**INTRODUCTION**

**1.1 OVERVIEW OF THE PROJECT**

These days, a new definition of social interactions has emerged as a sign of the growth of the internet and cyberspace in the electronic era. Social platforms such as Telegram, Twitter, WhatsApp, Instagram, and Snapchat are used by 58.11% of the world's total population for interaction. By using social media platforms job opportunities have increased and economic growth also increased but many side effects like online bullying started taking place. So, the content which was shared with the people is to be safe and should not distract the people. Cyberbullying is an evil, malicious attack that can be committed by a group or an individual. Cyberbullying or Cyber harassment is also called Online bullying, which can be called bullying by electronic method.

Social media is an interactive tool that brings people together to share information. The primary function of Online Social Networks is to allow people to communicate virtually by using the internet. The rise in web and social media interactions has resulted in the effortless proliferation of offensive language and hate speech. This action involves repeated online insulting, harassing, or attacking a target verbally This content adversely affects their mental health, and demeans integrity of social networking platforms. Some Cyberbullying include Isolation, rage, psychological impacts of sadness, anxiety, scholastic difficulties, miserable thoughts, and self-harm, among a few of the repercussions of cyberbullying. Other effects include the behavioral implications of using drugs or alcohol and skipping out on routine tasks.

Cyberbullying may occur in a variety of ways using multi-mode input sources including text, photos, and videos. Mostly these studies are focused on evaluating textual information, such as comments and messages, because at first, cyberbullying was unstructured and the usage was purely text. Images, memes, and messages characters are still employed in communication, making it grueling to spot online bullying. Bullying events are now a well-organized, multi-media data source. Websites for social media networking place a strong focus on photo sharing. As a result of tendencies, cyberbullying activities among sufferers switch from text to visual. Most of the research is limited only to some data like only text. There is an urgent need to identify and prevent cyberbullying to make available a safe platform for digital users [7]. So, in this paper, we took data from both image and text we used LSTM and VGG16 algorithms for finding the bullying.

**1.2 Feasibility study**

A feasibility Study is a high-level capsule version of the entire process intended to answer several questions: What is the problem? Is there any feasible solution to the given problem? Is the problem even worth solving? A feasibility study is conducted once the problem is clearly understood. A feasibility study is necessary to determine that the proposed system is Feasible by considering the technical, Operational, and Economical factors. By having a detailed feasibility study the management will have a clear-cut view of the proposed system. The following feasibilities are considered for the project to ensure that the project is variable and it does not have any major obstructions. Feasibility study encompasses the following things.

• Technical Feasibility

• Economical or financial feasibility

• Operational feasibility

**1.2.1 Technical Feasibility**

In this step, we verify whether the proposed systems are technically feasible or not. i.e., all the technologies required to develop the system are available readily or not. Technical Feasibility determines whether the organization has the technology and skills necessary to carry out the project and how this should be obtained. The system can be feasible because of the following grounds.

• All necessary technology exists to develop the system.

• This system is flexible and it can be expanded further.

• This system can give guarantee accuracy, ease of use, and reliability.

• Our project is technically feasible because all the technology needed for our project is readily available.

**1.2.2 Economical or financial feasibility**

In this step, we verify which proposal is more economical. We compare the financial benefits of the new system with the investment. The new system is economically feasible only when the financial benefits are more than the investments and expenditures. Economical Feasibility determines whether the project goal can be within the resource limits allocated to it or not. It must determine whether it is worthwhile to process the entire project or whether the benefits obtained from the new system are not worth the costs. Financial benefits must be equal to or exceed the costs. In this issue, we should consider:

• The cost of h/w and s/w for the class of application being considered.

• The development tools.

• The cost of maintenance etc.

Our project is economically feasible because the cost of development is very minimal when compared to the financial benefits of the application.

**1.2.3 Operational Feasibility**

In this step, we verify different operational factors of the proposed systems like manpower, time, etc., whichever solution uses fewer operational resources, is the best operationally feasible solution. The solution should also be operationally possible to implement. Operational Feasibility determines if the proposed system satisfied user objectives and could be fitted into the current system operation.

• The methods of processing and presentation are completely accepted by the clients since they can meet all user requirements.

• The clients have been involved in the planning and development of the system.

**1.3 Scope**

**2 LITERATURE SURVEY**

**2.1 Existing System & Drawbacks**

In recent years cyberbullying activities became more in social media. So the negativity increases on these social media platforms. In the existing system, the project can able to detect abusive messages that are posted on social media like Facebook, Twitter, Instagram, etc.., Machine learning and deep learning are used to find bullying messages. These algorithms were used to find those abusive messages and the user who posted them on social media also be recognized. Most of the models are considering only a single dataset with a limited no of data and data is not clean and considered only a single dataset only not multiple. The dataset is based on a few keywords only. Time consumption in model building. Most of the models use BOW i.e., treating each word as a separate one does not save the meaning or the context of the word. There are so many studies correlated to cyberbullying detection and prevention using text mining by classifying comments or messages as abuse or not. Using supervised learning, like using N-grams i.e., for labeling, and using TF-IDF i.e., for weighing.

**2.2 Proposed System & Advantages**

As fast as technology grows crime rate also grows in different ways. To decrease the crime rate we are detecting abusive messages and images that are posted on social media. The system would first need to collect a large dataset of text and images from social media platforms that can be used to train the machine learning models. This dataset would need to include a mix of both legitimate and illicit content. The database contains the trained data which is used to identify whether the message is abusive language or not. Not only messages but images are also classified as safe or unsafe.

2.2.1 DATASET

The dataset contains text and image data. The text contains the attributes like comments and checks whether they are abused or not. If they are abused, it is represented by 1; if they are not, it is represented by 0. Images are classified into two categories i.e., safe, and unsafe images. The total number of unsafe images gathered is 600 and the safe images gathered are 700.

This system takes text and images as inputs. LSTM is applied to text to find whether messages or texts are abusive or not. VGG16 is applied to images to find whether the image is safe or unsafe.

