**CS6611 CREATIVE AND INNOVATIVE PROJECT**

**B.E CSE VI Q - BATCH**

LAB NO:12 26/04/2021

TEAM MEMBERS TEAM NO:15

|  |  |  |
| --- | --- | --- |
| **S.No.** | **REG. NO.** | **NAME** |
| 1 | 2018103072 | SUJITHA R |
| 2 | 2018103557 | LAVANYA N.J |
| 3 | 2018103627 | RAYAVARAM KETHANA PRAGNA |

**Project Title:** RECOMMENDING CLASSIFYING METHOD AND SARCASM DETECTION USING TWITTER DATA

|  |  |
| --- | --- |
| Observation document (5) |  |
| On the Spot exercise (5) |  |
| Laboratory exercises identified (15) |  |
| Total (25) |  |
| Signature |  |

|  |  |
| --- | --- |
| Observation document (5) |  |
| On the Spot exercise (5) |  |
| Laboratory exercises identified (15) |  |
| Total (25) |  |
| Signature |  |

|  |  |
| --- | --- |
| Observation document(5) |  |
| On the spot exercise(5) |  |
| Laboratory exercises identified(15) |  |
| Total(25) |  |
| Signature |  |

|  |  |
| --- | --- |
| Observation document (5) |  |
| On the Spot exercise (5) |  |
| Laboratory exercises identified (15) |  |
| Total (25) |  |
| Signature |  |

### 1)ABSTRACT

Sarcasm is the use of positive words to annoy someone, or for humorous purposes.For instance,When you get an “F” on an exam and your friend says, “Nice job,Einstein.” Here, actually the friend annoyed Einstein for getting low grade but with the positive word like “Nice”. Sarcasm detection is a very narrow research field in NLP, a specific case of sentiment analysis where instead of detecting a sentiment in the whole spectrum, the focus is on sarcasm. Therefore the task of this field is to detect if a given text is sarcastic or not.For the classification we have used models of deep learning like CNN (Convolutional Neural Networks) and RNN(Recurrent Neural Networks).Also we have compared the classification models by analyzing their performance measures and have recommended which model suits best for sarcasm detection.Further,On e-commerce websites such as amazon, many times customers make the use of sarcasm in their reviews in an attempt to criticize the product. With the help of sarcasm detection, products can be classified into the relevant categories with more accuracy.

**Keywords** - Sarcasm,Sarcasm detection,NLP,CNN,RNN

**2)INTRODUCTION**

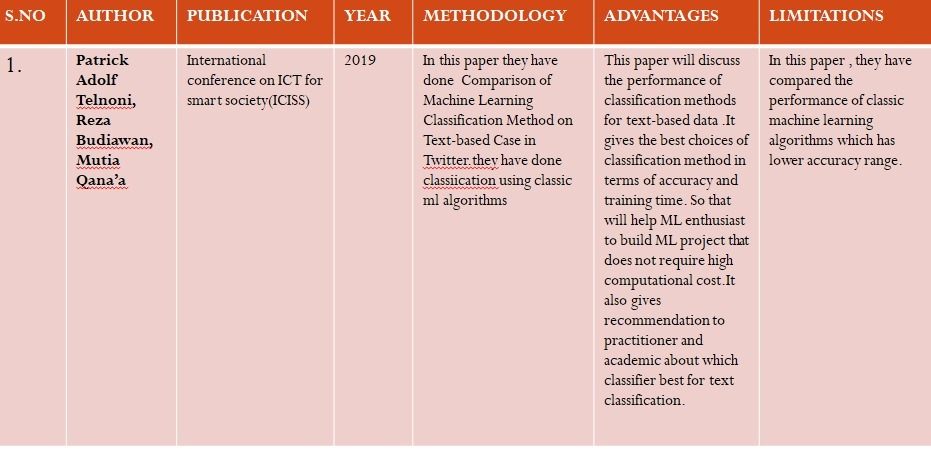
Users of Twitter may write a negative tweet if they are not feeling good with some news when they came to know it through social media. The tweet can be regular one if it is written in the straightforward way. The user can also make use of sarcasm in the post to mock or criticize any announcement/news. Detection of sarcasm in language in the form of text has been a challenging problem. This is because a sarcastic text might appear as a regular text. Using traditional opinion mining techniques can lead to incorrect classification of the . A sarcastic sentence written with the view of expressing contempt might contain abundant positive words. Humans can figure out sarcasm easily when they know the tone of the sentence and the context in which it is used. We can thus improve the performance of a sarcasm detection system if we use contextual features along with traditional features to train our model.And we are going to recommend best classifying algorithm or model based on the model performance measures.

