A

Major Project

On

DEA-RNN A HYBRID DEEP LEARNING APPROACH FOR CYBERBULLYING DETECTION IN TWITTER SOCIAL MEDIA PLATFORM

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

BATHULA PRAVALIKA (207R1A05K0)

MOHAMMED MUBEEN (207R1A05M3)

Under the Guidance of

Dr. J. NARASIMHARAO

(Associate Professor)

S

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

2020-2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled **"DEA-RNN A HYBRID DEEP LEARNING APPROACH FOR CYBERBULLYING DETECTION IN TWITTER SOCIAL MEDIA PLATFORM"** being submitted by **BATHULA PRAVALIKA (207R1A05K0) , MOHAMMED MUBEEN (207R1A05M3)**in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2023-24.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Dr. J. Narasimharao
(Associate Professor)
INTERNAL GUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. K. Srujan Raju
HOD

EXTERNAL EXAMINER

Submitted for viva voice Examination held on ________________________

ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard tomy guide **Dr. J. Narasimharao,** Associate Professor for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) **G. Vinesh Shanker, Dr. J. Narasimharao, Ms. Shilpa, & Dr. K. Maheswari** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

BATHULA PRAVALIKA (207R1A05K0)

MOHAMMED MUBEEN (207R1A05M3)

ABSTRACT

DEA-RNN, a pioneering solution in the realm of cyberbullying detection, represents a significant breakthrough by seamlessly integrating Elman-type Recurrent Neural Networks with the innovative Dolphin Echolocation Algorithm (DEA). This unique fusion not only accelerates the model's learning process but also greatly enhances its proficiency in identifying instances of cyberbullying on the Twitter platform. A comprehensive evaluation was conducted on a substantial dataset comprising 10,000 tweets, reaffirming DEA-RNN's excellence in the field. It consistently outperformed established algorithms, boasting an impressive average accuracy of 90.45%, a precision rate of 89.52%, a recall rate of 88.98%, an F1-score of 89.25%, and a remarkable specificity rate of 90.94%. These remarkable findings underscore DEARNN's potential as a powerful tool for enhancing online safety and providing users with an effective means to combat cyberbullying on one of the most prominent social media platforms. The model's consistent superiority over its peers reflects its promise as a crucial asset in the ongoing effort to create a more secure and respectful online environment.

LIST OF FIGURES/TABLES

LIST OF SCREENSHOTS

TABLE OF CONTENTS

1. INTRODUCTION

1. INTRODUCTION

1.1 PROJECT SCOPE

The DEA-RNN project aims to develop a hybrid deep learning model for cyberbullying detection on the Twitter social media platform. It focuses on enhancing the accuracy and efficiency of identifying instances of cyberbullying within a large dataset. The project encompasses the integration of Elman-type Recurrent Neural Networks and the Dolphin Echolocation Algorithm for improved performance. The scope includes rigorous evaluation and benchmarking against existing algorithms to assess its effectiveness. Ultimately, DEA-RNN aims to contribute to a safer and more respectful online environment on Twitter.

1.2 PROJECT PURPOSE

The purpose of the DEA-RNN project is to develop an advanced cyberbullying detection model for Twitter. It aims to enhance online safety by accurately identifying instances of cyberbullying on the platform. The project seeks to leverage deep learning techniques, specifically integrating Elman-type Recurrent Neural Networks and the Dolphin Echolocation Algorithm, to improve detection capabilities. Its core objective is to contribute to a more secure and respectful online environment for Twitter users.

1.3 PROJECT FEATURES

The key features of the DEA-RNN project include the integration of Elman-type Recurrent Neural Networks with the innovative Dolphin Echolocation Algorithm for improved cyberbullying detection. It offers robust performance evaluation against existing algorithms to showcase its effectiveness. The project emphasizes high accuracy, precision, recall, and specificity rates, ensuring reliable identification of cyberbullying instances on Twitter. Additionally, it aims to enhance online safety and promote a more respectful digital environment.

2. SYSTEM ANALYSIS

2. SYSTEM ANALYSIS

SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst hasa firm understanding of what is to be done.

2.1 PROBLEM DEFINITION

DEA-RNN is a cyberbullying detection project on Twitter that combines Elman-type Recurrent Neural Networks and the Dolphin Echolocation Algorithm for improved accuracy. It aims to provide a safer online environment by effectively identifying instances of cyberbullying, contributing to a culture of respect and safety on the platform.

2.2 EXISTING SYSTEM

Numerous research studies have explored the application of machine learning techniques to identify cyberbullying in tweets They employed classifiers like Support Vector Machine (SVM), Random Forests (RF), Naïve Bayes (NB), and others. These methods often relied on features extracted from the text, social characteristics, and network data. While some showed high performance, they had limitations such as model complexity, low accuracy, and challenges in handling sarcasm and multiple- meaning terms The studies also highlighted that the emotional content of tweets didn't significantly impact detection. These methods provide valuable insights into identifying and addressing cyberbullying on social media, but there is room for improvement in terms of accuracy and handling nuanced language.

2.2.1 DISADVANTAGES OF EXISTING SYSTEM

Following are the disadvantages of existing system:

- **Effective ML Classifiers**: The system suffers from the absence of efficient machine learning classifiers, hindering its ability to detect cyberbullying effectively.
- **Limited Predictive Capability**:The non-implementation of DEA-RNN techniques has led to reduced predictive power, resulting in lower accuracy in cyberbullying detection

2.3 PROPOSED SYSTEM

In the proposed system, we introduce DEA-RNN, a powerful hybrid deep learningbased system designed to automatically detect cyberbullying in tweets. DEA-RNN combines Elman-type Recurrent Neural Networks (RNN) with an enhanced Dolphin Echolocation Algorithm (DEA) to fine-tune RNN parameters, making it adept at handling the dynamic nature of short texts and extracting trending topics. In rigorous evaluations, DEA-RNN consistently outperforms existing approaches in cyberbullying detection on Twitter, achieving superior results in terms of recall, precision, accuracy, F1 score, and specificity. This system's key contributions include an optimized DEA model for parameter tuning, the integration of Elman-type RNN with improved DEA for tweet classification, a new Twitter dataset for evaluation, and an efficient approach for recognizing and classifying cyberbullying tweets.

2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

Following are the disadvantages of existing system:

- **Efficient Trending Topic Extraction:** The system effectively identifies and extracts trending topics from tweets, aiding in focusing on significant discussions.
- **Effective Cyberbullying Identification:** Through rigorous training and testing with classifiers like SVM, Multinomial Naive Bayes and Random Forests,the system excels in identifying instances of cyberbullying.

2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and a business proposalis put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This isto ensure that the proposed system is not a burden to the company. Three key considerations involved in the feasibility analysis:

- EconomicFeasibility
- TechnicalFeasibility
- SocialFeasibility

2.4.1 ECONOMIC FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

2.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.4.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specify the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

- Processor : Pentium –IV
- RAM : 4 GB (min)
- Hard Disk : 20 GB
- Key Board : Standard Windows Keyboard
- Mouse : Two or Three Button Mouse
- Monitor : SVGA

2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements.

- Operating System : Windows 7 Ultimate
- Coding Language : Python
- Front-End : HTML, CSS, JavaScript
- Back-End : Django-ORM
- Database : MySQL (WAMP Server)

3. ARCHITECTURE

3. ARCHITECTURE

3.1 PROJECT ARCHITECTURE

This project architecture shows the procedure followed for classification,starting from input to final prediction.

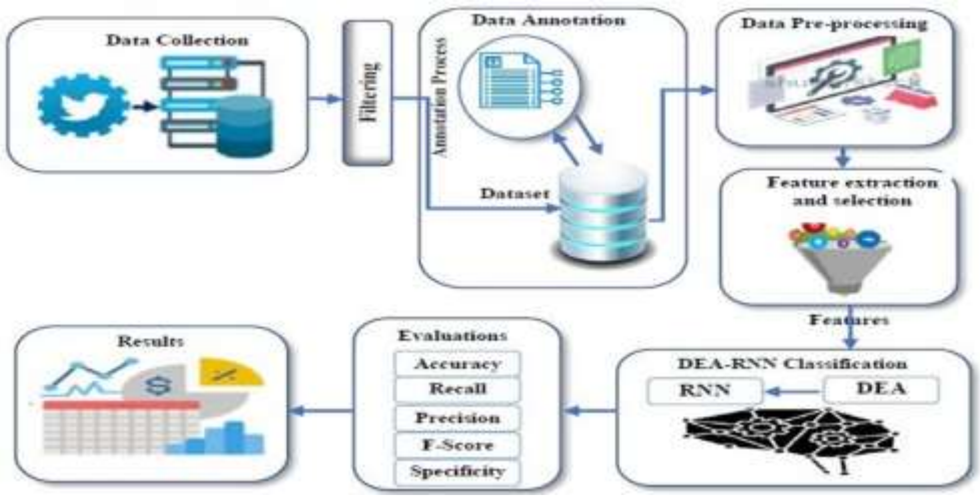


Figure 3.1: Project Architecture of Dea-Rnn A Hybrid Deep Learning Approach For Cyberbullying Detection In Twitter Social Media Platform

3.2 DESCRIPTION

The project aims to improve cyberbullying detection on Twitter by employing machine learning techniques. It explores the integration of advanced classifiers to enhance detection accuracy. The project also investigates the impact of emotional content and the handling of nuanced language in tweets. It seeks to address limitations in existing methods and contribute to more effective and precise cyberbullying identification on the platform.

