International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 11 Issue 1 ǁ January 2023 ǁ PP. 296-305

Recognition of Hate Mongering contents in Social Media using Global Vector (GloVe) plus BiLSTM and Cost Function Analysis

Prashant Dutta[1], Pranay Dutta[2] , Naveen Kumar Bharti[3]
1 Manager-IT, MPPKVVCL Jabalpur
2 Senior Staff Engineer , Intercontinental Exchange Pune
3 Manager-IT, MPPKVVCL Jabalpur

Abstract
The Internet has brought the peoples of distinct countries, ethnicity, caste, creed, dialect, race, gender, religion, thoughts and beliefs under one Umbrella of Social Media Online Platform. This Umbrella of Social Media Online Platform has enabled the common man to have a sneak peek into each other's life, religion, likes/dislikes, beliefs etc. Since the world consists of numerous societies having numerous beliefs, languages, caste/creed system etc, hence there is a huge probability that there can be a clash between thought processes & likes/dislikes. And whenever that clash happens in social media it happens as a result of offensive language and hate speeches.
If the clash of opinions is manifested by maintaining a social decorum then it can be accepted in a democratic way, but when the social decorum is neglected and the hate speech takes the shape of an offensive/abusive language then it is not acceptable in the society. But in the scenario depicted aforesaid when people express their offensive thoughts in various social media platforms then it is very hard for the agencies to pin point those particular Tweets/Shares/Photos/Posts etc which contain offensive language or hate speech. The Aim of this paper is to detect hate speech and offensive language through Machine learning algorithm termed as Global Vector (GloVe) + Bi-Long Short-Term Memory by using Python.
Keywords: *Social-Media, offensive language, hate speech, Global Vector (GloVe), Cost Function, Python, BiLSTM*

Date of Submission: 06-01-2023 Date of acceptance: 19-01-2023

I . Introduction

The social-media has witnessed a huge growth in data consumption. Each day Lakhs of users upload their contents(photos, videos etc) in the platforms like Facebook, WhatsApp,Instagram etc. Some Statistics are as Under :-

S. No	Social Media	Total Users	Data Processed Each minute/day/month
1	Facebook	2.5+ Billion	500+ TB
2	Twitter	350+ Million	700+ million tweets per day
3	Youtube	2+ Billion	450+ hours of new video every
4	Instagram	1.0+ Billion	100+ million photos and videos
5	Snapchat	350+ Million	4+ billion snaps each day

Table 1 Social Media Statistics[9]

From the Colossal amount of data as depicted in table1 it can be inferred that it's not cakewalk to identify Offensive language from the various Social media posts happening daily

Before making any further judgments on the negative and positive words we need to classify the negative words which can be further classified as hate-speech, offensive-language etc. The classification can be as under :-