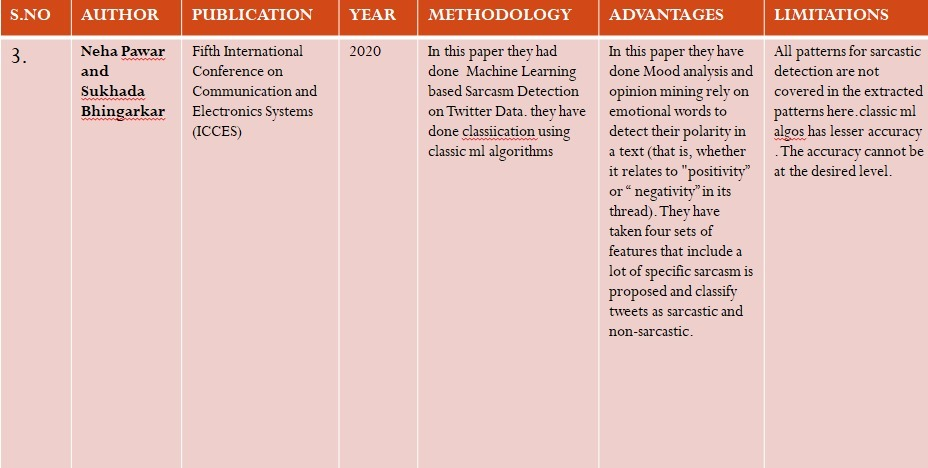
**3)PROBLEM STATEMENT**

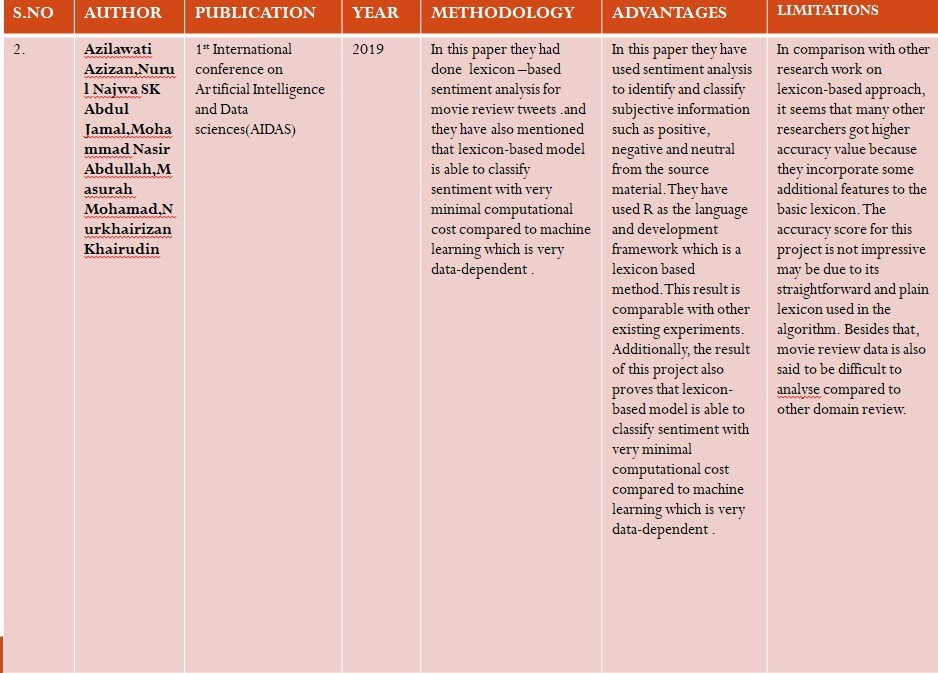
Customers of e-commerce website like amazon may write a negative review of a product if they are not happy with the product’s price, quality or service of the dealer. The review can be regular one if it is written in the straightforward way. The customer can also make use of sarcasm in the review to criticize the product. Suppose a customer buys a very costly headphone and it turns out the quality of sound is not very good. He then writes a review as “the speaker is so good that I can hear sounds from other galaxies! Totally worth the money” The user is obviously not satisfied with the product and is mocking the product’s quality and price using sarcasm.So,in these kind of situations it seems very difficult to detect the negative comment with the given positive words.So,the user personalization of the website becomes inaffective.Also,finding the classifier model with better accuracy matters a lot.Since it signifies the efficiency of the system.