3.3 USE CASE DIAGRAM

In the use case diagram, we have basically one actor who is the user in the trained model. A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of usersthe system has. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.

- **Use case**

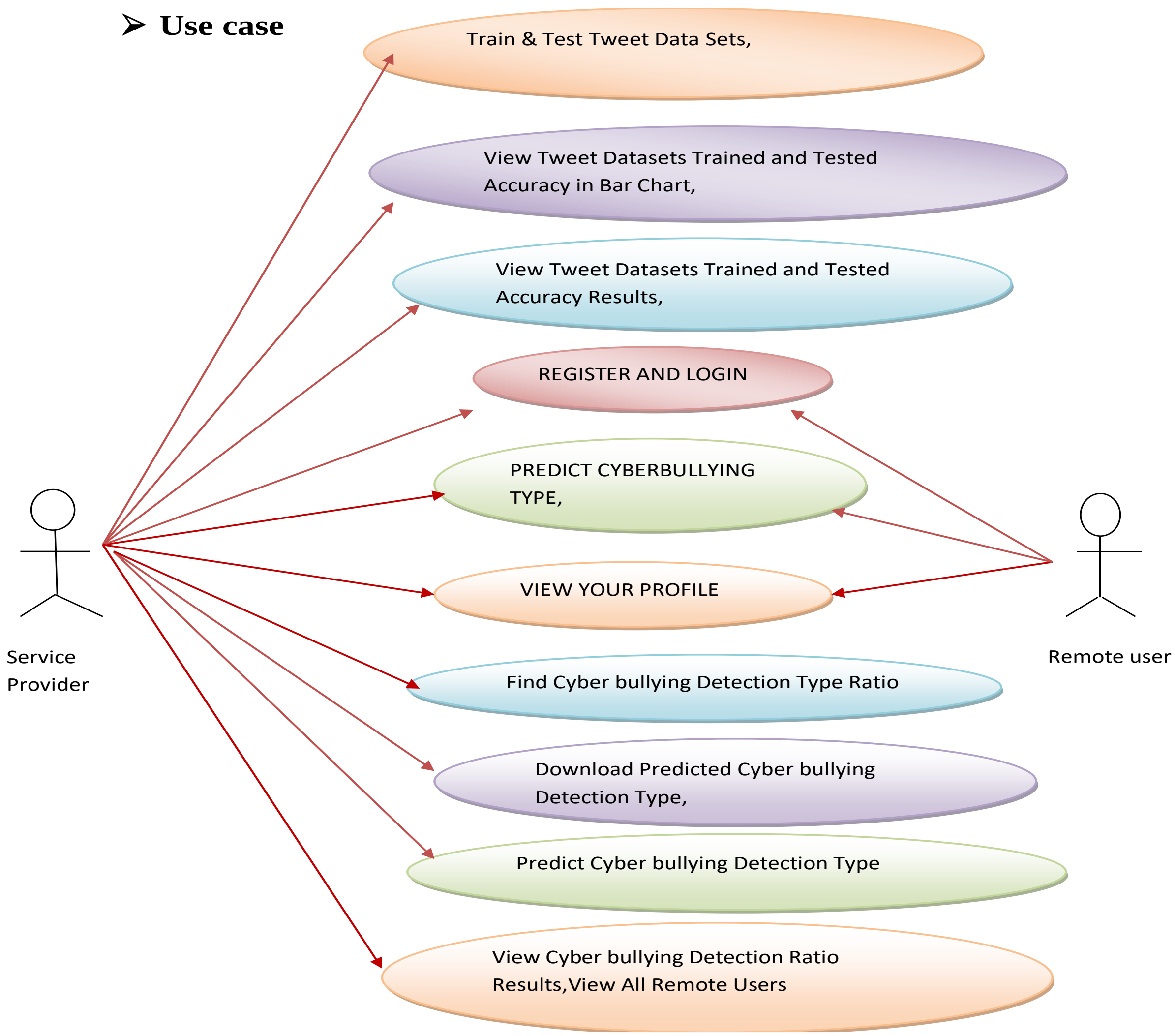


Figure 3.2: Use Case Diagram Of Dea-Rnn A Hybrid Deep Learning Approach For Cyberbullying Detection In Twitter Social Media Platform

3.4 CLASS DIAGRAM

Class diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations(or methods), and the relationships among objects.

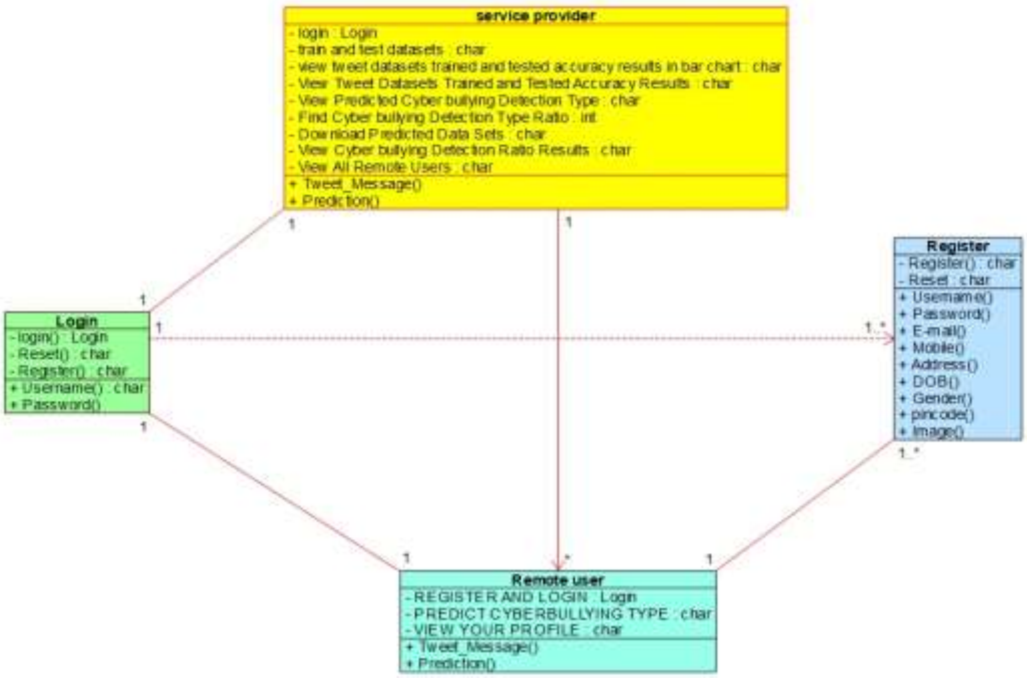


Figure 3.3: Class Diagram of Dea-Rnn A Hybrid Deep Learning Approach For Cyberbullying Detection In Twitter Social Media Platform

3.5 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development.