2.2.2 TEXT-BASED CLASSIFICATION USING LSTM

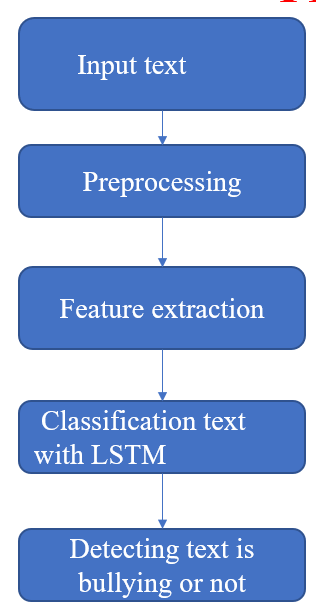


Fig. 1. Flow chart for text classification

Here the data cleaning is done by removing symbols, special symbols, stop words, and unnecessary words. The verbs and adjectives are sent to the LSTM algorithm for checking.

B. CLASSIFICATION OF TEXT USING LSTM

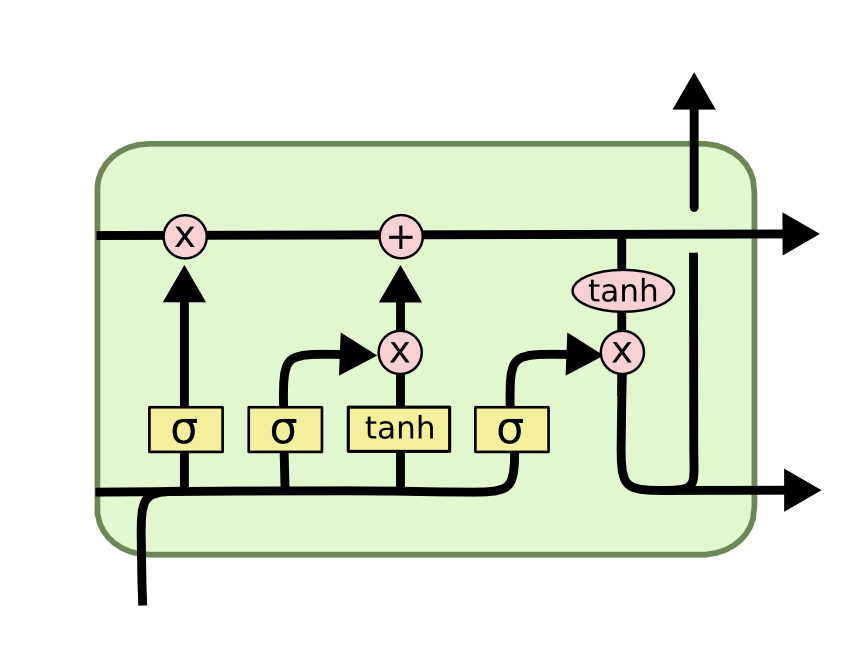
A general **LSTM**unit is composed of a cell, an input gate, an output gate, and a forget gate. The cell remembers values over arbitrary time intervals, and three gates regulate the flow of information into and out of the cell. LSTM is well-suited to classify, process, and predict the time series given of unknown duration.

**1. Input gate-** It discovers which value from input should be used to modify the memory. The sigmoid function decides which values to let through 0 or 1. And **tanh** function gives weightage to the values which are passed, deciding their level of importance ranging from

**-1** to**1.**

**2. Forget gate-** It discovers the details to be discarded from the block. A sigmoid function decides it. It looks at the previous state **(ht-1)** and the content input (Xt) and outputs a number between 0(omit this) and 1(keep this) for each number in the cell state **Ct-1**.

**3. Output gate-** The input and the memory of the block are used to decide the output. The sigmoid function decides which values to let through 0 or 1. And **tanh** function decides which values to let through 0, 1. And tanh function gives weightage to the values which are passed, deciding their level of importance ranging from **-1 to 1** and multiplied with an output of **sigmoid.**



2.2.3 IMAGE-BASED CLASSIFICATION USING VGG16

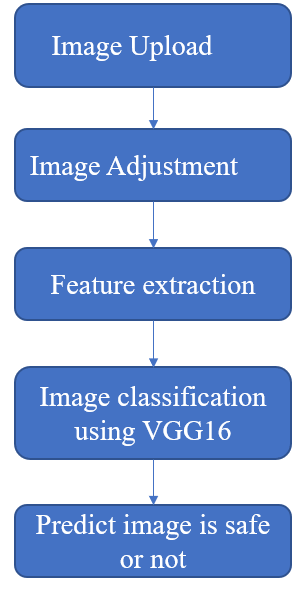


Fig. 2. Flowchart for image classification

VGG16 is one of the most used CNN architectures recently. It was developed by VG Team at the University of Oxford. It is a deep learning process that contains a total of 16 layers. It mainly contains a convolution layer, a max pooling layer, and a dense layer.

Here the images are converted into RGB and are initially adjusted to 224\*224 size.

1.Convolutional layer

A Convolutional layer is a particular kind of layer in a convolutional neural network that carries out the convolution operation. It is used to extract features from the input image by applying multiple filters to it. Each filter is designed to capture a specific type of feature from the input such as edges, texture, and patterns the result of the convolution operation is a feature map that represents the presence of these features in the input image.

2.ReLU activation Function:

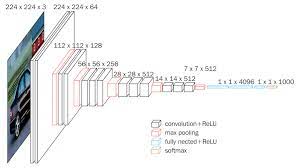
While this procedure represents all the inputs into numbers with zero and the positive values are retained. ReLU activation function provides major benefits in solving the possible vanishing issue grading issues that might arise when employing activation functions like nonlinear functions. Due to the vanishing gradient problem, the gradient loss between the relation to the parameters becomes extremely small making it difficult to train the model. The ReLU activation function also provides non-linearity to the model to appropriate complex relationships in the data

3. Max pooling layer

CNN frequently employed Max pooling a down-sampling procedure to procedure, to lower the pixel density of the extracted features while preserving the most crucial data. The feature maps are divided into a grid of distinct, non-overlapping areas, and the greatest values of each region are used as the new values of the region. This results in a new future map with samples spatial dimensions, but with the same number of channels as the original feature map.

4. Fully Connected Layer

The fully connected layer of a traditional neural network connects every neuron in the previous stage to every neuron in the current layer.