**4)LITERATURE SURVEY**

**1.DETAILED EXPLANATION**









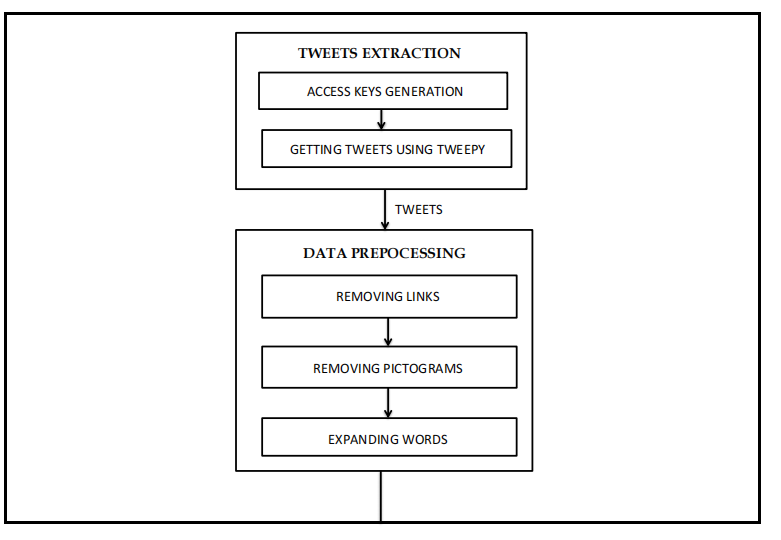
**2.SUMMARY:-**

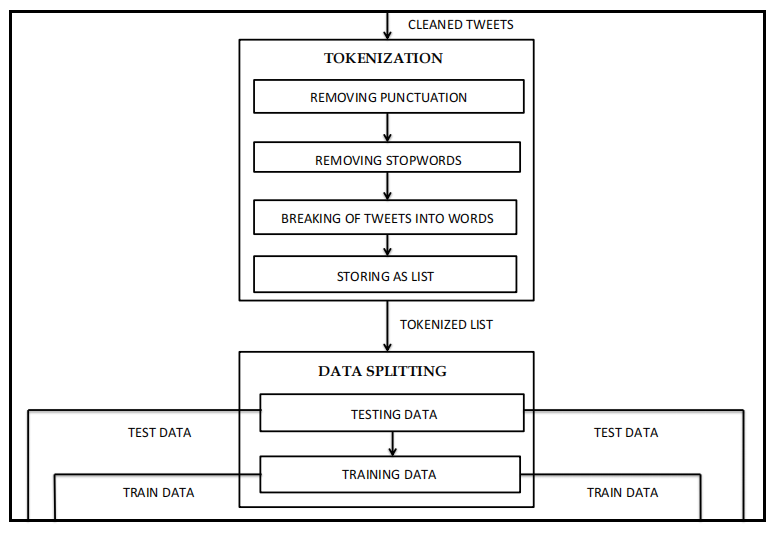
We detect the sarcasm by deep neural networks(solving the cons of paper [2] using deep neuro network as explained in paper[4].We used the supervised learning approach explained in paper[2] but we have done it as transfer learning approach.To do sarcasm detection ,we should do some pre-processing to the dataset that are explained in paper[3].we also using the tweets taken in general(solving the cons of paper[3]) and we have used GLOVE model (solving cons of paper[4])for better performance.