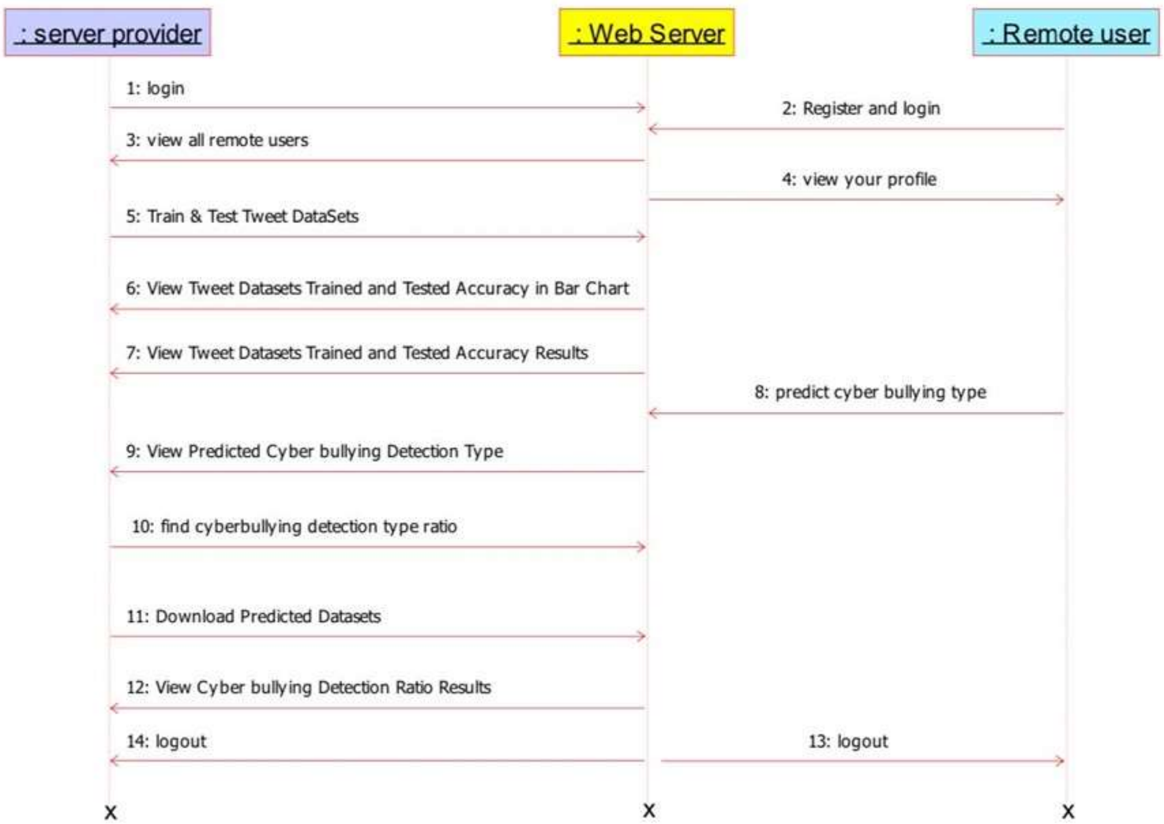


Figure 3.4: Class Diagram of Dea-Rnn A Hybrid Deep Learning Approach For Cyberbullying Detection in Twitter Social Media Platform

3.6 FLOW CHART DIAGRAM

A flow chart diagram is a visual representation of a process or system. It uses symbols and arrows to illustrate the sequence of steps or decisions within the process. Flow charts aid in understanding, analyzing, and improving processes in various fields such as business, engineering, and software development.

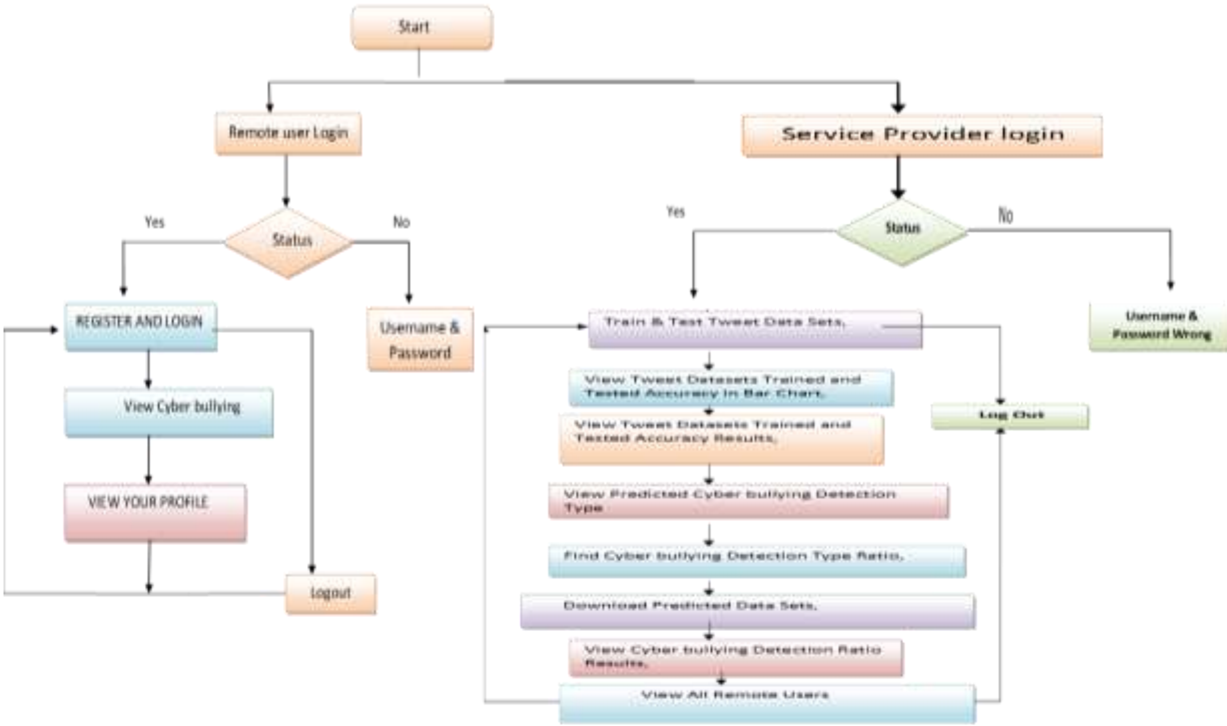


Figure 3.5: Flow Chart Diagram of Dea-Rnn A Hybrid Deep Learning Approach For Cyberbullying Detection in Twitter Social Media Platform