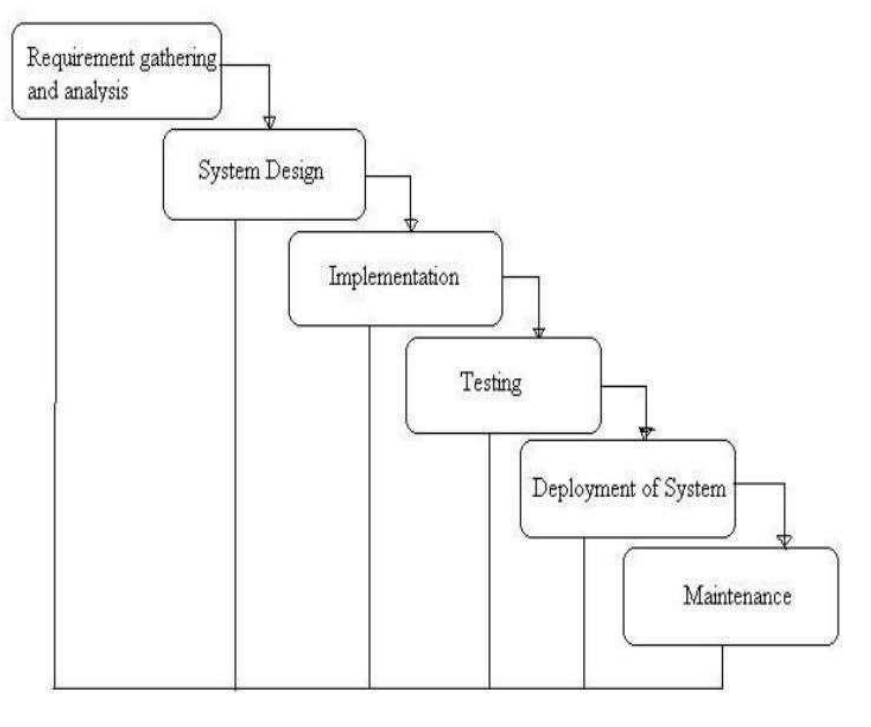
**3. SYSTEM ANALYSIS**

**3.1 Overview of System Analysis**

The model that is being followed is the WATERFALL MODEL, which states that the phases are organized in a linear order. First of all, the feasibility study is done. Once that part is over the requirement analysis and project planning begins. If a system exists and modification and addition of a new module are needed, analysis of the present system can be used as the basic model.

The design starts after the requirement analysis is complete and the coding begins after the design is complete. Once the programming is completed, the testing is done. In this model, the sequence of activities performed in a software development project is Requirement Analysis, Project Planning, System design, Detail design, Coding, Unit testing, System integration & testing.

Here the linear ordering of these activities is critical. The end of the phase and the output of one phase is the input of another phase. The output of each phase is to be consistent with the overall requirement of the system. Some of the qualities of the spiral model are also incorporated like after the people concerned with the project review completion of each phase of the work.



Development Model

**3.1.1 Requirement Gathering and Analysis**

All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.

**3.1.2 System Design**

The requirement specifications from the first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.

**3.1.3 Implementation**

With inputs from the system design, the system is first developed in small programs called units, which are integrated into the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.

**3.1.4 Integration and Testing**

All the units developed in the implementation phase are integrated into a system after the testing of each unit. Post integration the entire system is tested for any faults and failures.

**Unit Testing**

It is done by the developer itself in every stage of the project and fine-tuning the bug and module-predicated additionally done by the developer only here we are going to solve all the runtime errors

**Manual Testing**

As our Project is on academic Leave, we can do any automatic testing so we follow manual testing by endeavor and error methods

**3.1.5 Deployment of System**

Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.

**3.1.6 Maintenance**

Some issues come up in the client environment. To fix those issues, patches are released. Also, to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment. All these phases are cascaded to each other in which progress is seen as flowing steadily downward through the phases.

**3.2 Software Used in The Project**

What things do you need to install the software and how do install them:

1. python 3.7.1
2. Keras
3. Tensorflow

Process for installation refer: https://www.python.org/downloads/

**SOFTWARE REQUIREMENTS:**

• Operating System: Windows 7

• Language : Python • Software : Python 3

• Frontend: HTML

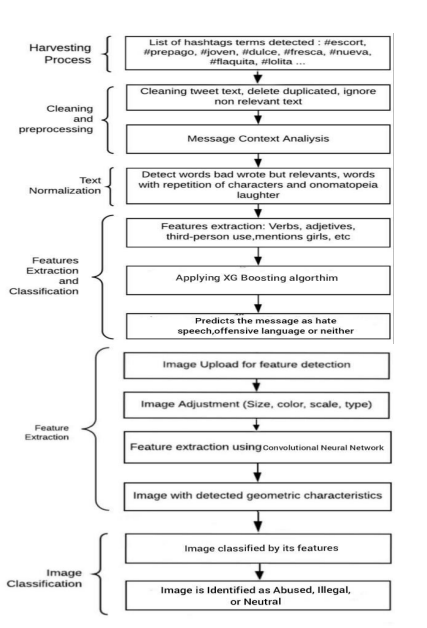
**HARDWARE** **REQUIREMENT**:

• Processor - core i7

• RAM - 12GB

• Hard Disk - 256GB

1. **SYSTEM DESIGN**



A.PREPROCESSING AND FEATURE EXTRACTION

Here the data cleaning is done by removing symbols, special symbols, stop words, and unnecessary words. The verbs and adjectives are sent to the LSTM algorithm for checking.

B. CLASSIFICATION OF TEXT USING LSTM

LSTM detects the important feature at the starting phase instead of the final phase. It has mainly four gates. The first gate is the forgotten gate it will check and decide what information is relevant and irrelevant. It will throw the irrelevant information away from the cell. Here 1 represents “completely keep the data” and 0 represents “Completely get rid of data”. The input decides which data or information should be stored. It finds which data is necessary for the current state. It contains a sigmoid layer and its value ranges between 0 to 1. It also contains the Tanh layer. The output gate decides what the next hidden state should be and here initially information is passed to the sigmoid and next to the modified tanh. At last, it produces a final output.