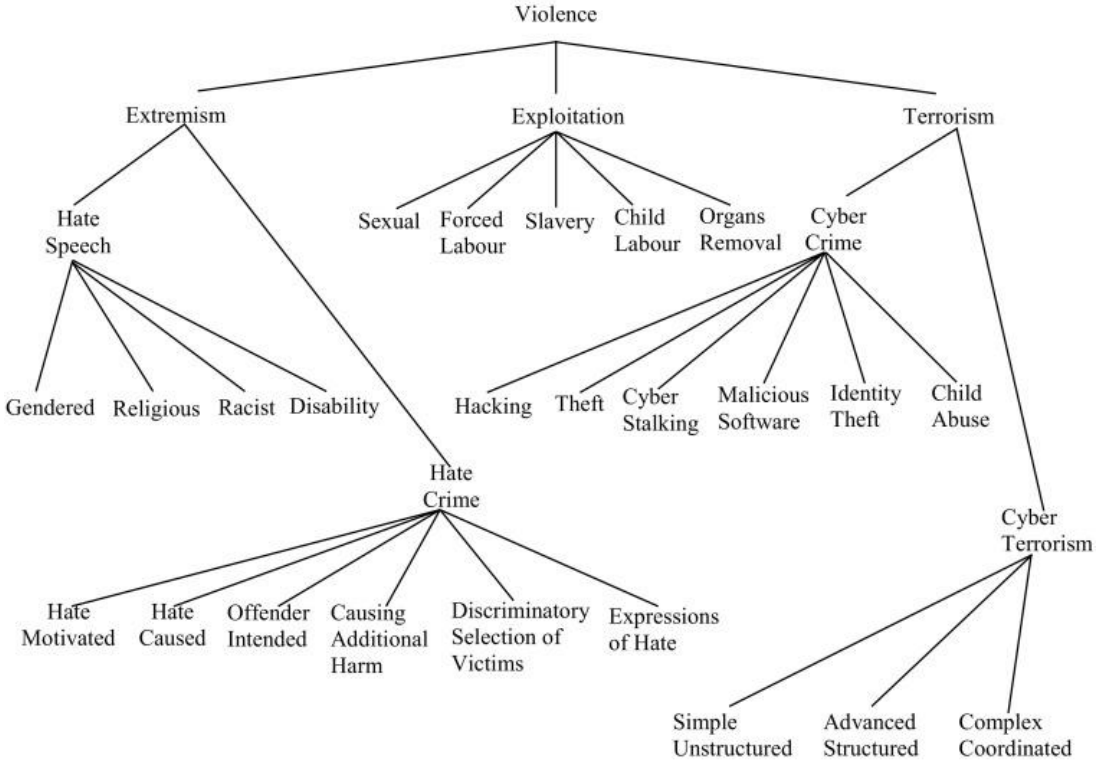


Fig 1 : Hate-Speech Classifications

1.1 Research Methodology

Our Research follows an intricate path towards recognizing the incident followed by sampling, categorization and sentiment analysis.

The Foul Language detection process has been largely divided into 5 different procedural steps :-

1.1.1 Creation of Data Sets

Out of many Social Media Platforms we are using Twitter as a platform to perform our research studies. Every second, on average, around 6,000 tweets are tweeted on Twitter, corresponding to over 350,000 tweets sent per minute, 500 million tweets per day and around 200 billion tweets per year[7].

The Data Set has been created using various sources, but the greater part has been taken from two renowned sources namely :-

(i) www.kaggle.com[10]

(ii) https://hatespeechdata.com[11]

The Data Set has finally been compiled into a CSV File. There are basically 7 columns to this data set.

Index_No	Count_Of_Tweets	Count_of_ Judgement	Count_of_ Offensive_Language	Count_of_Neither	Class_Label	Tweet_Body

The description of the column is as follows:

a. Index_No – It contains the Index_Value

b. Count_Of_Tweets – It contains number of tweets and retweets

c. Count_of_Judgement – No of Persons who claimed it to be offensive or not

d. Count_of_Offensive_Language - No of Persons who claimed it to be specifically offensive.

e. Count_of_Neither - No of Persons who remained Neutral

f. Class_Label – The class Label '-1' means not hate speech, '0' means neutral, '1' means hate speech and '2' means offensive language.

g. Tweet_Body – This contains the the entire tweet text.

1.1.2 Preparing Data to be Machine Readable.

Any data which is in the form of text cannot be fed into the machine learning algorithm for that it needs to be converted into binary form. But before that one more step has to be gone through and that is the step of data distillation and preprocessing.

1.1.2.1 Data Distillation and Preprocessing :- Just like Fossil fuels undergo Distillation and preprocessing to be finally converted into Diesel and Petrol , similarly twitter data needs to be made free of unwanted texts and annotations . The aforesaid text pre-processing makes far better classification-results. Therefore our data set was carefully prepared after applying pre-processing methods to remove all the irrelevant texts. Even we changed the Case by converting all the upper case to lower-cases. All the hashtags(#), "SPACE", Punctuations(; '' , . "" !)

1.1.3 Sampling of Data

In Statistics, the sampling-method / sampling-technique is the method of analyzing the population by collecting information and making inferences from that data. It is the source of the data where the sample-space is vast.