4.IMPLEMENTATION

4.1 SAMPLE CODE

```
from django.db.models import Count
from django.db.models import Q
from django.shortcuts import render, redirect, get_object_or_404
import datetime
import openpyxl
import nltk
import re
import string
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import CountVectorizer
from nltk.stem.wordnet import WordNetLemmatizer
import pandas as pd
from wordcloud import WordCloud, STOPWORDS
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import VotingClassifier
# Create your views here.
from Remote_User.models import
ClientRegister_Model,Cyberbullying_Detection_Type,detection_ratio,detection_accuracy

def login(request):
    if request.method == "POST" and 'submit1' in request.POST:
        username = request.POST.get('username')
        password = request.POST.get('password')
        try:
            enter = ClientRegister_Model.objects.get(username=username,password=password)
```

```
          request.session["userid"] = enter.id
          return redirect('ViewYourProfile')
      except:
          pass
  return render(request,'RUser/login.html')
def Register1(request):
  if request.method == "POST":
      username = request.POST.get('username')
      email = request.POST.get('email')
      password = request.POST.get('password')
      phoneno = request.POST.get('phoneno')
      country = request.POST.get('country')
      state = request.POST.get('state')
      city = request.POST.get('city')
      address = request.POST.get('address')
      gender = request.POST.get('gender')
      ClientRegister_Model.objects.create(username=username, email=email,
password=password, phoneno=phoneno,
                                          country=country, state=state, city=city, address=address,
gender=gender)
      obj = "Registered Successfully"
      return render(request, 'RUser/Register1.html', {'object': obj})
  else:
      return render(request,'RUser/Register1.html')
def ViewYourProfile(request):
  userid = request.session['userid']
  obj = ClientRegister_Model.objects.get(id= userid)
  return render(request,'RUser/ViewYourProfile.html',{'object':obj}
def Predict_Tweet_Meesage_Type(request):
  if request.method == "POST":
      if request.method == "POST":
          tweettext = request.POST.get('tweettext')
```

```
data = pd.read_csv("Datasets.csv",encoding='latin-1')
def clean_text(text):
    text = text.lower()
    text = re.sub('\[.*?\]', '', text)
    text = re.sub('https?://\S+|www\.\S+', '', text)
    text = re.sub('<.*?>+', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\n', '', text)
    text = re.sub('\w*\d\w*', '', text)
    return text
    data['text'] = data['tweet_text'].apply(lambda x: clean_text(x))
def apply_results(results):
    if (results == "not_cyberbullying"):
        return 0
    elif (results == "gender"):
        return 1
    elif (results == "religion"):
        return 2
    elif (results == "other_cyberbullying"):
        return 3
    elif (results == "age"):
        return 4
    elif (results == "ethnicity"):
        return 5
data['Results'] = data['cyberbullying_type'].apply(apply_results)
x = data['tweet_text']
y = data['Results']
cv = CountVectorizer(lowercase=False, strip_accents='unicode', ngram_range=(1, 1))
x = cv.fit_transform(x)
models = []
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.20)
```

```
X_train.shape, X_test.shape, y_train.shape
print("Multinomial Naive Bayes")
from sklearn.naive_bayes import MultinomialNB
nb_clf = MultinomialNB()
nb_clf.fit(X_train, y_train)
MultinomialNB()
nb_pred = nb_clf.predict(X_test)
mnb = accuracy_score(y_test, nb_pred) * 100
print(mnb)
print(confusion_matrix(y_test, nb_pred))
print(classification_report(y_test, nb_pred))
models.append(('nb_pred', nb_clf))
# SVM Model
print("SVM")
from sklearn import svm
lin_clf = svm.LinearSVC()
lin_clf.fit(X_train, y_train)
predict_svm = lin_clf.predict(X_test)
svm_acc = accuracy_score(y_test, predict_svm) * 100
print(svm_acc)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, predict_svm))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, predict_svm))
models.append(('svm', lin_clf))
print("Logistic Regression")
from sklearn.linear_model import LogisticRegression
reg = LogisticRegression(random_state=0, solver='lbfgs').fit(X_train, y_train)
y_pred = reg.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, y_pred) * 100)
print("CLASSIFICATION REPORT")
```

```
print(classification_report(y_test, y_pred))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, y_pred))
models.append(('logistic', reg))
print("Decision Tree Classifier")
dtc = DecisionTreeClassifier()
dtc.fit(X_train, y_train)
dtcpredict = dtc.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, dtcpredict) * 100)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, dtcpredict))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, dtcpredict))
models.append(('DecisionTreeClassifier', dtc))
classifier = VotingClassifier(models)
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)
tweettext1 = [tweettext]
vector1 = cv.transform(tweettext1).toarray()
predict_text = classifier.predict(vector1)
pred = str(predict_text).replace("[", "")
pred1 = pred.replace("]", "")
prediction = int(pred1)
if prediction == 0:
    val='not_cyberbullying'
elif prediction == 1:
    val= 'gender'
elif prediction == 2:
    val = 'religion'
elif prediction == 3:
    val = 'other_cyberbullying'
```

```
        elif prediction == 4:
            val = 'age'
        elif prediction == 5:
            val = 'ethnicity'
        print(prediction)
        print(val)
        Cyberbullying_Detection_Type.objects.create(Tweet_Message=tweettext,Prediction=val)
        return render(request, 'RUser/Predict_Tweet_Meesage_Type.html',{'objs': val})
    return render(request, 'RUser/Predict_Tweet_Meesage_Type.html')
from django.db.models import  Count, Avg
from django.shortcuts import render, redirect
from django.db.models import Count
from django.db.models import Q
import datetime
import xlwt
from django.http import HttpResponse
import re
import string
import pandas as pd
from wordcloud import WordCloud, STOPWORDS
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import VotingClassifier
# Create your views here.
from Remote_User.models import
ClientRegister_Model,Cyberbullying_Detection_Type,detection_ratio,detection_accuracy
def serviceproviderlogin(request):
    if request.method  == "POST":
        admin = request.POST.get('username')
```

```
        password = request.POST.get('password')
        if admin == "Admin" and password =="Admin":
            return redirect('View_Remote_Users')
    return render(request,'SProvider/serviceproviderlogin.html')
def Find_Predicted_Cyberbullying_Detection_Ratio(request):
    detection_ratio.objects.all().delete()
    ratio = ""
    kword = 'not_cyberbullying'
    print(kword)
    obj = Cyberbullying_Detection_Type.objects.all().filter(Q(Prediction=kword))
    obj1 = Cyberbullying_Detection_Type.objects.all()
    count = obj.count();
    count1 = obj1.count();
    ratio = (count / count1) * 100
    if ratio != 0:
        detection_ratio.objects.create(names=kword, ratio=ratio)
    ratio1 = ""
    kword1 = 'gender'
    print(kword1)
    obj1 = Cyberbullying_Detection_Type.objects.all().filter(Q(Prediction=kword1))
    obj11 = Cyberbullying_Detection_Type.objects.all()
    count1 = obj1.count();
    count11 = obj11.count();
    ratio1 = (count1 / count11) * 100
    if ratio1 != 0:
        detection_ratio.objects.create(names=kword1, ratio=ratio1)
    ratio12 = ""
    kword12 = 'religion'
    print(kword12)
    obj12 = Cyberbullying_Detection_Type.objects.all().filter(Q(Prediction=kword12))
    obj112 = Cyberbullying_Detection_Type.objects.all()
    count12 = obj12.count();
```

```
count112 = obj112.count();
ratio12 = (count12 / count112) * 100
if ratio12 != 0:
    detection_ratio.objects.create(names=kword12, ratio=ratio12)
ratio123 = ""
kword123 = 'other_cyberbullying'
print(kword123)
obj123 = Cyberbullying_Detection_Type.objects.all().filter(Q(Prediction=kword123))
obj1123 = Cyberbullying_Detection_Type.objects.all()
count123 = obj123.count();
count1123 = obj1123.count();
ratio123 = (count123 / count1123) * 100
if ratio123 != 0:
    detection_ratio.objects.create(names=kword123, ratio=ratio123)
ratio1234 = ""
kword1234 = 'age'
print(kword1234)
obj1234 = Cyberbullying_Detection_Type.objects.all().filter(Q(Prediction=kword1234))
obj11234 = Cyberbullying_Detection_Type.objects.all()
count1234 = obj1234.count();
count11234 = obj11234.count();
ratio1234 = (count1234 / count11234) * 100
if ratio1234 != 0:
    detection_ratio.objects.create(names=kword1234, ratio=ratio1234)
ratio123491 = ""
kword123491 = 'ethnicity'
print(kword123491)
obj123491 = Cyberbullying_Detection_Type.objects.all().filter(Q(Prediction=kword123491))
obj1123491 = Cyberbullying_Detection_Type.objects.all()
count123491 = obj123491.count();
count1123491 = obj1123491.count();
ratio123491 = (count123491 / count1123491) * 100
```

```
    if ratio123491 != 0:
        detection_ratio.objects.create(names=kword123491, ratio=ratio123491)
    obj = detection_ratio.objects.all()
    return render(request, 'SProvider/Find_Predicted_Cyberbullying_Detection_Ratio.html',
{'objs': obj})
def View_Remote_Users(request):
    obj=ClientRegister_Model.objects.all()
    return render(request,'SProvider/View_Remote_Users.html',{'objects':obj})
def ViewTrendings(request):
    topic =
Cyberbullying_Detection_Type.objects.values('topics').annotate(dcount=Count('topics')).order_b
y('-dcount')
    return  render(request,'SProvider/ViewTrendings.html',{'objects':topic})
def charts(request,chart_type):
    chart1 = detection_ratio.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/charts.html", {'form':chart1, 'chart_type':chart_type})
def charts1(request,chart_type):
    chart1 = detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/charts1.html", {'form':chart1, 'chart_type':chart_type})
def View_Predicted_Cyberbullying_Detection_Type(request):
    obj =Cyberbullying_Detection_Type.objects.all()
    return render(request, 'SProvider/View_Predicted_Cyberbullying_Detection_Type.html',
{'list_objects': obj})
def likeschart(request,like_chart):
    charts =detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/likeschart.html", {'form':charts, 'like_chart':like_chart})
def Download_Predicted_DataSets(request):
    response = HttpResponse(content_type='application/ms-excel')
    # decide file name
    response['Content-Disposition'] = 'attachment; filename="Predicted_Data.xls"'
    # creating workbook
    wb = xlwt.Workbook(encoding='utf-8')
```