1. **CODING & IMPLEMENTATION**

**CODE**

**Trail.py**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.preprocessing import LabelEncoder

from keras.preprocessing.sequence import pad\_sequences

from tensorflow.keras.optimizers import RMSprop

from keras.preprocessing.text import Tokenizer

from keras.preprocessing import sequence

from keras.callbacks import EarlyStopping

import pickle

import RNN

if(1):

file=open("model.pickle","rb")

model1=pickle.load(file)

file.close()

if(0):

df = pd.read\_csv("train.csv")

dft=pd.read\_csv("test\_with\_solutions.csv")

df.head()

dft.head()

df.drop(['Date'],axis=1,inplace=True)

dft.drop(['Date','Usage'],axis=1,inplace=True)

sns.countplot(df.Insult)

plt.xlabel('Label')

plt.title('Number of non-bully vs bully messages in trianing dataset')

plt.show()

X = df.Comment

Y = df.Insult

le = LabelEncoder()

Y = le.fit\_transform(Y)

Y = Y.reshape(-1,1)

le = LabelEncoder()

#X\_train,X\_test,Y\_train,Y\_test = train\_test\_split(X,Y,test\_size=0.15)

X\_train,Y\_train=X,Y

max\_words = 1000

max\_len = 100

tok1 = Tokenizer(num\_words=max\_words)

tok1.fit\_on\_texts(X\_train)

fl=open("tok1.pickle","wb")

pickle.dump(tok1,fl)

fl.close()

sequences = tok1.texts\_to\_sequences(X\_train)

sequences\_matrix = sequence.pad\_sequences(sequences,maxlen=max\_len)

model1 = RNN.RNN()

model1.summary()

model1.compile(loss='binary\_crossentropy',optimizer=RMSprop(),metrics=['accuracy'])

k=model1.fit(sequences\_matrix,Y\_train,batch\_size=128,epochs=10,

validation\_split=0.2,callbacks=[EarlyStopping(monitor='val\_loss',min\_delta=0.0001)])

f=open("model1.pickle","wb")

pickle.dump(model1,f)

f.close()

# loss

plt.plot(k.history['loss'],label="train loss")

plt.plot(k.history['val\_loss'],label="val loss")

plt.ylabel('Loss')

plt.xlabel('Epoch')

plt.legend()

plt.show()

plt.savefig("LossVal\_loss")

# accuracy

plt.plot(k.history['accuracy'],label="train accuracy")

plt.plot(k.history['val\_accuracy'],label="val accuracy")

plt.ylabel('Accuracy')

plt.xlabel('Epoch')

plt.legend()

plt.show()

plt.savefig("AccVal\_accuracy")

max\_words = 1000

max\_len = 100

X\_test=[input("Enter ")]

print(X\_test)

tok = pickle.load(open("tok.pickle","rb"))

test\_sequences=""

test\_sequences = tok.texts\_to\_sequences(X\_test)

print(test\_sequences)

test\_sequences\_matrix = sequence.pad\_sequences(test\_sequences,maxlen=max\_len)

print(test\_sequences\_matrix)

ans=model1.predict(test\_sequences\_matrix,batch\_size=None,verbose=0,steps=None)

print(ans)

st=""

st=st.join(str(ans[0][0]))

print(st)

if float(st)>0.5:

print("hate speech")

else:

print("normal")

LSTM.py

from keras.models import Model

from keras.layers import LSTM, Activation, Dense, Dropout, Input, Embedding

max\_words = 1000

max\_len = 100

def RNN():

inputs = Input(name='inputs',shape=[max\_len])

layer = Embedding(max\_words,50,input\_length=max\_len)(inputs)

layer = LSTM(64)(layer)

layer = Dense(256,name='FC1')(layer)

layer = Activation('relu')(layer)

layer = Dropout(0.5)(layer)

layer = Dense(1,name='out\_layer')(layer)

layer = Activation('sigmoid')(layer)

model = Model(inputs=inputs,outputs=layer)

return model

hello.py

from keras.applications.vgg16 import VGG16,preprocess\_input

from keras.preprocessing.image import ImageDataGenerator,image

from keras.layers import Dense,Activation,Flatten,Dropout

from keras.models import Sequential,Model,load\_model

from keras import optimizers

from keras.callbacks import ModelCheckpoint,EarlyStopping

import numpy as np

import matplotlib.pyplot as plt

from tensorflow import keras

model = keras.models.load\_model('Final\_model\_vgg.h5')

#define height and width of the image

height=300

width=300

base\_model=VGG16(weights='imagenet',include\_top=False,input\_shape=(height,width,3))#define directory containing training and validation data)

#define directory containing training and validation data

train\_dir=r"C:\Users\NIVITHA\Desktop\Anti-Cyber-Bullying-master\dataset\train"

validation\_dir=r"C:\Users\NIVITHA\Desktop\Anti-Cyber-Bullying-master\dataset\val"

#number of batches the data has to be divided into

batch\_size=32

#create datagen and generator to load the data from training directory

train\_datagen=ImageDataGenerator(preprocessing\_function=preprocess\_input,rotation\_range=90,horizontal\_flip=True,vertical\_flip=True)

train\_generator=train\_datagen.flow\_from\_directory(train\_dir,target\_size=(height,width),batch\_size=batch\_size)

#create datagen and generator to load the data from validation directory

validation\_datagen=ImageDataGenerator(preprocessing\_function=preprocess\_input,rotation\_range=90,horizontal\_flip=True,vertical\_flip=True)

validation\_generator=validation\_datagen.flow\_from\_directory(validation\_dir,target\_size=(height,width),batch\_size=batch\_size)

#our own model which will be added onto the VGG16 model

def build\_finetune\_model(base\_model,dropout,fc\_layers,num\_classes):

for layer in base\_model.layers:

layer.trainable=False

x=base\_model.output

x=Flatten()(x)

for fc in fc\_layers:

x=Dense(fc,activation='relu')(x)

x=Dropout(dropout)(x)

predictions=Dense(num\_classes,activation='softmax')(x)

finetune\_model=Model(inputs=base\_model.input,outputs=predictions)

return finetune\_model

class\_list=['safe','unsafe'] #the labels of our data

FC\_Layers=[1024,1024]

dropout=0.5

finetune\_model=build\_finetune\_model(base\_model,dropout=dropout,fc\_layers=FC\_Layers,num\_classes=len(class\_list))

if(1):