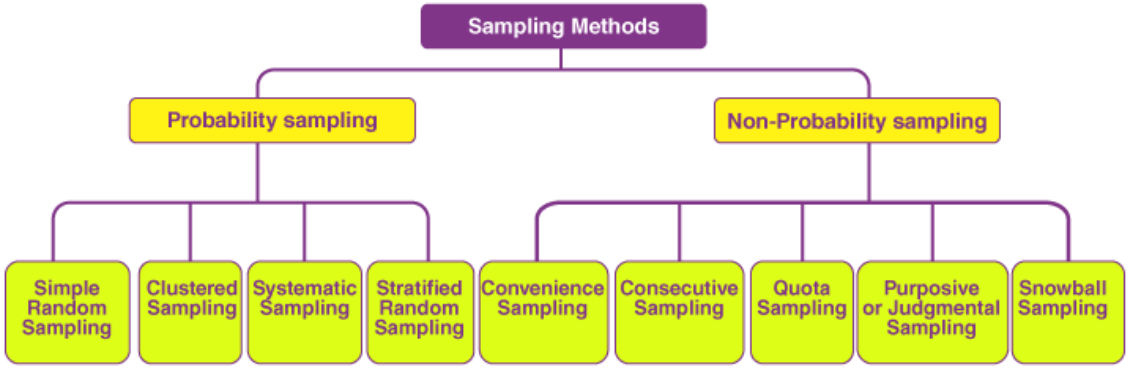


Fig 2 : Sampling Methods

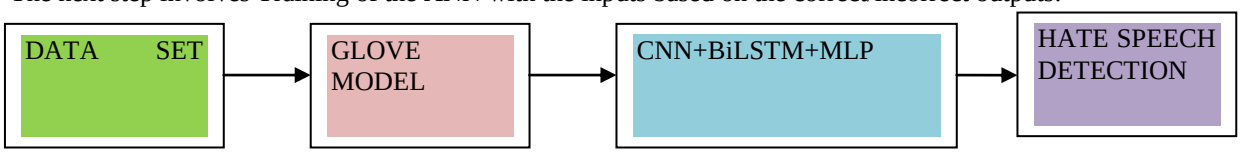
The Sampling technique which we are using is called as Stratified Sampling. In stratified sampling the sampling data is broken into Stratas. We have performed random stratified sampling on 7 stratas of the data sets which composed of texts having 0,10,20,30,40,50,60 %percent of sentences that features 'terms' from the list-of-keywords

S.No	Various Strata	Keywords
1	Race	Asian, White, Black, Nigga, Nigger
2	Gender	Nympho, Slut, Bimbo
3	Religion	Jewish, Buddhist
4	Nationality	Chineese, Chink, Spanish, Latin
5	Sexual Orientation	Queer, Lesbian, Gay
6	Class	Poor, Rich
7	Disability	Mental, Moron

Table 2 : Keywords

1.1.4 Machine Learning

The next step involves Training of the ANN with the inputs based on the correct/incorrect outputs.



Global-Vector(Glove) with CNN, Bi-Long Short-Term Memory, and Multilayer-Perceptron

In this technique, the text-dataset based on Eng is classified using the deep-learning model for the study of hate-speech. The word-embedding is prepared through the glove-model for doing the 'distributed word representation'.

Global-Vector(Glove) encodes a quantity into pretrained-weights. Next, deep-neural-networks use this embedding layer as an input-layer. 02 convolutionalLayers, 02 dropoutLayers, 02 max poolingLayers, a flattenLayer, and a denseLayer were all used in the 'CNN model'.
02 hidden layers of a multilayer-perceptron (MLP) incorporate the LSTM to the MLP. The LSTM NN processes speech embeddings one at a time even as maintaining the array of the words. Hyperbolic-tangent activation is used to hold the O/P of the 'LSTM neural network'. This is a 500-by-500-pixel vector. The MLP-Network has 03 layers:
(i)An I/P layer with 500-Neurons
(ii)A hidden layer with 1,500-Neurons and 100-Neurons activated by 'ReLU'
(iii)An O/P layer

GloVe comes into picture during co occurrence of words in a frame. In the co occurrence matrix if X words have co occurrence and 'Xi,j' symbolize ith and jth co uccurence. If xi = ∑j xi,j symbolize the no of counts the word comes in the framework of the ith word. The GloVe can be considered as a weighted least-square-regression with the cost-function as under:

$$J = \sum_{i,j=1}^{V} f(X_{i,j})(w_i^{\mathrm{T}}\tilde{w}_j + b_i + \tilde{b}_j - logX_{i,j})^2,$$

