure for implementation, there have to import some libraries for the Logistic Regression such as,

import matplotlib.pyplot as plt

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import classification\_report, confusion\_matrix

Code can be written in many ways a sample way can be such as

x = np.ar(12).rsha(-1, 1)

y = np.ar([0, 0, 0, 0, 0, 1, 1, 1, 1, 1,1, 1])

model = LogisticRegression(sol=’lib’, ran\_st=0)

model.fit(x, y)

LogisticRegression(c=1.0, cla\_wei=None, du=false, fit\_inter=True, inter\_scal=1,

11\_ra=None, max\_it=100, ran\_sta=0, sol=’lib’, to=0.0001, ver=0, war\_start=False )

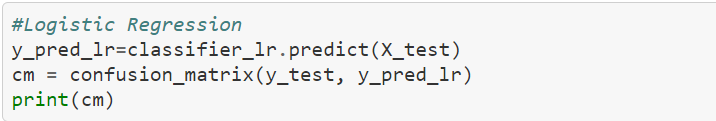
model = LogisticRegression(sol=’lib’, ran\_st=0). fit(x, y)

arr([0, 1])

arr([-1, 0.5407])

print(classification\_report(y, model.pre(x)))

From the implementation part of this study,



**Fig. 3-16** Logistic Regression

**Chapter 4**

# Results or findings

Afterward gone through the Methodology part, result, and discussion part comes and also plays a noticeable part to detect or complete “Conceptual Clarity and Terminology of Hate-Speech Intensity Scale” work because for this part the research can find its desired destination easily. So, after applying Linear Support Vector Machine (SVM), Long-Short Term Memory (LSTM), Bidirectional Long-Short Term Memory (Bi-LSTM), Decision Tree, and Logistic Regression into the implementing part, a good and noticeable number of accuracy or results come that are comparable. In the implementing part, both train and test datasets are also used. Furthermore, precision, recall, and F-1 score for each of the hate emotion classes or square as well as the total acquiring that are average accuracy is measured with its efficacy are given into the “Conceptual Clarity and Terminology of Hate-Speech Intensity Scale” proposed paper in its method section. Here, the detected hate emotions categories are such as stupidity, pathetic, greedy, and retarded. It is must say that this paper can consider those four types of hate emotions categories, all the list with their prediction and accuracy will be shown that how to find and computable outcome according to each or a sentence.

|  |  |
| --- | --- |
| **Hate Emotions** | **Probability** |
| Stupidity | 0.4898 |
| Pathetic | 0.1867 |
| Greedy | 0.2642 |
| Retarded | 0.0593 |

**Table 4-A:** Detailed evaluation using best model

Here the probability comes by using a rule that is,

P (Hate Emotion = Stupidity) = Total stupidity data / Total train data

As similar way all the probability are detected to the whole dataset.

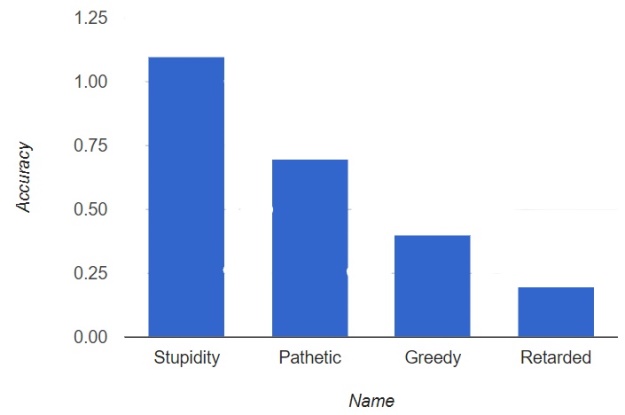
Whenever the “Conceptual Clarity and Terminology of Hate-Speech Intensity Scale” paper is trained then it got the best model or classifier that is Bidirectional Long-Short Term Memory (Bi-LSTM) according to other classifiers that are Linear Support Vector Machine (SVM), Long-Short Term Memory (LSTM), Decision Tree, and Logistic Regression. After completing or checking the actual accuracy, Bidirectional Long-Short Term Memory (Bi-LSTM) gives 87.07%, Support Vector Machine (SVM) gives 86.18%, and Long-Short Term Memory (LSTM) gives 82.09%, Decision Tree gives 72.23%, and Logistic Regression gives 85.16%. According to those accuracies, 87.07% is the highest which is coming from Bidirectional Long-Short Term Memory (Bi-LSTM). After observing table 4-A or the whole implementation part, this model or classification prediction for the Stupidity class or category is quietly gives very well accuracy but at the same time, this model or classification gives poor accuracy for the Retarded class or category. The cause of these types of unbalanced outcomes is the training dataset is unbalanced and also the Retarded class or categories have a few training data samples. However, it is the happiest information that it is most essential and replicates for the real-world situation, and then the applying approaches are mostly generalizable and analysable.