#define number of epochs(the number of times the model will be trained) and number of training images

num\_epochs=10

num\_train\_images=692

#checkpoint in case anything goes wrong

checkpoint=ModelCheckpoint("Final\_model\_vgg.h5",monitor='val\_acc',verbose=1,save\_best\_only=True,save\_weights\_only=False,mode='auto',period=1)

early=EarlyStopping(monitor='val\_acc',min\_delta=0,patience=40,verbose=1,mode="auto")

#compile the model before using

finetune\_model.compile(loss="categorical\_crossentropy",optimizer=optimizers.SGD(lr=0.000001,momentum=0.9),metrics=['accuracy'])

#train the model

k=finetune\_model.fit\_generator(generator=train\_generator,steps\_per\_epoch=num\_train\_images//batch\_size,epochs=num\_epochs,validation\_data=validation\_generator,validation\_steps=1,callbacks=[checkpoint,early])

#save the model

finetune\_model.save(r"C:\Users\NIVITHA\Desktop\Anti-Cyber-Bullying-master\Final\_model\_vgg.h5")

if (0):

#testing the model

from keras.preprocessing import image

import matplotlib.image as mpimg

img = image.load\_img(r"C:\Users\NIVITHA\Desktop\Anti-Cyber-Bullying-master\dataset\test\5.jpg",target\_size=(300,300))

img = np.asarray(img)

plt.imshow(mpimg.imread(r"C:\Users\NIVITHA\Desktop\Anti-Cyber-Bullying-master\dataset\test\5.jpg"))

plt.ion()

plt.show()

img = np.expand\_dims(img, axis=0)

#finetune\_model.load\_weights(r"C:\Users\NIVITHA\Desktop\Anti-Cyber-Bullying-master\Final\_model\_vgg.h5")

output=finetune\_model.predict(img) #predicting the image using model created

if(output[0][0]>output[0][1]): #comparison

print("safe")

else:

print("unsafe")

app.py

from flask import Flask, render\_template,request

from flask import Flask,render\_template,url\_for,request

from keras.preprocessing.sequence import pad\_sequences

from keras.preprocessing import sequence

import time

import webbrowser

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from keras.preprocessing import image

import matplotlib.image as mpimg

from keras.models import load\_model

from datetime import datetime

import warnings

warnings.filterwarnings("ignore")

import matplotlib.image as mpimg

import cv2

from PIL import Image

import numpy as np

from skimage import transform

import pickle

import joblib

from tensorflow import keras

from keras.preprocessing.sequence import pad\_sequences

model = keras.models.load\_model('Final\_model\_vgg.h5')

height=300

width=300

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.preprocessing import LabelEncoder

from keras.preprocessing.sequence import pad\_sequences

from tensorflow.keras.optimizers import RMSprop

from keras.preprocessing.text import Tokenizer

from keras.preprocessing import sequence

from keras.callbacks import EarlyStopping

import pickle

import RNN

app = Flask(\_\_name\_\_)

@app.route('/')

def load(filename):

np\_image = Image.open(filename)

np\_image = np.array(np\_image).astype('float32')/255

np\_image = transform.resize(np\_image, (300, 300, 3))

np\_image = np.expand\_dims(np\_image, axis=0)

img=mpimg.imread(filename)

plt.imshow(img)

return np\_image

start = time.time()

from PIL import Image

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

return render\_template("index.html", name="Tariq")

@app.route('/predict',methods=['POST'])

def predict():

file=open("model.pickle","rb")

model1=pickle.load(file)

file.close()

print('hi')

max\_words = 1000

max\_len = 100

if request.method == 'POST':

message = request.form['message']

data = [message]

print(data)

tok = pickle.load(open("tok.pickle","rb"))

test\_sequences=""

test\_sequences = tok.texts\_to\_sequences(data)

print(test\_sequences)

test\_sequences\_matrix = sequence.pad\_sequences(test\_sequences,maxlen=max\_len)

print(test\_sequences\_matrix)

ans=model1.predict(test\_sequences\_matrix,batch\_size=None,verbose=0,steps=None)

print(ans)

st=""

st=st.join(str(ans[0][0]))

print(st)

if float(st)>0.5:

print("illicit")

lb='illicit'

else:

print("normal")

lb='normal'

return render\_template("prediction.html", data=lb)

@app.route('/prediction', methods=["POST"])

def prediction():

img = request.files['img']

img.save('./test.jpg')

img = image.load\_img("test.jpg",target\_size=(300,300))

img = np.asarray(img)

plt.imshow(mpimg.imread("test.jpg"))

plt.ion()

plt.show()

img = np.expand\_dims(img, axis=0)

#finetune\_model.load\_weights(r"C:\Users\NIVITHA\Desktop\Anti-Cyber-Bullying-master\Final\_model\_vgg.h5")

output=model.predict(img) #predicting the image using model created

if(output[0][0]>output[0][1]): #comparison

print("Unsafe")

lbl='Unsafe'

else:

print("safe")

lbl='Safe'

print("Overall the pic is identified as :" + lbl)

return render\_template("prediction.html", data=lbl)

if \_\_name\_\_ =="\_\_main\_\_":

webbrowser.open('http://127.0.0.1:5000/')

app.run("127.0.0.1", port=5000, debug=False,threaded=False)

index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Flask App</title>

</head>

<style>

body{

text-align: center;

font-family: monospace;

top: 50%;

left: 50%;

position: absolute;

transform: translate(-50%,-50%);

}

input[type="submit"]{

background: none;

display: block;

border: 0;

margin: 20px auto;

text-align: center;

border: 2px solid #3498db;

border-radius: 24px;

border-color:#2ecc71;

padding: 10px 40px;

color: rgb(22, 21, 21);

outline: none;

transition: 0.24s;

cursor: pointer;

}

input[type="submit"]:hover{

background-color: #2ecc71;

}

input[type="file"]{

background-color: rgb(46, 80, 12);

color: white;

padding: 16px;

border-radius: 6px;

margin: 10px;

font-size: xx-large;