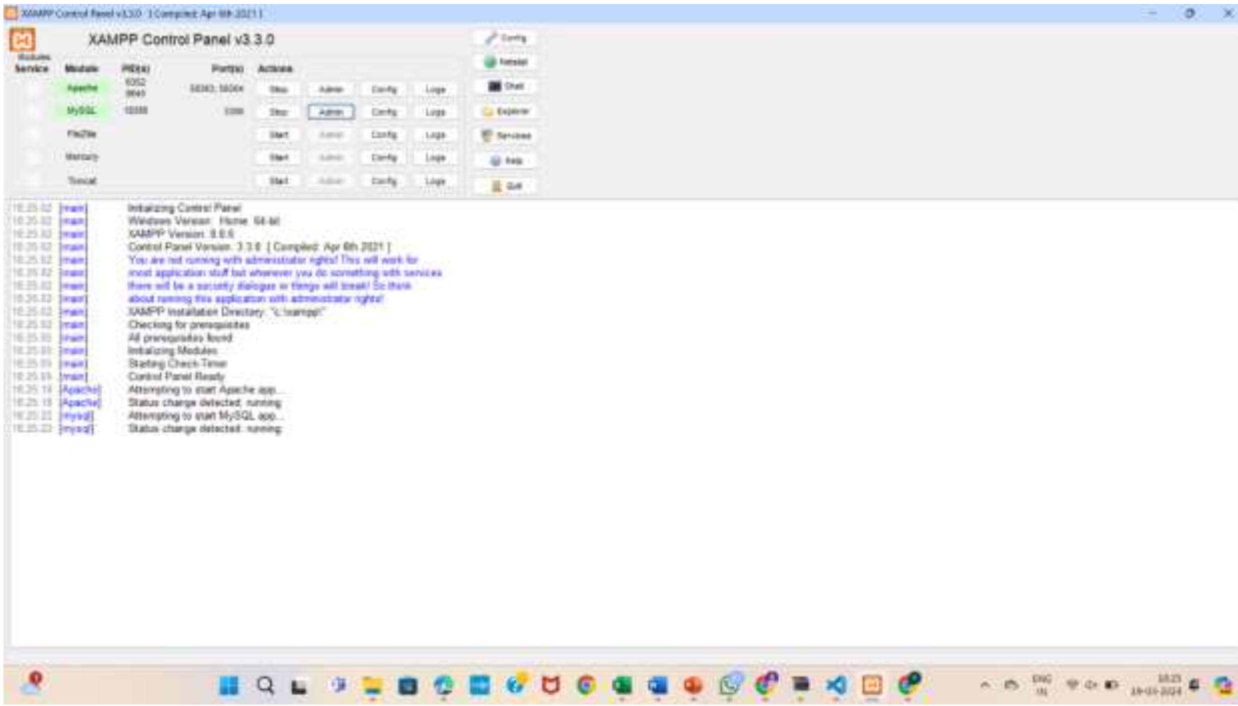
```
    # adding sheet
    ws = wb.add_sheet("sheet1")
    # Sheet header, first row
    row_num = 0
    font_style = xlwt.XFStyle()
    # headers are bold
    font_style.font.bold = True
    # writer = csv.writer(response)
    obj = Cyberbullying_Detection_Type.objects.all()
    data = obj  # dummy method to fetch data.
    for my_row in data:
        row_num = row_num + 1
        ws.write(row_num, 0, my_row.Tweet_Message, font_style)
        ws.write(row_num, 1, my_row.Prediction, font_style)
    wb.save(response)
    return response
def Train_Test_DataSets(request):
    detection_accuracy.objects.all().delete()
    data = pd.read_csv("Datasets.csv",encoding='latin-1')
    def clean_text(text):
        text = text.lower()
        text = re.sub('\[.*?\]', '', text)
        text = re.sub('https?://\S+|www\.\S+', '', text)
        text = re.sub('<.*?>+', '', text)
        text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
        text = re.sub('\n', '', text)
        text = re.sub('\w*\d\w*', '', text)
        return text
        data['text'] = data['tweet_text'].apply(lambda x: clean_text(x))
    def apply_results(results):
        if (results == "not_cyberbullying"):
            return 0
```

```
    elif (results == "gender"):
        return 1
    elif (results == "religion"):
        return 2
    elif (results == "other_cyberbullying"):
        return 3
    elif (results == "age"):
        return 4
    elif (results == "ethnicity"):
        return 5
data['Results'] = data['cyberbullying_type'].apply(apply_results)
x = data['tweet_text']
y = data['Results']
cv = CountVectorizer(lowercase=False, strip_accents='unicode', ngram_range=(1, 1))
x = cv.fit_transform(x)
models = []
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.20)
X_train.shape, X_test.shape, y_train.shape
print("Multinomial Naive Bayes")
from sklearn.naive_bayes import MultinomialNB
nb_clf = MultinomialNB()
nb_clf.fit(X_train, y_train)
MultinomialNB()
nb_pred = nb_clf.predict(X_test)
mnb = accuracy_score(y_test, nb_pred) * 100
print(mnb)
print(confusion_matrix(y_test, nb_pred))
print(classification_report(y_test, nb_pred))
models.append(('nb_pred', nb_clf))
detection_accuracy.objects.create(names="MultinomialNB", ratio=mnb)
# SVM Model
```

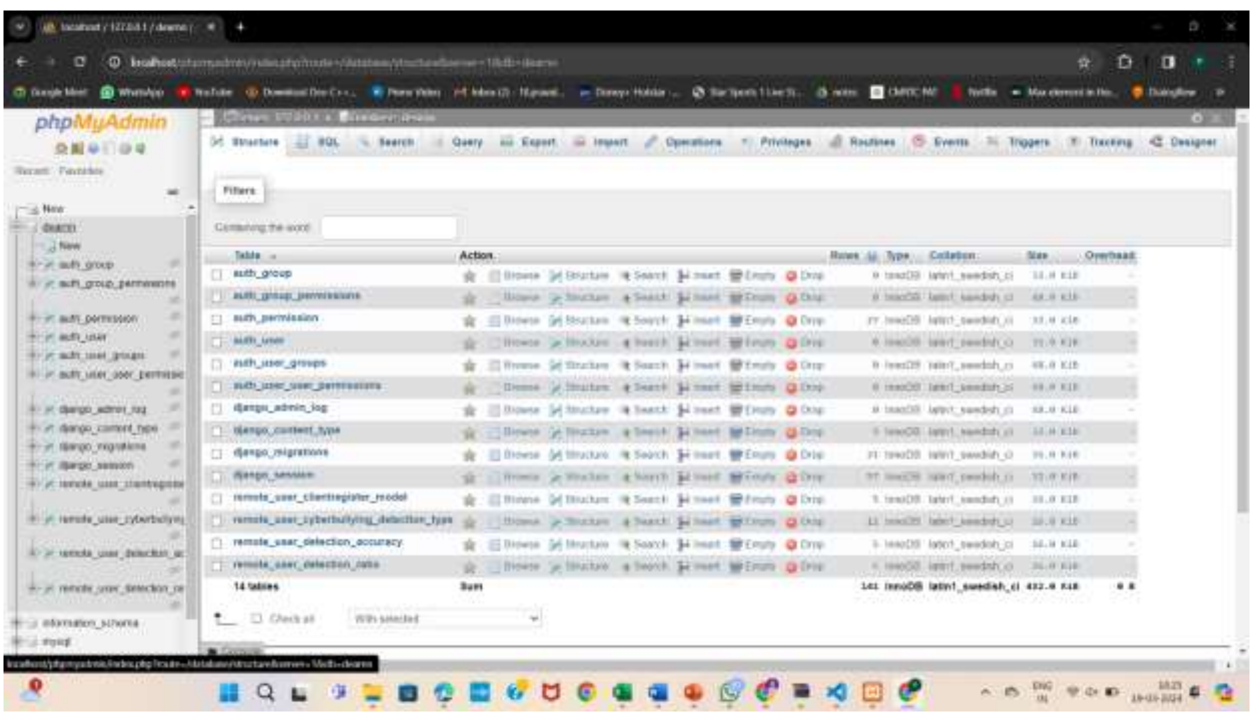
```
print("SVM")
from sklearn import svm
lin_clf = svm.LinearSVC()
lin_clf.fit(X_train, y_train)
predict_svm = lin_clf.predict(X_test)
svm_acc = accuracy_score(y_test, predict_svm) * 100
print(svm_acc)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, predict_svm))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, predict_svm))
models.append(('svm', lin_clf))
detection_accuracy.objects.create(names="SVM", ratio=svm_acc)
print("Logistic Regression")
from sklearn.linear_model import LogisticRegression
reg = LogisticRegression(random_state=0, solver='lbfgs').fit(X_train, y_train)
y_pred = reg.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, y_pred) * 100)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, y_pred))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, y_pred))
models.append(('logistic', reg))
detection_accuracy.objects.create(names="Logistic Regression", ratio=accuracy_score(y_test,
y_pred) * 100)
print("Decision Tree Classifier")
dtc = DecisionTreeClassifier()
dtc.fit(X_train, y_train)
dtcpredict = dtc.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, dtcpredict) * 100)
```

```
    print("CLASSIFICATION REPORT")
    print(classification_report(y_test, dtcpredict))
    print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, dtcpredict))
    models.append(('DecisionTreeClassifier', dtc))
    detection_accuracy.objects.create(names="Decision Tree Classifier",
ratio=accuracy_score(y_test, dtcpredict) * 100)
    print("SGD Classifier")
    from sklearn.linear_model import SGDClassifier
    sgd_clf = SGDClassifier(loss='hinge', penalty='l2', random_state=0)
    sgd_clf.fit(X_train, y_train)
    sgdpredict = sgd_clf.predict(X_test)
    print("ACCURACY")
    print(accuracy_score(y_test, sgdpredict) * 100)
    print("CLASSIFICATION REPORT")
    print(classification_report(y_test, sgdpredict))
    print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, sgdpredict))
    models.append(('SGDClassifier', sgd_clf))
    detection_accuracy.objects.create(names="SGD Classifier", ratio=accuracy_score(y_test,
sgdpredict) * 100)
    csv_format = 'Results.csv'
    data.to_csv(csv_format, index=False)
    data.to_markdown
    obj = detection_accuracy.objects.all()
    return render(request,'SProvider/Train_Test_DataSets.html', {'objs': obj})
```