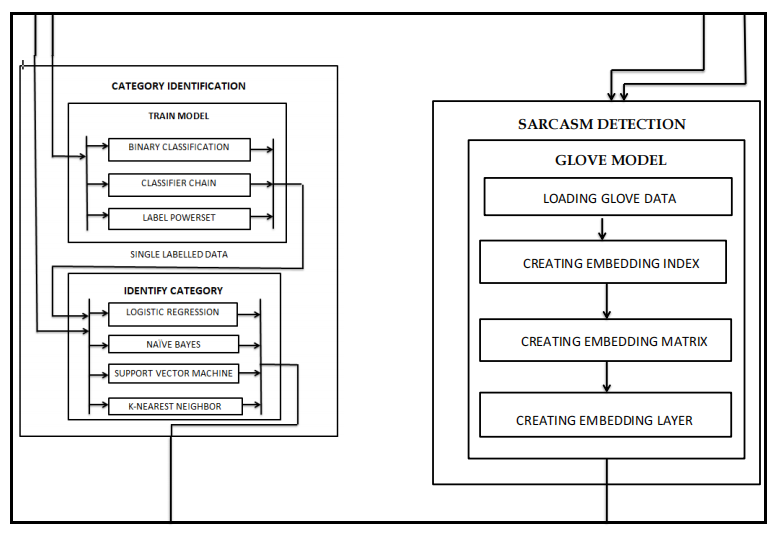
**5)ISSUES IDENTIFIED:-**

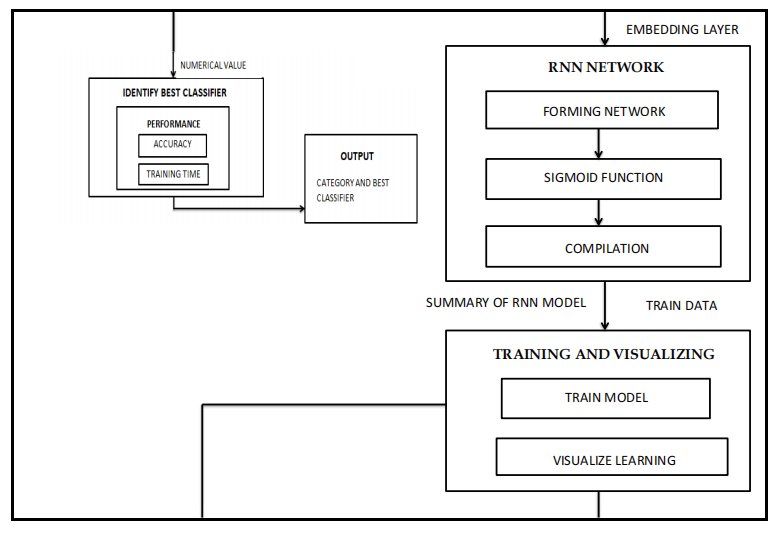
All the papers we took have only discussed about identifying sarcasm using calssic machine learning classifications algorithms . And they didn't use neural networks which has higher performance. Also that they have analysed through Movie Review Tweets which is not very efficient.These are the issues we have identified.