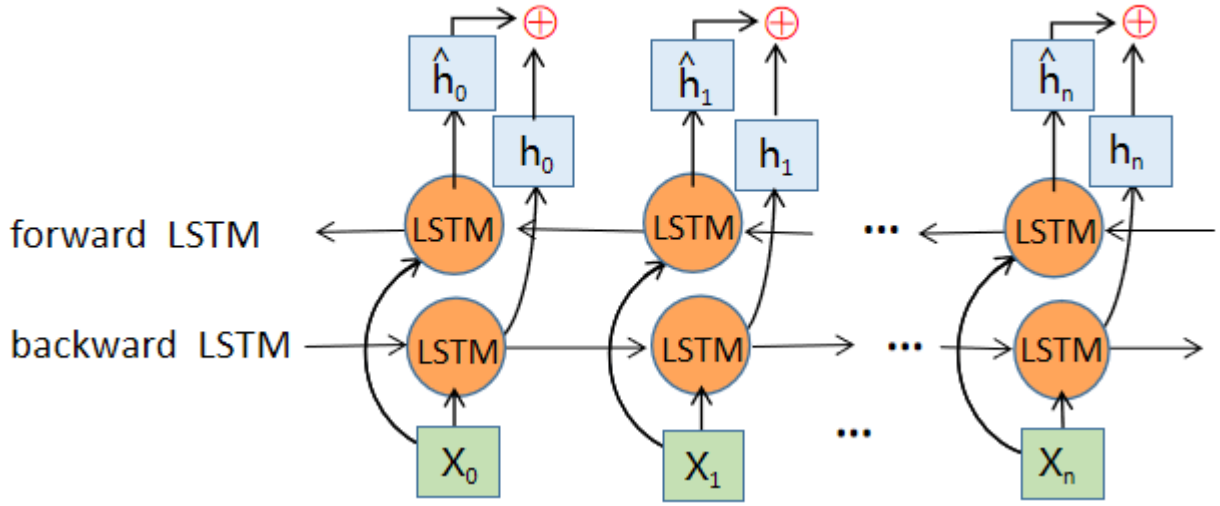
where V is the size of dictionary, (w ∈ Rd and w˜∈Rd are word-vectors, b and ˜b are biases). The weighted-function will have following characteristics :

1. f(0) = 0. If f is viewed as a continuous function, it should obey that limx→0 f(x)log2x is finite.
2. f(x) should be non-decreasing in case that rare co-occurrences are overweighted.
3. f(x) should be relatively small for large value of x, in case that frequent co-occurrences are overweighted.

f(x) is given as:

$$f(x) = \begin{cases} (x/x_{max})^{\alpha}, & x > x_{max} \\ 1, & otherwise \end{cases}$$

The word-embedding matrix work as weighted matrix in the model and hence the o/p of the model is a vector-> of inner yield of word-vectors.



Each word/speech has a constant illustration (wt ∈ Rd) and the likelihood of a word/speech is uttered at time (t) given the speech vector-> is represented with a log linear word-production-model

$$P[w \text{ emitted at time } t|c_t] \propto exp(c_t^T w_t).$$

1.1.5 Cost Function Analysis

Once we have trained our model then we need to verify its performance and it can be done using cost function. The cost function helps us to realize how good or bad our model has been trained or in other words is it working the way we desire it to work or not.