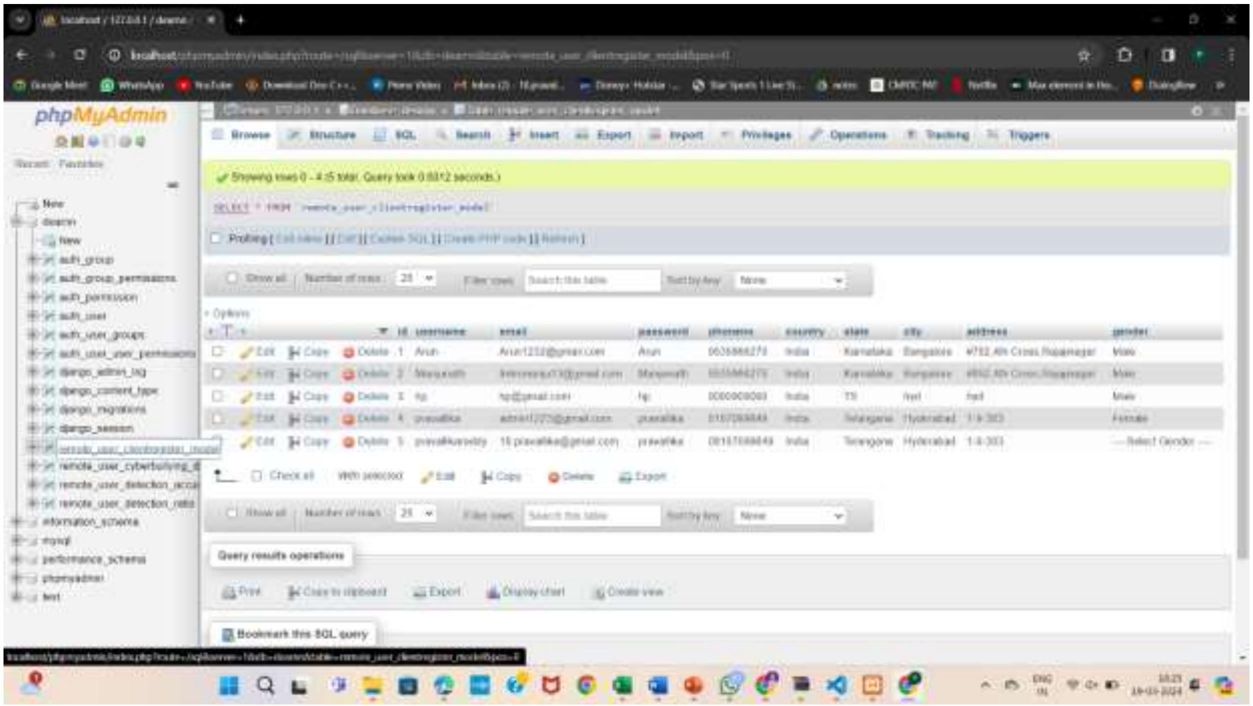
5.SCREENSHOTS



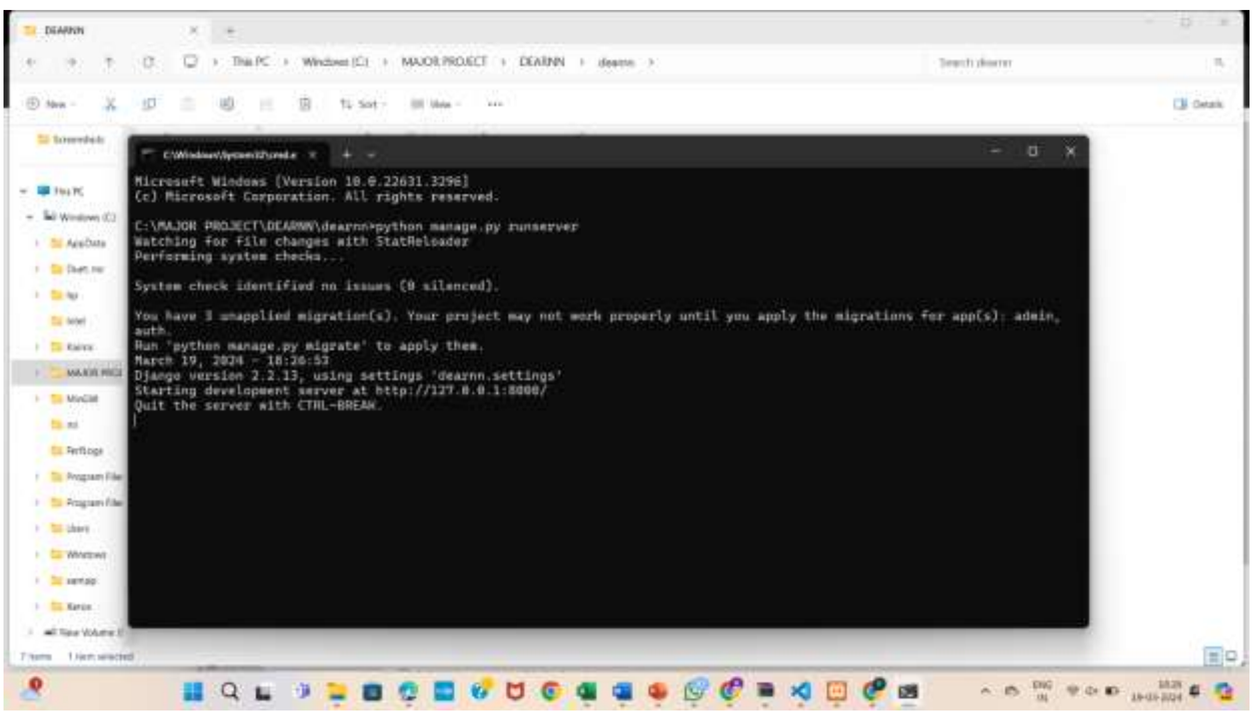
Screenshot 5.1: Open xamp and start apache and mysql-admin