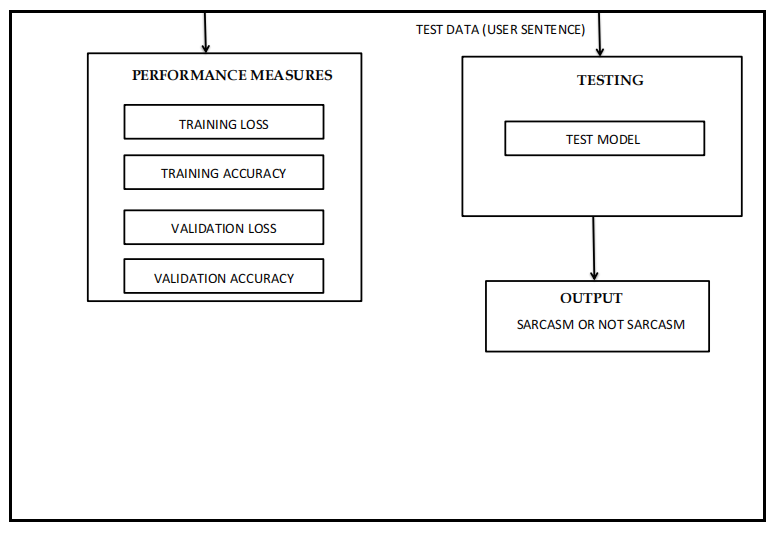
**6)BLOCK DIAGRAM**











**7)MODULES WITH I/O:-**

**MODULE 1 : TWEETS EXTRACTION**

**Input:** access keys with csv file to store tweets

**Output:** tweets

**PSEUDO CODE:**

Module TweetsExtraction():

import python libraries

declare access keys

set the authorization variable

call Tweepy.API()

Begin loop

for tweets in tweepy.Cursor

do

Extract Sarcastic tweets

write it into a csv file

repeat

Endloop

repeat for regular tweets

EndModule

Twitter is a popular social network where users share messages called tweets. Twitter allows us to mine the data of any user using Twitter API or Tweepy. The data will be tweets extracted from the user. The first thing to do is get the consumer key, consumer secret, access key and access secret from twitter developer available easily for each user. These keys will help the API for authentication.

Tweepy is one of the libraries that should be installed using pip. Now in order to authorize our app to access Twitter on our behalf, we need to use the OAuth Interface. Tweepy provides the convenient Cursor interface to iterate through different types of objects. Twitter allows a maximum of 3200 tweets for extraction. These all are the prerequisite that have to be used before getting tweets of a user.

**Types of tweets to be extracted:**

We are planning to collect tweets, using Twitter’s streaming API. To collect sarcastic tweets, we queried the API for tweets containing the hashtag ‘‘#sarcasm’’. Although this hashtag is not the best way to collect sarcastic tweets, the fact that this hashtag can be used for this purpose. However, the hashtag cannot be reliable and is used mainly for 3 purposes:

* to serve as a search anchor,
* to serve as a sarcasm marker in case of a very subtle sarcasm where it is very hard to get the sarcasm without an explicit marker, as in ‘‘Today was fun. The first time since weeks! #Sarcasm’’,
* to clarify the presence of sarcasm in a previous tweet, as in ‘‘I forgot to add #sarcasm so people like you get it!’’.

In total, we are going to collect 58 609 tweets with the hashtag ‘‘#sarcasm’’, which we clean up by removing the noisy and irrelevant ones, as well as ones where the use of the hashtag does fall into one of the two first uses of the three described above. As for non-sarcastic tweets, we are going to collect tweets dealing with different topics and made sure they have some emotional content.

**MODULE 2 - DATA PRE-PROCESSING(CLEANING)**

**Input:** raw tweets from twitter API

**Output:** Pre-processed tweets

Module DataPreprocessing()

Load data files

combine them into a single file

Begin loop

for tweet in tweetscolumn

do

CleanText(tweet):

tweet.toLowerCase

remove emoticons,links,symbols & pictographs,transport & map symbols,

flags (iOS),special characters[,.\"\'!@#$%^&\*(){}?/;`~:<>+=-]

perform expansion of shortened words[I'm - I am]

repeat

Endloop

return cleanedtweet

EndModule

We retrieve tweets with Twitter API by using the manually produced query. Final queries for the retrieval are a combination of sarcastic and non-sarcastic tweets. However, there is a problem with the final queries. Tweets often contain noisy/unwanted data. To remove such noisy data, we perform morphological analysis.Here,in this module we have defined a function called CleanText() in which tweet from tweets column of the dataframe undergo several refinery process which includes converting the tweets to its lowercase form, removing float/numeric values if it contains any in between the text,removing special characters like [,.\"\'!@#$%^&\*(){}?/;`~:<>+=-] and by using regular expressions we have expanded the shortened form of words to its full form,removed pictograms,emojis if any.So,from this function we can get a cleaned form of tweets which can be further tokenized by the next module.