The cost-function shall be the minimum of the Root-Mean-Squared-Error of the Trained-Model and it is derived by subtracting the 'predicted values' from 'actual values'[12]

$$\text{Cost Function (J)} = \frac{1}{n}\sum_{i=0}^{n}(h_\theta(x^i) - y^i)^2$$

By Gradient descent we means we need to analyze the path in which the error is decreasing. The little difference between errors can be found by differentiating the cost-function and subtracting it from the previous-gradient descent.

$$\text{Gradient Descent } \theta_j = \theta_j - \alpha\frac{\partial J}{\partial \theta}$$

The cost-function of a Artificial neural network(ANN) will be the sum-of-errors in every layer[12]. This is achieved by identifying the error at each layer first and then summarizing the individual-error to get the total-error. In the end, it can represent a ANN with cost-function as :

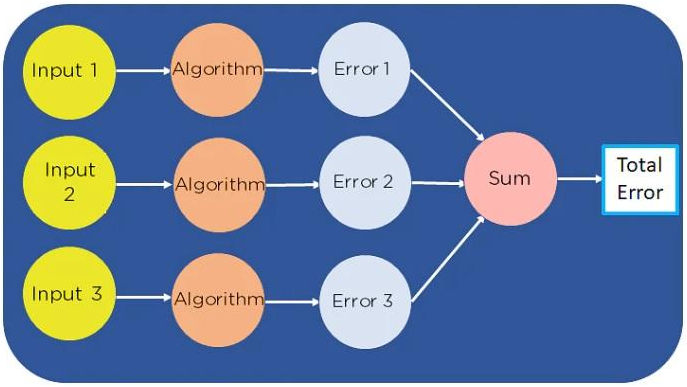


Fig 3 : ANN

Each layer of ANN will have its own cost function and each cost-function have its least error-value. We need to find the minimum value out of all local-minima. This value is called the global-minima.

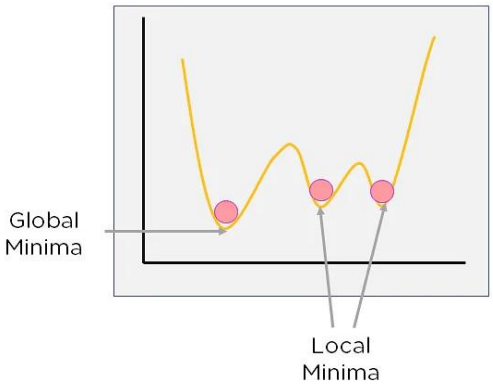


Fig : 4 Minima

The cost function for ANN is given as :

$$\text{Cost Function (J)} = \frac{1}{n}\sum_{i=0}^{n}(y^i - (mx^i + b))^2$$

Gradient-descent is just the differentiation of the cost-function. It is given as :

$$\text{Gradient Descent } (\frac{\partial J}{\partial \theta}) = \begin{bmatrix} \frac{1}{N}\sum_{i=0}^{n}(-2\, x_i(y_i - (mx_i + b)) \\ \frac{1}{N}\sum_{i=0}^{n}(-2\,(y_i - (mx_i + b)) \end{bmatrix}$$