Screenshot 5.2: dearnn contains databases



Screenshot 5.3: Databases

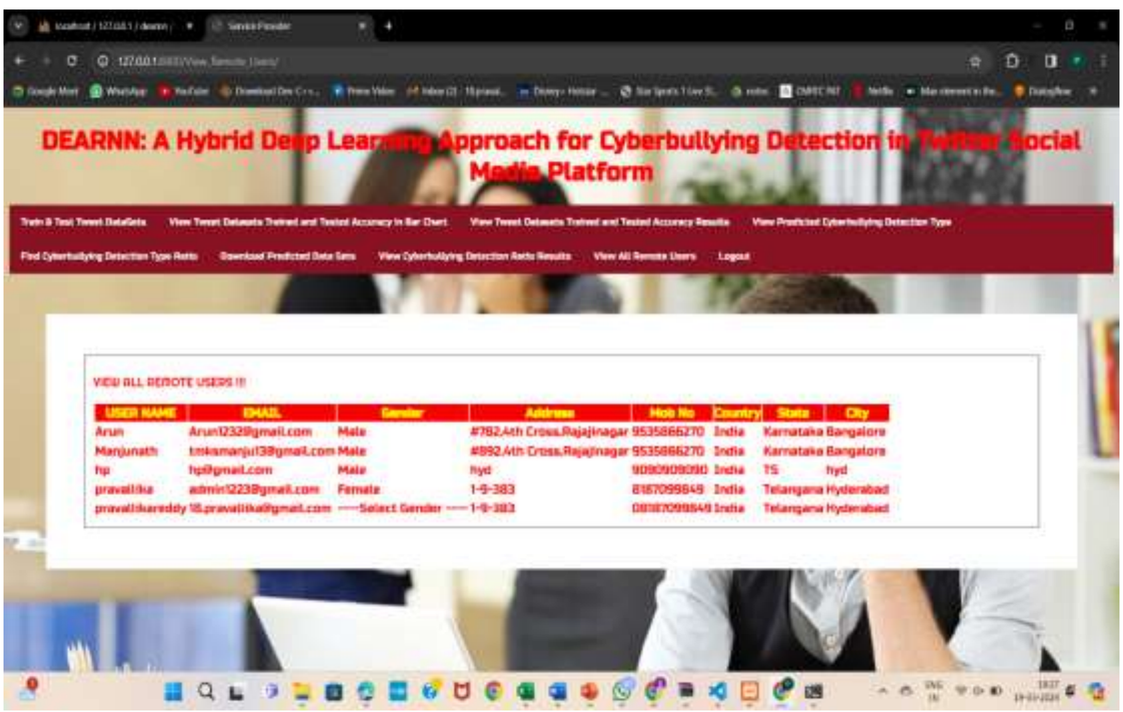


Screenshot 5.4: Open Command prompt

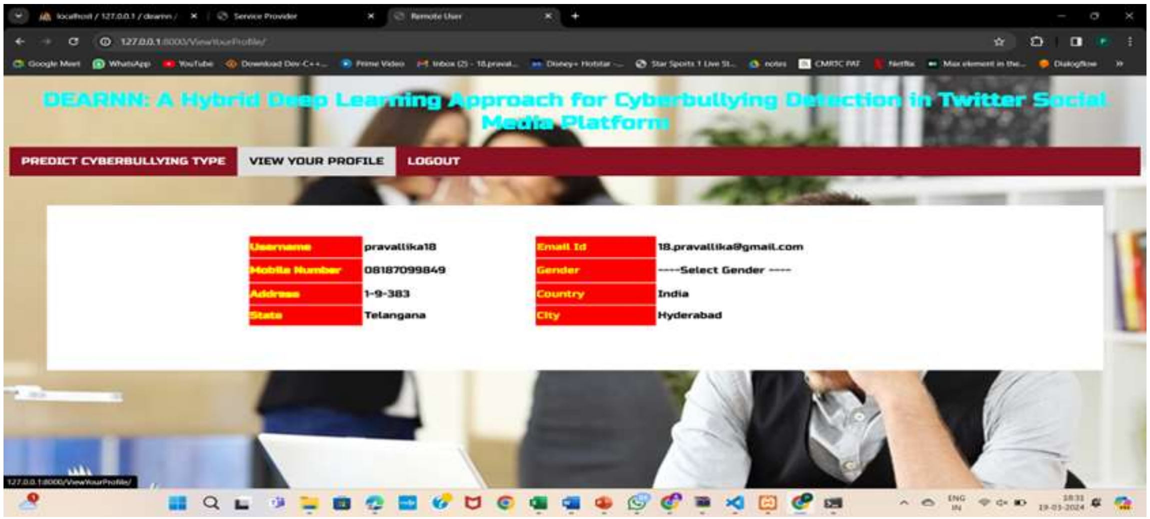
Screenshot 5.5: Login page

Screenshot 5.6: Login service provider

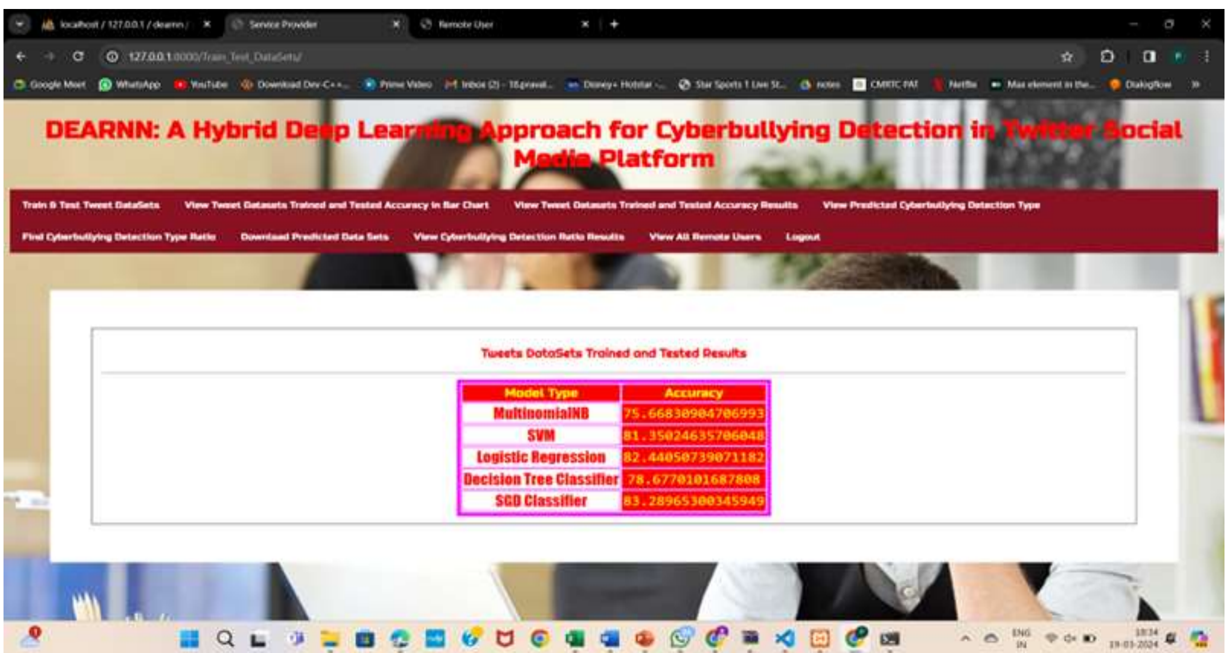
Screenshot 5.7: Register remote user



Screenshot 5.8: View all remote users in service provider

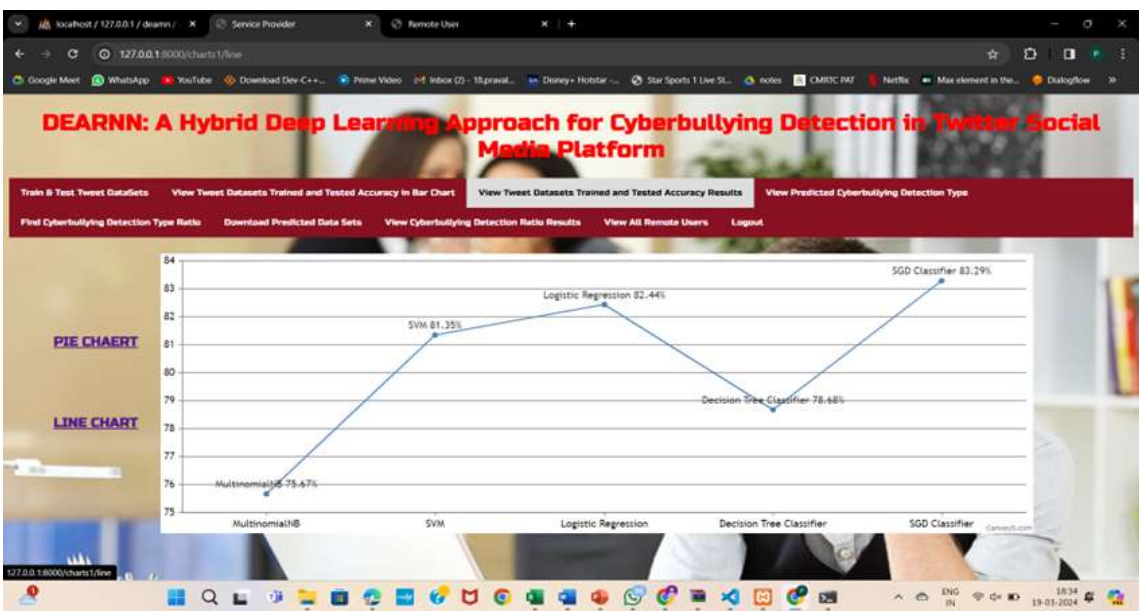


Screenshot 5.9: View your profile of remote user



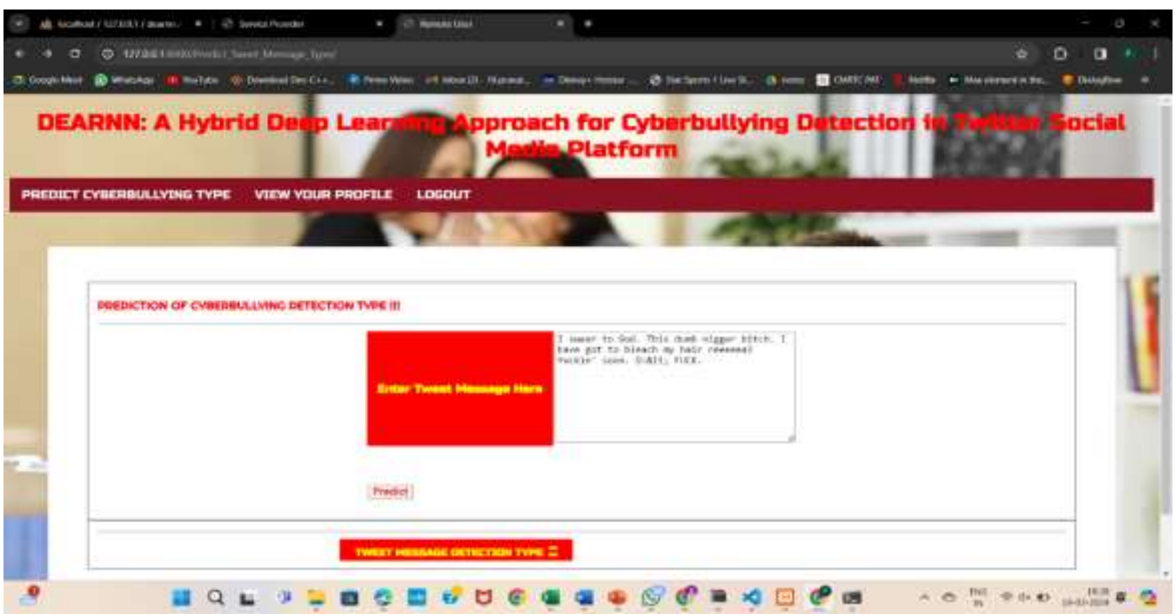
Screenshot 5.10: Train and test tweet data sets