**MODULE 3:-TOKENIZATION**

**Input:** cleaned tweets

**Output:** tokenized list

Module Tokenization()

import word\_tokenize,stopwords from nltk

call CleanTokenize(df):

CleanTokenize(df):

Separating the tweets column from file

convert it into list

Begin loop

for tweet in tweetslist

do

CleanText(tweet)

tokenize the text

remove puntuations

remove non alphabetic characters

remove stop words

repeat

Endloop

return Tokenizedlist

EndModule

The cleaned text from the previous module gets tokenized and stored into a list in this module.We break each sentence into words in tokenization. If a tweet contains unwanted information then that tweet is removed from the dataset.If a tweet contains any stopwords or punctuation they too removed using nltk toolkit.In this module we had make use of nltk toolkit and it is the Natural Language Toolkit (NLTK) is a platform used for building Python programs that work with human language data for applying in statistical natural language processing (NLP). It contains text processing libraries for tokenization, parsing, classification, stemming, tagging and semantic reasoning.In this module,it will read each tweets line and separate the words in the each tweet and store it in a list.For each tweets line,we will have one list as output.

**MODULE 4:-DATA SPLITTING**

**Input:** corpus vector space(extracted data)

**Output:** splitted data set(test ,train)

Module DataSplit()

Declaring validation\_split and max\_length

Creating an object of class Tokenizer()

fitting our data by fit\_on\_textsconverting into sequences using text\_to\_sequences()

finding unique tokens and vocabulary size

call pad\_sequences

declaring sentiment

call np.random.shuffle()

print dimensions of train,test data.

Visualize wordcloud

EndModule

In this module,firstly we have converted all the words into numbers most particularly vectors.Then for datasplit we have used softmax class.This module just splits the tweets into test and train data set in the ratio 2:8 which means splits the 100% dataset into 20%test data and 80% train data.Then for accuracy,we have set max\_length parameter as 25 which tells that the maximum number of words to be taken from a tweet line is 25. .After that,shuffling happens so that no partiality occurs.We will use tokenizer() from keras which encode the text to be given as input to machine learning model.After datasplit,we have visualized a word cloud by taking the words from train data and depicted a picture which contains words of different frequencies.

**MODULE 5:- GLOVE MODEL**

**Input:** train\_data,test\_data

**Output:** embedding layer

Module GloveModel()

Declaring embedding index{}

Declaring dimension for matrix

loading Glove model

Begin loop

for line in glovefile

do

call split()

storing the first word in a variable

declare coef array with list except the first word

mapping the first word with its co-efficients in embedding\_index{}

End for

End loop

Print(No of word vectors found)

Initialize embedding\_matrix as Zero matrix

Begin loop

for index in word\_index.items

do

if (word)

add index to matrix

increase count

End for

End loop

EndModule

Here,in this module we have used transfer learning approach i.e.,reusing the pre-trained model and making some changes on top of that model to work for our desired purpose.Pre-trained Glove model developed as an open-source project at Stanford will be loaded into our project.The GLOVE file contains many words with their corresponding vector representation in 100 dimensions since we have downloaded glove.twitter.100d.txt .From the GLOVE file we parse line by line and for each line we have make a embedding\_index dictionary which contais the word as key and its vector representation as value.Then,It makes a co-relation matrix among words in our project.If the word from our data is present in GLOVE model data then we add that word’s index to the matrix and the row represents the vector representation of that particular word.After that we will create an embedding layer from the embedding\_matrix which is to be passed as first layer to form a deep neural networks.

**MODULE 6:-BULIDING RECURRENT NEURAL NETWORK(RNN)**

**Input:** embedding layer

**Output:** summary of that model

Module Build\_RNN()

call model.sequential()

add embedding layer as first layer

add LSTM ,RNN

call model.compile()

print(Model Summary)

EndModule

We have used Sequential()module from keras to build a neural network.After that,the passed embedding layer from the previous module and added as the first layer to our Neural Network.Then we have added the LSTM layer(Long Short Term Memory) with 54 neurons and RNN.RNNs are designed to recognize a data's sequential characteristics and use patterns to predict the next likely scenario.Since ,it’s a binary classification we have used activation function as SIGMOID(),and the last layer would contain only one node.And then,on compilation,our RNN model get ready and have displayed the summary of our built model.