Whenever any sentence of the used dataset is considered, the sentence split is shown with its accuracy for each word of the sentence is like;

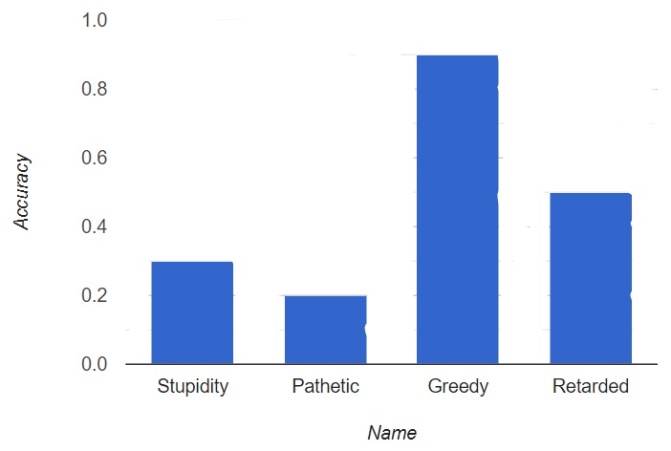
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| it | aint | nothing | to | cut | a | bitch | off |
| 0.004 | 0.002 | 0.402 | 0.0023 | 0.37 | 0.003 | 0.66 | 0.0012 |

**Table 4-B:** Sentence split and accuracy

There, if any random sentence or comment is choosing for detected hate emotion or speech then it will be define smoothly.



**Fig. 4-1:** Detect Hate emotions (Stupidity)



**Fig. 4-2:** Detect Hate emotions (Greedy)

The possible Accuracy =100 \* correct \_prediction/total sentences

Here, in Figure 4-1, and Figure 4-2, the Stupidity and the Greedy bar are showing the biggest outcome among the rest of the other three classes or categories of hate emotion, or speech. However, 87.07% accuracy is getting from Bidirectional Long-Short Term Memory (Bi-LSTM) classification overall others accuracy rating that is high enough for this paper.

|  |  |
| --- | --- |
| **Classifiers** | **Accuracy** |
| Linear SVM | 0.8618 |
| LSTM | 0.8209 |
| Bi-LSTM | 0.8707 |
| Decision Tree | 0.7223 |
| Logistic Regression | 0.8516 |

### Table 4-C: Classifiers

**Chapter 5**

# Discussion

### After completing the methodology and result section, this research gives outstanding outcomes for all the searchers and this developing society. Here, the “Conceptual Clarity and Terminology of Hate-speech Intensity Scale”, the paper represents a wonderful result by using five different approaches with their features converting speech or voice to text. There the noticeable thing is that the four Hate emotions named Stupidity, Pathetic, Greedy, and Retarded are detected frequently who is the classification of Hate-speech. Using the Linear Support Vector Machine (SVM) gives 86.18% accuracy, the Long-Short Term Memory (LSTM) gives 82.09% accuracy, the Bidirectional Long-Short Term Memory (Bi-LSTM) gives 87.07% accuracy, the Decision Tree gives 72.23% accuracy, and the Logistic Regression gives 85.16% accuracy. Therefore, seeing that, accuracy the Bidirectional Long-Short Term Memory (Bi-LSTM) gives the best accuracy among them with the 20485 train dataset. It is indicated that according to using this huge dataset the Bidirectional Long-Short Term Memory (Bi-LSTM) performance is best and most preferable for this research.

### Comparison:

### Additionally, there has a compared table which is built with some existing works from the literature review section and also adds the result of this paper. The comparison table is created with used approaches, and outcomes. Among them, the “Conceptual Clarity and Terminology of Hate-Speech Intensity Scale” give the best outcome by using a big dataset, features, and classifiers. For this reason, this also outstretched a significant impact of those types of research.