}

h1{

font-size: 35px;

}

</style>

<body>

<header>

<div class="container">

<div id="brandname">

<h1>CYBERBULLYING DETECTION</h1>

</div>

</div>

</header>

<div class="ml-container">

<form action="/predict" method="POST">

<h2><p>Enter Message</p></h2>

<textarea name="message" rows="6" cols="50"

placeholder="Enter Your Message..." ></textarea>

<br/>

<input type="Submit" class="btn-info" value="Predict">

</form>

</div>

<h2>Browse Image</h2>

<form action="prediction" method="POST" enctype="multipart/form-data">

<input type="file" name="img"/><br/>

<br/>

<input type="submit" name="btn"/>

</form>

</body>

</html>

Prediction.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

</head>

<style>

body{

text-align: center;

background-color: #191919;

color: white;

font-family: monospace;

text-align: center;

font-family: monospace;

top: 50%;

left: 50%;

position: absolute;

transform: translate(-50%,-50%);

}

h1{

font-size: 35px;

}

</style>

<body>

<h2>Prediction</h2>

<h1>{{data}}</h1>

<br/><br/><br/>

<a href="/">Next Prediction</a>

</body>

</html>

**6.** **SYSTEM TESTING**

**6.1 OVERVIEW OF TESTING**

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property function as a unit. The test data should be chosen such that it passed through all possible conditions Actually testing is the state of implementation which aimed at ensuring that the system works accurately and efficiently before the actual operation commence. The following is the description of the testing strategies, which were carried out during the testing period.

**6.2 TYPES OF TESTS**

**6.2.1 UNIT TESTING**

Unit testing is a level of software testing where individual units/ components of the software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output.

**6.2.2 INTEGRATION TESTING**

Integration testing is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing

**6.2.3 ACCEPTANCE TESTING**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**7. RESULTS**

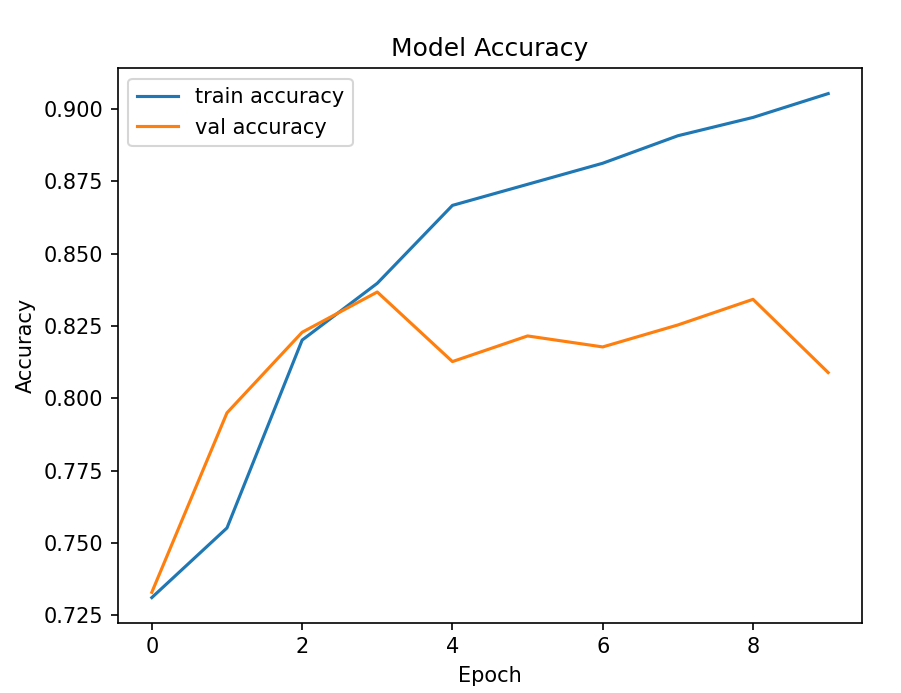


Fig. 3. Visualization of model accuracy for text dataset

We consider x-axis as epoch values and y-axis as accuracy values. Here orange line indicates the accuracy of the validation dataset and blue line indicates the accuracy of the training dataset. The accuracy of training dataset of text is 0.9072 and accuracy of validation dataset is 0.8316.

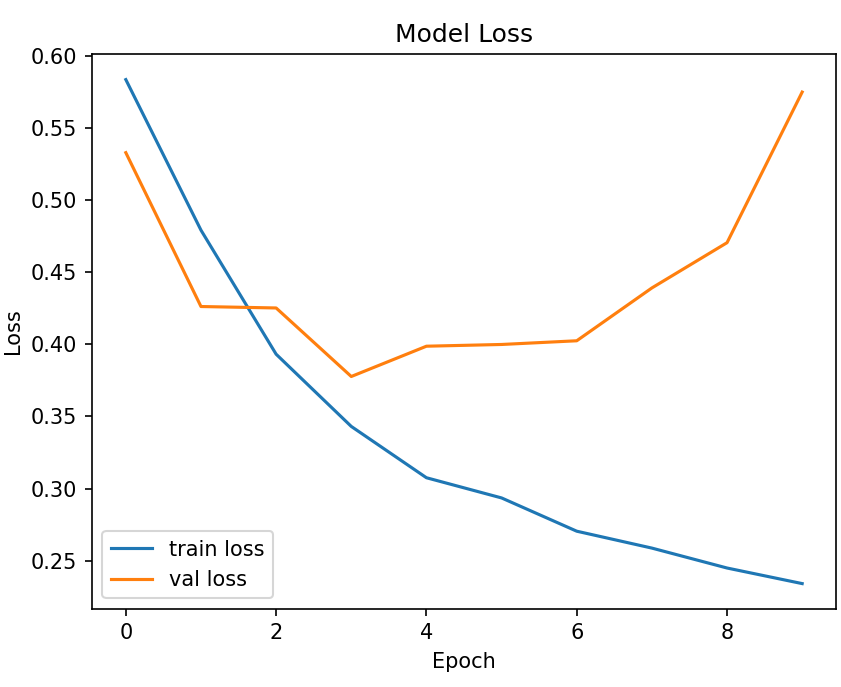


Fig. 4. Visualization of model loss for text dataset

We consider the x-axis as epoch values the and y-axis as loss values. Here orange line indicates the loss of the validation dataset and the blue line indicates the loss of the training dataset. The loss of the training dataset of text is 0.2342 and the loss of the validation dataset is 0.5748.