**MODULE 7:-BULIDING CONVOLUTIONAL NEURAL NETWORK(CNN):-**

**Input:** embedding layer

**Output:** summary of that model

Module Build\_CNN()

  call model.sequential()

add embedding layer as first layer

add CNN

add LSTM

call model.compile()

print(Model Summary)

EndModule

We have used Sequential()module from keras to build a neural network. After that,the passed embedding layer from the previous module is added as the first layer to our Neural Network.Then we have added a convolutional layer (which are used mainly for image processing, classification, segmentation and also for other auto correlated data. A convolution is essentially sliding a filter over the input)and then an LSTM layer on top of CNN layer. Further we have called ReLu activation function for CNN and sigmoid function for LSTM layer.And then,on compilation,our model get ready and have displayed the summary of our built model.

**MODULE 8:-TRAINING AND VISUALIZING**

**Input:** train data,validation\_data

**Output:** Accuracy and loss values

Module Training\_and\_Visualization()

Declaring epochs,batch\_size,validation\_data

call model.fit()

declare label,color

call plt.legend()

call plt.figure()

call plt.show()

EndModule

In this module we have trained the models separately by providing train data,validation data.Also specified the running parameters such as epochs,batchsize then we have fitted the model and had run it.As a result of this we can view the incremental accuracy values and decremental loss values which are changing from epochs to epochs. We have utilized matplotlib library of python to use graphs and to visualize the learning.

**MODULE 9:-TESTING:-**

**Input:** user sentence

**Output:** sarcasm or not

Module Testing()

Declaring function predict\_sarcasm(s)

getting the user sentence

call CleanTokenize(s)

call text\_to\_sequence()

call pad\_sequence()

call model.predict()

if (pred[0][0] > 50)

return "It's a sarcasm"

else "It's not a sarcasm"

# to test

call predict\_sarcasm(user Input Sentence)

EndModule

For detecting Sarcasm,we have a testing function predict\_sarcasm() that accepts user input sentence from an external file .The sentence then undergo necessary pre-processing , tokenization etc.,.. and finally it gets predicted by the trained RNN and CNN models. Then the models provide a probabilistic value for that sentence based on the prediction. By that value we come to an conclusion that the given input sentence is sarcatic or not .

**MODULE 10:- PERFORMANCE MEASURES:**

**Input:** training model

**Output:** performance measures values

Module PerformanceMeasures()

Calculate Precision,Recall,F1score,Confusion matrix

#comparing the accuracies

if CNN\_training\_accuracy > RNN\_training\_accuracy

print "Cnn is better than Rnn by comparing training accuracy"

else

print "Rnn is better than Cnn by comparing training accuracy"

if CNN\_validation\_accuracy > RNN\_validation\_accuracy

print "Cnn is better than Rnn by comparing validation accuracy"

else

print "Rnn is better than Cnn by comparing validation accuracy"

#comparing the losses

if CNN\_validation\_loss < RNN\_validation\_loss

print "Cnn is better than Rnn by comparing validation loss"

else

print "Rnn is better than Cnn by comparing validation loss"

if CNN\_training\_loss < RNN\_training\_loss

print "Cnn is better than Rnn by comparing training loss"

else

print "Rnn is better than Rnn by comparing training loss"

EndModule

In this module we have calculated the precision,F1score,Recall,Confusion matrix for each of the training models and on training we will get performance measures like

* Training accuracy
* Validation accuracy
* Training loss
* Validation loss

by comparing these values in this performance model we can find out which model serves best for sarcasm detection with higher performance.

1. **CONTRIBUTION/INNOVATION HANDLED:-**

By analysing all the papers we found that all these papers have done model training using normal classifiers.They have not done using neural networks which has higher performance than normal classifiers.we are going to do sarcasm detection using neural networks like RNN and CNN. We are going to use glove model for doing this.And we are going to recommend model which has high performance.In module 6 and 7 we have built our Innovative CNN and RNN models using keras().