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Paper** | **Algorithm/Approach** | **Result**  **(Accuracy/ F1-Score)** |
| Ariadna Matamoros Fernandez and Johan Farkas | Racism, Hate Speech and Social Media: A Systematic Review and Critique | * CNN * RNN | * 40.38% * 35.58%   (Detect toxic word from 500 words) |
| Sergio Andres, Natalia Suarez, and Luz Magnolia | Internet, social media and online hate speech. Systematic review | * BERT fine-tunning * fastText embedding * BERT embedding | * 78.0% * 75.7% * 76.2%   (Detected hate speech from four different countries) |
| Salla-Maaria Laaksonen, Jesse-Haapoja, Teemu Kinnunen and Matti Nelimarkka” | The Datafication of Hate: Expectations and Challenges in Automated Hate Speech Monitoring | * Bag-of-words (BOW) * SVM | * 0.3140% * 0.7941%   (Detected hate & non-hate speech) |
| Wenjie Yin and Arkaitz Zubiaga | Towards generalizable hate speech detection: a review on obstracles and solutions | * GBDT * LSTM | * 65% * 55%   (Detected harmful words from 475 words) |
| Julian Riseh and Ralf Krestel | Toxic Comment Detection in Online Discussion | * CNN * LSTM * GRU | * 78% * 88% * 43%   (Detected hate speech from 1337) |
| Ashwin Geet d’Sa, Irina Illina and Dominique Fohr | Classification of Hate Speech Using Deep Neural Networks | * SVM * CNN * CRNN | * 65.8% * 82.2% * 72.3%   (Detected offensive & non-offensive word) |
| Alice Tontodilmamma, Eugenia Nissi, Annalina Sarra, and Lara Fontanella | Thirty years of research into hate speech: topics of interest and their evolution | * SVM * CNN * Uni-gram | * 77% * 56% * 58%   (Detected hate speech) |
| Our Work | Conceptual Clarity and Terminology of Hate-Speech Intensity Scale | * Linear SVM * LSTM * Bi-LSTM * Decision Tree * Logistic Regression | * 86.18% * 82.09% * 87.07% * 72.23% * 85.16%   (Detected hate speech & also find four class) |

**Table 5-A:** Comparison with relevant state of art work

Analyzing this comparison table, with this paper's outstanding result is unexpected because working with this big dataset is very much difficult. Previously, maximum researchers are not worked on those kinds of big datasets and have not detected four hate emotions separately. They are trying to detect that is the user data is hate, offensive, or harmful speech, word or not. For that reason, this outcome is unexpected but it brings a pearl of significant wisdom for future generation searchers. But it is a glad news that applying five different approaches the Bidirectional Long-Short Term Memory (Bi-LSTM) has shown good results. At the same time, this research has limitations like this paper only detected four emotions of Hate-speech whereas there might be several hate emotions. Again, this paper represents five different classifications but in this world, there have a few other models also that might be giving good accuracy from that. But the exception is higher from those five approaches but it can not be possible might be the used dataset are not well labeled, some libraries are not working properly, and so on. However, this paper gives satisfaction by providing this outstanding outcome by using this best model named the Bidirectional Long-Short Term Memory (Bi-LSTM) with 87.07% accuracy.

**Chapter 6**

# Conclusion

In the perfecting time, to detect or identify the hate emotion and its classification from English speech which is converted into text data for suitably uses, in the Machine Learning- based approaches namely the Linear Support Vector Machine (SVM), the Long-Short Term Memory (LSTM), the Bidirectional Long-Short Term Memory (Bi-LSTM), the Decision Tree, and the Logistic Regression algorithms are implemented frequently. After discovering the required or desirable corpus datasets, it gives 20485 comments, text, or words from the training dataset and 6455 comments, text, or words from the testing dataset. The possible probability of Stupidity (48.98%), Pathetic (18.67%), Greedy (26.42%), and Retarded (5.93%) are the four detecting hate emotion categories from datasets. Every unwanted symbol (e.g. commas, dots, hyphens, emoticons, and so on) are separated during the text sectionalisation period for using the dataset suitably. There, Retarded has a dataset of only (5.93%). Afterward, the Command-Line string split function is used in this paper to tokenize all the words or characters in statements. Moreover, for finding hate speech from text with four hate emotion classes, precision, recall, and F1-score are utilized smoothly for achieving the desired and preferable accuracy. The obtained accuracy of the Linear Support Vector Machine (SVM) is 86.18 percent, the Long-Short Term Memory (LSTM) is 82.09 percent, the Bidirectional Long-Short Term Memory (Bi-LSTM) is 87.07 percent, the Decision Tree is 72.23 percent, and the Logistic Regression is 85.26 percent when the whole procedures are completed. Here, with 87.07 percent accuracy, the Bidirectional Long-Short Term Memory (Bi-LSTM) gives the best outcome. For furthermore working, researchers can be used new or other preferable techniques to earn perfect or suitable outcomes. This paper only expresses four categories of Hate speech that are not so worked prior time but for future researchers can be detected new categories with or without those Hate speech emotions.