|  |  |  |
| --- | --- | --- |
| TEXT DATASET | ACCURACY | LOSS |
| TRAIN DATASET | 0.9072 | 0.2342 |
| VAL\_DATASET | 0.8316 | 0.5748 |

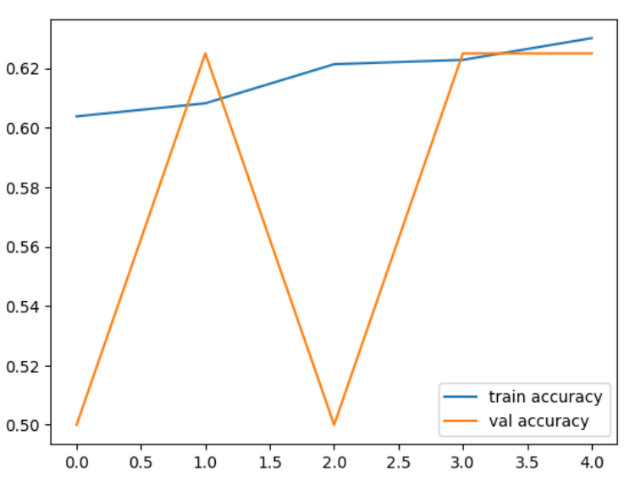


Fig. 5. Visualization of model accuracy for image dataset

We consider x-axis as epoch values and y-axis as accuracy values. Here orange line indicates the accuracy of the validation dataset and blue line indicates the accuracy of the training dataset. The accuracy of training dataset of images is 0.6314 and accuracy of validation dataset is 0.6109.

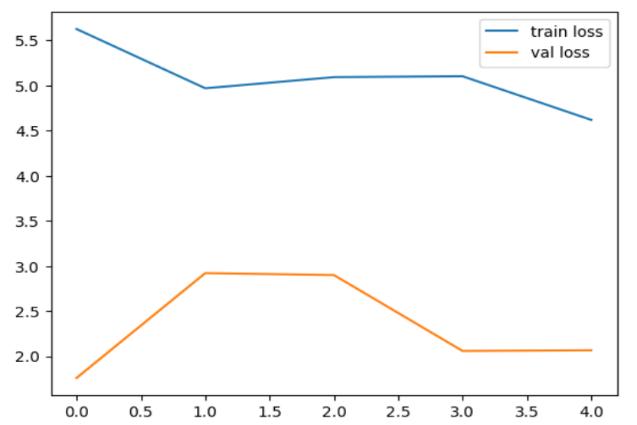
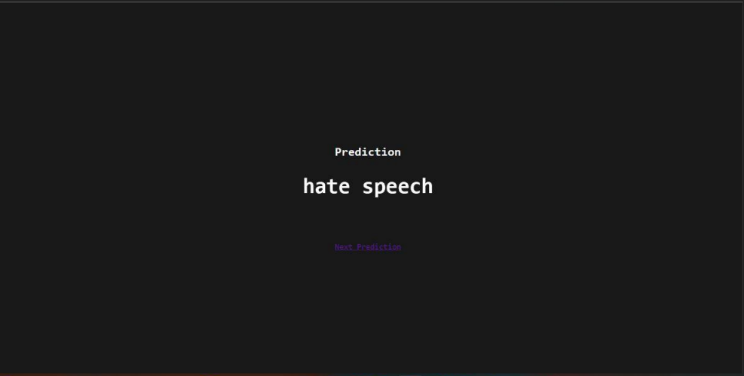


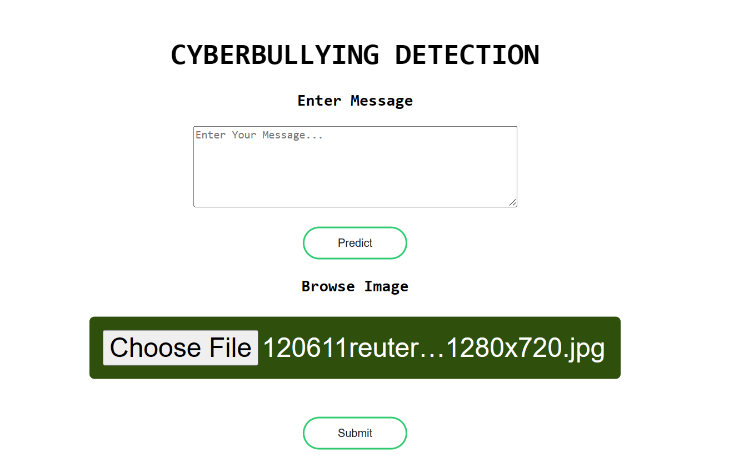
Fig. 6. Visualization of model loss for the image dataset