Screenshot 5.11: View tweet Datasets Trained andTested Accuracy in Bar Chart

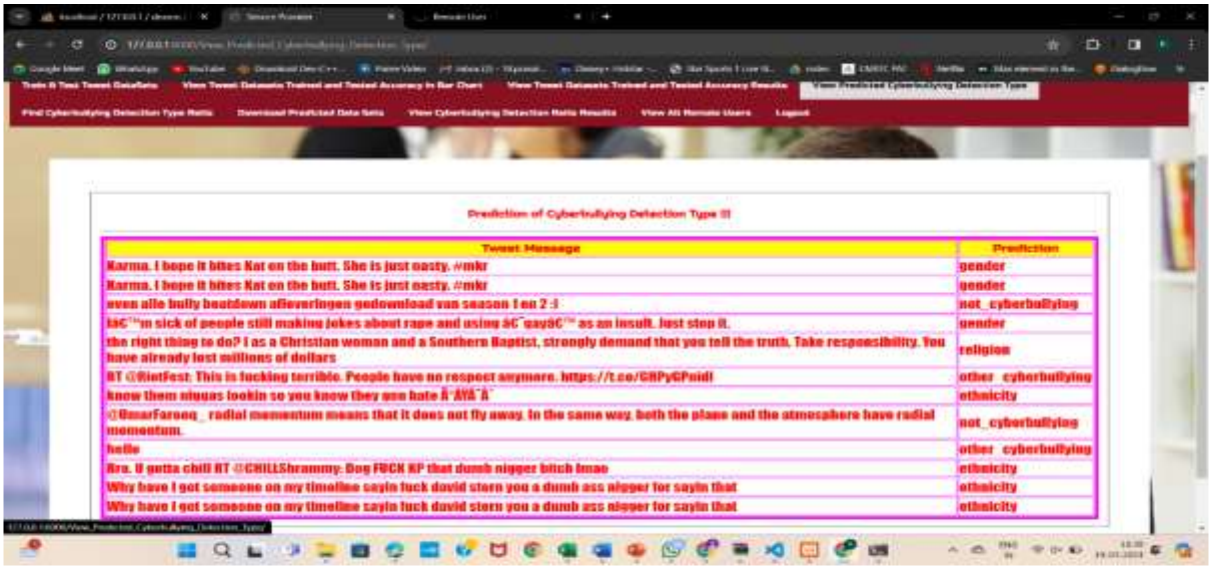


Screenshot 5.12: View Tweet Datasets Trained and Tested Accuracy Results

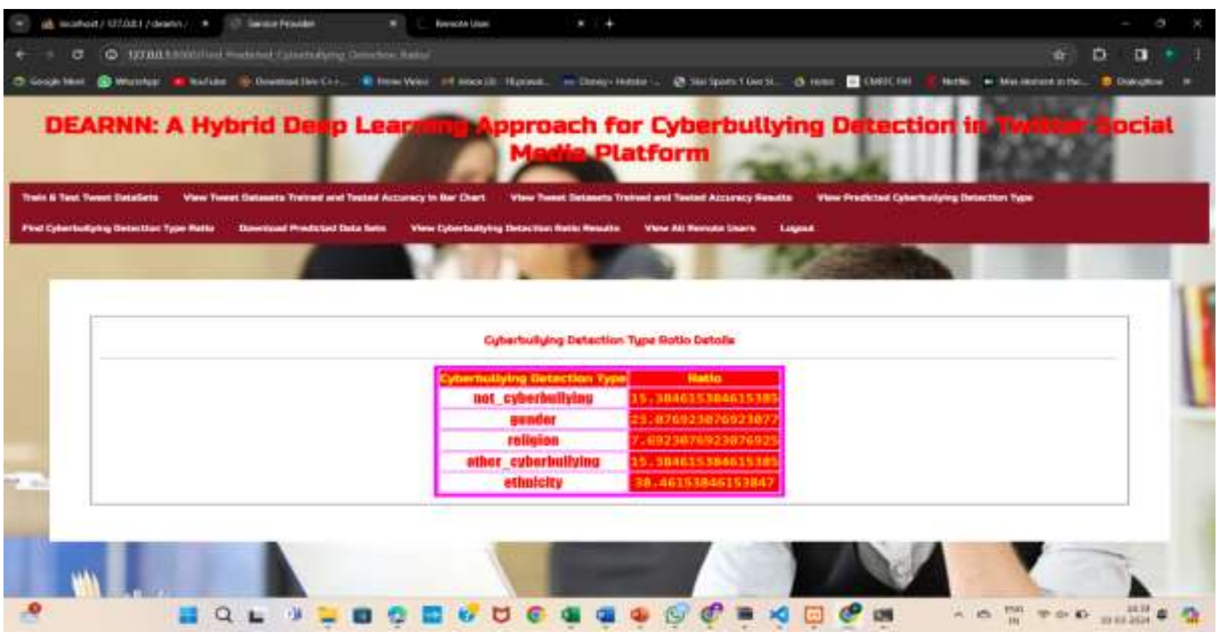
Screenshot 5.13: Pie chart



Screenshot 5.14: Predict cyberbullying type in remote user



Screenshot 5.15: view predicted cyberbullying detection type



Screenshot 5.16: Find cyberbullying Detection Type Ratio

Screenshot 5.17: View Cyberbullying Detection Ratio Results

6.TESTING

6.TESTING

6.1 INTRODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

6.2 TYPES OF TESTING

6.2.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .It is done after the completion of an individual unit before integration. This is a structural testing that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application and/or system configuration. Unit testsensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Integration tests demonstrate thatalthough the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases.

6.3 TEST CASES

6.3.1 CLASSIFICATION

S.NO	Test Case	Excepted Result	Result	Remarks(IF Fails)
1.	Cyberbullying Identification	The system should accurate Ly identify tweets containin g instances of cyberbullying.	Pass	Successful detection of cyberbullying content in tweets.
2.	False Positive Minimization	Ensure the system does not misclassify harmless tweets as cyberbullying.	Pass	Harmless tweets not flagged as cyberbullying.
3.	Contextual Understanding	The system should understand tweet context for accurate cyberbullying detection.	Pass	Proper differentiation between friendly banter and harmful behavior.
4.	Real-time Detection	Prompt identification of cyberbullying instances shortly after occurrence.	Pass	Real-time detection capability confirmed.
5.	Multilingual Support	Effective cyberbullying detection across various languag es commonly used on Twitter.	Pass	Detection across multiple languages demonstrated.
6.	Sarcasm and Irony Handling	Detect cyberbullying even within sarcastic or ironic tweets.	Pass	Accurate detection within sarcastic or ironic contexts.
7.	Scalability	Consistent cyberbullying detection	Pass	Performance maintained under varying tweet loads.
8.	User-based Detection	Accurate identification of cyberbullying	Pass	Successful detection of user-specific cyberbullying instances

7.CONCLUSION

7. CONCLUSION & FUTURE SCOPE

7.1 PROJECT CONCLUSION

The DEA-RNN method, which combines deep autoencoders and recurrent neural networks, is proving effective in spotting cyberbullying on Twitter. It does well in accurately identifying instances of cyberbullying by understanding both the language used and the context. Although more improvements and real-world testing are required, DEA-RNN is a valuable step towards tackling cyberbullying and making online spaces safer. Its potential to encourage responsible online behavior is significant.

7.2 FUTURE SCOPE

The future scope of the "DEA-RNN: A Hybrid Deep Learning Approach for Cyberbullying Detection in Twitter Social Media Platform" project is promising. It can be expanded to incorporate advanced natural language processing techniques for better context understanding and sentiment analysis. Additionally, integration with other social media platforms and real-time monitoring systems could enhance its applicability in combating cyberbullying across diverse online environments. Furthermore, exploring potential collaborations with social media companies and law enforcement agencies can facilitate the implementation of proactive measures for cyberbullying prevention and intervention.

8.BIBLIOGRAPHY

8. BIBLIOGRAPHY

8.1 REFERENCES

[1] F. Mishna, M. Khoury-Kassabri, T. Gadalla, and J. Daciuk, ``Risk factors for involvement in cyber bullying: Victims, bullies and bully_victims,"

Children Youth Services Rev., vol. 34, no. 1, pp. 63_70, Jan. 2012, doi: 10.1016/j.childyouth.2011.08.032.

[2] K. Miller, ``Cyberbullying and its consequences: How cyberbullying is contorting the minds of victims and bullies alike, and the law's limited

available redress," Southern California In++terdiscipl. Law J., vol. 26, no. 2, p. 379, 2016.

[3] A. M. Vivolo-Kantor, B. N. Martell, K. M. Holland, and R. Westby, ``A systematic review and content analysis of bullying and cyber-bullying measurement strategies," Aggression Violent Behav., vol. 19, no. 4, pp. 423_434, Jul. 2014, doi: 10.1016/j.avb.2014.06.008.

[4] H. Sampasa-Kanyinga, P. Roumeliotis, and H. Xu, ``Associations between cyberbullying and school bullying victimization and suicidal ideation,

plans and attempts among Canadian schoolchildren," PLoS ONE, vol. 9, no. 7, Jul. 2014, Art. no. e102145, doi: 10.1371/journal.pone.0102145.

[5] M. Dadvar, D. Trieschnigg, R. Ordelman, and F. de Jong, ``Improving cyberbullying detection with user context," in Proc. Eur. Conf. Inf. Retr.,

in Lecture Notes in Computer Science: Including Subseries Lecture Notes in Arti_cial Intelligence and Lecture Notes in Bioinformatics, vol. 7814, 2013, pp. 693_696.

8.2 GITHUB LINK

https://github.com/207R1A05K0/dea-rnn-A-Hybrid-Deep-Learning-Approach-for-Cyberbullying-Detection-in-Twitter-Social-Media-Platform