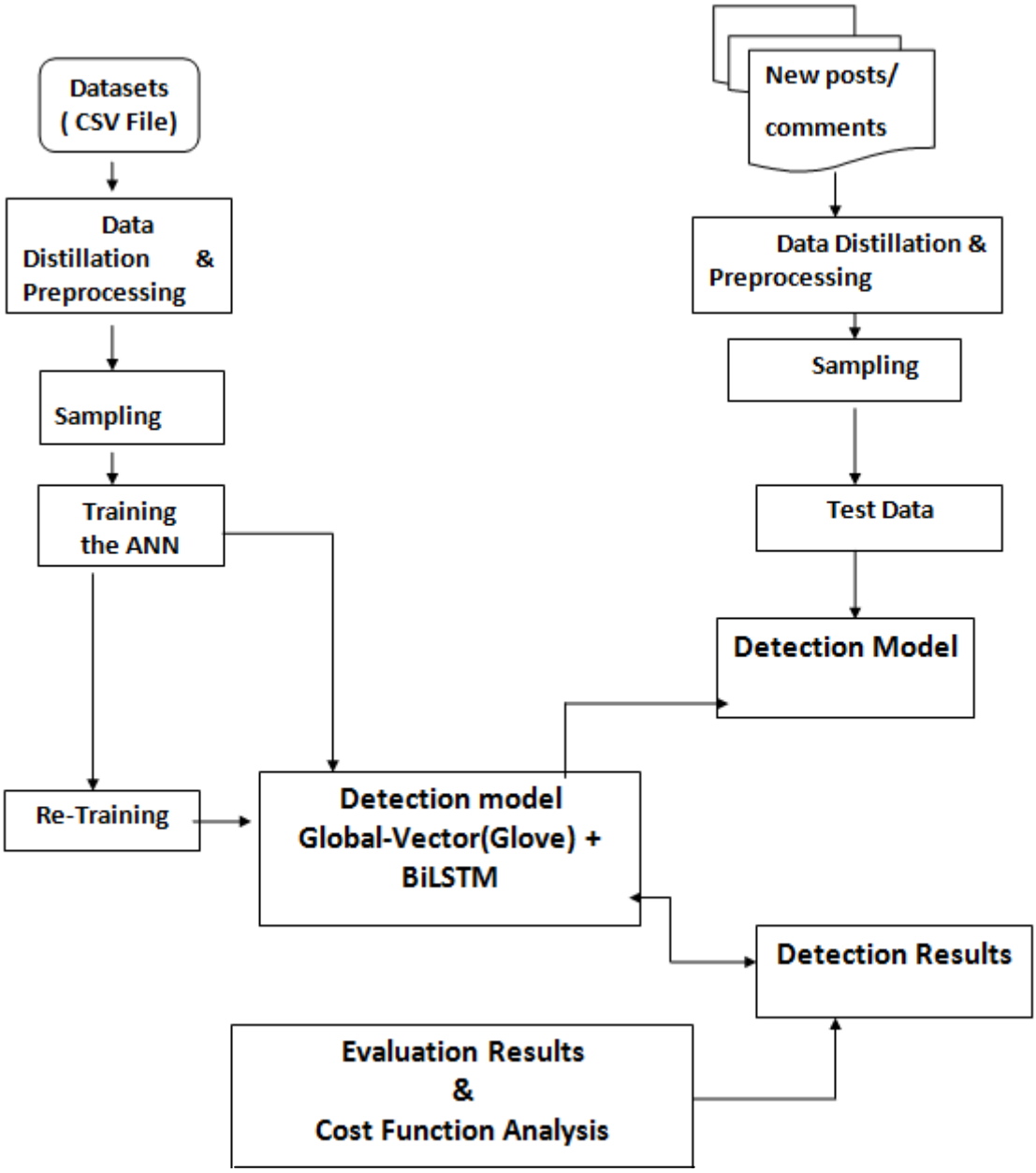


Fig 5 : The Complete Process of Hate Speech Detection

II. Software Implementation

2.1 Python Code : The Code is written in Python 3.8 and the required packages and libraries are as under :-

```
#-------------- For Glove --------------
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import keras
from keras.models import Sequential
from keras.initializers import Constant
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.optimizers import Adam
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D
from keras.layers import BatchNormalization
from keras.callbacks import ReduceLROnPlateau,CSVLogger
from tqdm.notebook import tqdm
import tensorflow as tf
from tensorflow.keras.models import Sequential,LSTM,BatchNormalization,Bidirectional
from tensorflow.keras.layers import Dense, Dropout, Activation,Flatten,Embedding,Conv1D, GlobalMaxPooling1D,
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, f1_score, confusion_matrix,classification_report
import shutil
```

2.2 Data Distillation

df_cleaned= pd.read_csv("DataSet.csv").iloc[:,1:]
df_cleaned = df_cleaned.dropna()

df_typo=pd.read_csv("DatSet.csv").iloc[:,1:]
df_typo = df_typo.dropna()

2.3 Train

```
for train_index, test_index, k, word_nums in K_FOLD_LIST:
  if data_name=='Clean':
    print("Model : CLEAN")
    LR = build_model(num_words_clean,embedding_matrix_clean)
  elif data_name=='Typo':
    print("Model : Typo")
    LR = build_model(num_words_typo,embedding_matrix_typo)
  else :
    print("**ERROR** Model Not Found")
  # Get Train, Test
  X_train, X_test = X[train_index], X[test_index]
  y_train, y_test = y.iloc[train_index], y.iloc[test_index]
```