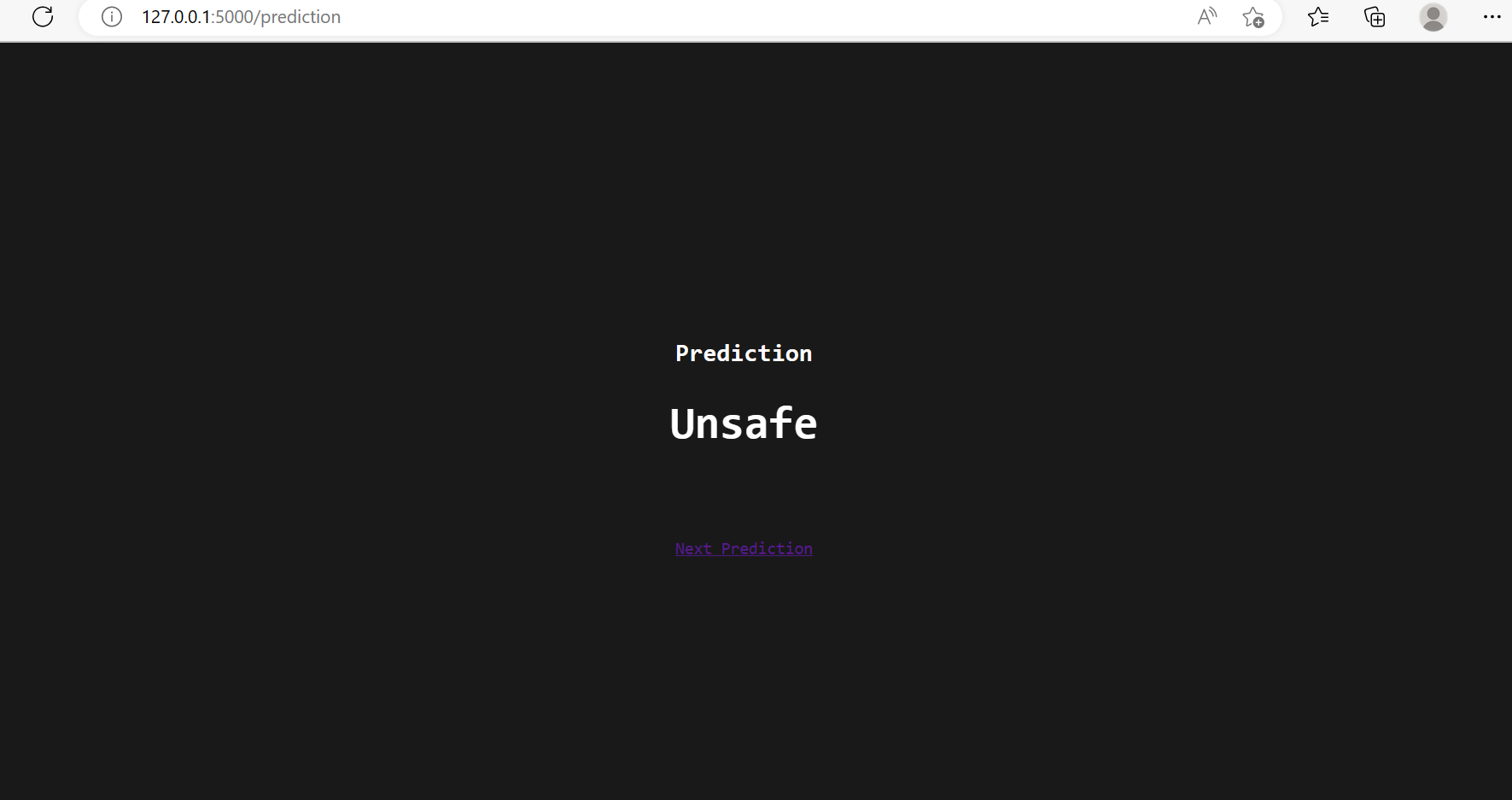
We consider the x-axis as epoch values and the y-axis as loss values. Here orange line indicates the loss of the validation dataset and the blue line indicates the loss of the training dataset. The loss of the training dataset of text is 2.342 and the loss of the validation dataset is 4.748

|  |  |  |
| --- | --- | --- |
| IMAGE DATASET | ACCURACY | LOSS |
| TRAIN DATASET | 0.6314 | 2.342 |
| VAL\_DATASET | 0.6109 | 4.748 |









**8. CONCLUSION**

Cyberbullying is the sort of abusing or bullying that is well-known these days. Nowadays online platforms are used for communicating between people and they have become one of the platforms for bullying mainly in this era of technology. We have to eradicate it as it affects many users even the younger generation. The trouble of cyberbullying has been worst as it results in a rapid increment of sadness and homicides. We apply machine learning approaches to predict cyberbullying. The current study’s results will help advance the field and provide light on the problem of finding aggressive human activities, namely cyberbullying on online social networking sites.

**9. REFERENCES**

1. Md Manowarul Islam, Md Ashraf Uddin, Linta Islam, Arnisha Akter “Cyberbullying Detection on Social Networks Using Machine Learning Approaches” IEEE,2020,  pp. 1-6, doi: 10.1109/CSDE50874.2020.9411601.
2. Pradeep Kumar, Fenish, Umeshbhai “Cyberbullying detection using deep transfer learning” Springer, 2022,doi:10.1007/s40747-022-00772-z.
3. T. A. Buan and R. Ramachandra, “Automated cyberbullying detection in social media using an svm activated stacked convolution lstm

network,” in Proceedings of the 2020 the 4th International Conference on Compute and Data Analysis, 2020, DOI:10.1109/CSDE50874.2020.9411601.

1. R. R. Dalvi, S. Baliram Chavan and A. Halbe, Detecting A Twitter Cyberbullying Using Machine Learning, ICICCS,2020 DOI:10.1109/ICICCS48265.2020.9120893.
2. Amrita Mangaonkar, Rohit Pawar, Nahida Sultana Chowdhury, Rajeev R. Raje, Enhancing collaborative detection of cyberbullying behavior in Twitter data, Springer, 2022, pp 1263–1277,DOI:10.1007/s10586-021-03483-1.
3. Shardul Suryawanshi, Chakravarthi, Mihael Arcan, Paul Buitelaar “Multimodal Meme Dataset (MultiOFF) for Identifying Offensive Content in Image and Text”, pp 32–41, France. European Language Resources Association (ELRA).
4. Dr. Vijayakumar V, Dr. Hari Prasad D, Adolf P “Multimodal Cyberbullying Detection using Hybrid Deep Learning Algorithms” International Journal of Applied Engineering Research,2022 pp. 568-57, DOI:10.37622/IJAER/16.7.2021.568-574.
5. Theyazn H. H. Aldhyani, Mosleh Hmoud Al-Adhaileh and Saleh Nagi Alsubari “Cyberbullying Identification System Based Deep Learning Algorithms”, Electronics, 2022, DOI: 10.3390/electronics11203273.
6. Agrawal, S., Awekar, A.: Deep Learning for Detecting Cyberbullying Across Multiple Social Media Platforms, European Conference on Information,2018, Retrieval, pp. 141–153, DOI: 10.48550/arXiv.1801.06482.

Sinchana, Pradyumna, Janhavi, Deepika “Detection of Cyberbullying using Machine Learning, Turkish Journal of Computer and Mathematics