# Bibliography

[1] J. M.-D. María Antonia Paz1, Hate Speech: A Systematized Review, Universidad International de La Rioja (UNIR), October-December 2020: 1–.

[2] I. I. D. F. Ashwin Geet D'Sa, Towards Non-Toxic Landscapes: Automatic Toxic Comment Detection Using DNN, F-54000 Nancy, France: University de Lorraine, CNRS, Inria, LORIA.

[3] A. M.-F. a. J. Farkas2, Racism, Hate Speech, and Social Media: A Systematic Review and Critique, Vol. 22(2) 205–224: Television & New Media, 2021.

[4] N. S.-B. L. M. T. V. H. M. H. L. Sergio Andr´es Casta˜no-Pulgarína, Internet, social media and online hate speech. A systematic review, International Journal of Communication, Received 14 July 2020; Received in revised form 26 January 2021; Accepted 23 March 2021.

[5] J. H. T. K. M. N. a. R. P. 6. Salla-Maaria Laaksonen1\*, The Datafication of Hate: Expectations and Challenges in Automated Hate Speech Monitoring, Jyväskylä, Finland: Department of Language and Communication Studies, University of Jyväskylä, February 2020 | Volume 3.

[6] Hate Speech Detection using Transformer Ensembles on the HASOC dataset?, Szeged, Hungary: MTA-SZTE Research Group on Artificial Intelligence, 2019-02996.

[7] A. C. F. D. M. P. a. M. T. Fabio Del Vigna12, Hate me, hate me not: Hate speech detection on Facebook, Istituto di Informatica e Telematica, CNR, Pisa, Italy., October 23, 2016.

[8] S. B. S. G. K. B. Y. Al-Onaizan, NeuralWord Decomposition Models for Abusive Language Detection, Amazon AWS, USA, August 2019.

[9] W. Y. a. A. Zubiaga, Towards generalizable hate speech detection: a review on obstacles and solutions, United Kingdom: School of Electronic Engineering and Computer Science, Queen Mary University of London, London, 17 June 2021.

[10] J. R. a. R. Krestel, Toxic Comment Detection in Online Discussions, Hasso Plattner Institute, University of Potsdam.

[11] I. I.. F. Ashwin Geetd'Sa, Classification of Hate Speech Using Deep Neural Words, Université de Lorraine, CNRS, Inria, LORIA.

[12] A. M. H. O. V. Radu Meza, Targets of Online Hate Speech in Context. A Comparative Digital Social Science Analysis of Comments on Public Facebook Pages from Romania and Hungary, Romania: Communication Sciences, Babeș–Bolyai University, Cluj–Napoca, 27 February 2019.

[13] A. T. ·. E. N. ·. A. S. ·. L. Fontanella3, Thirty years of research into hate speech: topics of interest and their evolution, Scientometrics (2021) 126:157–179, 30 October 2020.

[14] J. R. R. K. a. A. L. Betty van Aken1, Challenges for Toxic Comment Classification: An In-Depth Error Analysis, Germany: 1Beuth University of Applied Sciences, Second Workshop on Abusive Language Online (ALW2), pages 33–42.

[15] F. D. M. P. M. T. Fabio Del Vigna, Hate me, hate me not: Hate speech detection on Facebook, Venice, Italy.: First Italian Conference on Cybersecurity (ITASEC17), 2017.

[16] 1. D. W. M. M. I. W. Thomas Davidson, Automated Hate Speech Detection and the Problem of Offensive Language, Doha, Qatar: Qatar Computing Research Institute, HBKU, 2017.

[17] E.Z.W.K.H.S.L Elizaveta Zinovyeva, Antisocial Online Behavior Detection Using Deep Learning, Berlin, Germany: International Research Training Group 1792, ISSN 2568-5619.

[18] S.A.S.S.Z.A.S.K.G.M Sindhu Abro, Automatic Hate Speech Detection Using Machine Learning: A Comparative Study, Sukkur, Pakistan: International Journal of Advanced Computer Science and Application, Vol 11, No, 8, 2020.

*End quote goes here.*