2.4 Inference & Evaluate

```
LR.fit(X_train,y_train,epochs=10,batch_size=256)
print(f"{data_name} - {k} Trained")
fitted = LR.predict_classes(X_train)
fitted_proba = LR.predict_proba(X_train)
start = time.time()
pred = LR.predict_classes(X_test)
inference_time = time.time()-start
pred_proba = LR.predict_proba(X_test)
print(f"{data_name} - {k} Inferenced : {inference_time}s",end='\t')
# Evaluate
train_acc = accuracy_score(y_train.values,fitted)
train_auc = roc_auc_score(y_train.values,fitted_proba)
train_f1 = f1_score(y_train.values,fitted)
test_acc = accuracy_score(y_test.values,pred)
test_auc = roc_auc_score(y_test.values,pred_proba)
test_f1 = f1_score(y_test.values,pred)
print(f"train ACC : {train_acc} test ACC : {test_acc} test F1 : {test_f1}")


    eval = eval.append(pd.DataFrame([LR_list],columns=eval.columns))
eval
```

2.5 Output

id	comment_text	race	religion	sexual Orientation	caste	Gender bias	Disability
527278	The negro is a such a FATSO	1.0	0.0	0.0	0.0	1.0	1.0
624878	looks like they are male chauvinist	0.0	0.0	0.0	0.0	1.0	0.0
547258	Pope is in the town	0.0	1.0	0.0	0.0	0.0	0.0
825278	women Commandos	0.0	0.0	0.0	0.0	1.0	0.0
322218	The aborigines in Australia	1.0	0.0	0.0	1.0	0.0	0.0

III. Conclusion and Future Work

The present model only classifies the Hate-Speech into various categories but it cannot justify the severity of the Hate-Speech Text. Therefore in future the aim should be to train the CNN to detect the severity of the Hate-Speech. Also the present model works with text messages such as tweets/comments/posts etc. However the social media is extended far more than mere texts. The social media contains audio, videos, images which can be hate mongering. Therefore advanced algorithms needs to be developed which can perform video and image analysis and can do Hate Mongering detection in Images, Audios and Videos.

References :

[1]. Greevy, E. and A.F. Smeaton. Classifying racist texts using a support vector machine. in Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval. 2004. ACM.

[2]. Schmidt, A., & Wiegand, M. (2017, April). A survey on hate speech detection using natural language processing. In Proceedings of the Fifth International workshop on natural language processing for social media (pp. 1-10).

[3]. Shaikh, S. and S.M. Doudpotta, Aspects Based Opinion Mining for Teacher and Course Evaluation. Sukkur IBA Journal of Computing and Mathematical Sciences, 2019. 3(1): p. 34-43.

[4]. Debajyoti Chatterjee. Making neural machine reading comprehension faster. ArXiv, abs/1904.00796, 2019.

[5]. Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. ArXiv,abs/1503.02531, 2015.

[6]. Pytube, 2019. [Online]. Available at: https://python-pytube.readthedocs.io/en/latest.

[7]. https://blog.hootsuite.com/twitter-statistics/

[8]. Kwok I and Wang Y 2013 Locate the Hate: Detecting Tweets against Blacks. Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence (pp. 1621-1622). Association for the Advancement of Artificial Intelligence

[9]. . http://www.ijesrt.com/issues%20pdf%20file/Archive-2020/July-2020/8_SCRAPING%20OF%20SOCIAL%20MEDIA%20DATA%20USING%20PYTHON-3%20AND%20PERFORMING%20DATA%20ANALYTICS%20USING%20MICROSOFT%20POWER%20BI.pdf

[10]. https://hatespeechdata.com/

[11]. https://www.kaggle.com/datasets/mrmorj/hate-speech-and-offensive-language-dataset

[12]. https://www.enjoyalgorithms.com/blog/loss-and-cost-functions-in-machine-learning

[13]. https://towardsdatascience.com/classifying-hate-speech-an-overview-d307356b9eba

[14]. https://www.sciencedirect.com/science/article/pii/S1877050920310498
[15]. https://www.kaggle.com/datasets/usharengaraju/dynamically-generated-hate-speech-dataset
[16]. https://nlp.stanford.edu/projects/glove/