**9.RESULTS DISCUSSION**

**TABULAR COLUMN1:**

**Input- A sample text from the dataset**

**Outputs - preprocessed data(module-2) and tokenized data(module-3)**

**CLEAN\_TEXT – MODULE(2)**

**CLEAN\_TOKENIZED-MODULE(3)**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **DATASET**  **(INPUT)** | **CLEAN\_TEXT**  (INTERMEDIATE  LEVEL OUTPUT) | **CLEAN\_TOKENIZED**  (OUTPUT) |
| 1. | I don't need a license  from the state to drive a car. I already know how to drive.   The state can't tell me what to do. | i do not need a  license from the state to drive a car i already  know how to drive the state can not tell me what to  do | ['need',  'license',  'state',  'drive',  'car',  'already',  'know',  'drive',  'state',  'tell'] |
| 2. | Today's profit \xf0\x9f\x92\xb8\xf0\x9f\x8e\xaf\xf0\x9f\x93\x88\n.\n.\n.\n.\n.\n.\n.\n.\n.\n#profit #regular #everyday #forex #forextrading #forexsignals #forextrader \xe2\x80\xa6 https://t.co/K0sL7jXaTG | todays profit profit regular  everyday forex  forextrading  forexsignals  forextrader | ['todays',  'profit',  'profit',  'regular',  'everyday',  'forex',  'forextrading',  'forexsignals',  'forextrader'] |
| 3. | @bhatiamanu What is that #Sarcasm ?? I literally read it as scam \xf0\x9f\xa4\xa3. May be bcos of spending too much time on twitter | what is that  sarcasm I  literally read it as scam may be  bcos of spending too much time on twitter | ['sarcasm',  'literally',  'read',  'scam',  'may',  'bcos',  'spending',  'much',  'time',  'twitter'] |
| 4. | b'It\'s said that #sarcasm is the lowest form of #wit. To  that I say ""No, really? Hey, somebody give this person a #Nobel #peace #prize! | it is said that  sarcasm is the  lowest form of  wit to that i say no really hey  somebody give  this person a  nobel peace prize | ['said',  'sarcasm',  'lowest',  'form',  'wit',  'say',  'really',  'hey',  'somebody',  'give',  'person',  'nobel',  'peace',  'prize'] |
| 5. | b""\xe3\x80\x90New Arrival\xe3\x80\x91 Atolo Pen's Chinese Literary Copy Book just arrival! Most of them are classic poems. Check it out! | new arrival atolo pens chinese literary copy book just arrival most of them are classic poems check it out | ['atolo',  'pens',  'chinese',  'literary',  'copy',  'book',  'arrival',  'classic',  'poems',  'check'] |
| 6. | b""@RichLowry Yes and no one died in #PuertoRico after the hurricane and over 500K people didn't die from #COVID19. It\xe2\x80\xa6 https://t.co/MuDoVCNESG" | brichlowry yes and no one died in puertorico after the hurricane and over 500k people didnt die from covid19 it | ['brichlowry',  'yes',  'one',  'died',  'puertorico',  'hurricane',  'people',  'didnt',  'die'] |
| 7. | @LeaderMcConnell lied, again. Call me  shocked. Republicans whole platform is  gaslighting!\n\nWow. | lied again call me shocked republicans whole platform is gaslighting wow | ['lied',  'call',  'shocked',  'republicans',  'whole',  'platform',  'gaslighting',  'wow'] |
| 8. | The #Original #DonutShop #Regular   #MediumRoast #Coffee 18 ct. #Keurig #Kcups   BB 10/21! #bargains #beverages\xe2\x80\xa6   https://t.co/PJaQrqkuAO | the original donutshop regular mediumroast coffee 18 ct keurig kcups bb 1021 bargains beverages | ['original',  'donutshop',  'regular',  'mediumroast',  'coffee',  'ct',  'keurig',  'kcups',  'bb',  'bargains'] |
| 9. | Surprized everytime how lockdown solves so many problems. #sarcasm \n#NoCovidJETZT BS is trending again. | surprized everytime how lockdown solves so many problems sarcasm nocovidjetzt bs is trending again | ['surprized',  'everytime',  'lockdown',  'solves',  'many',  'problems',  'sarcasm',  'nocovidjetzt',  'bs',  'trending'] |
| 10. | 79% off #Men #Regular #Fit #Short  #Sleeve #Cotton #T-Shirt \n\nClip the  Extra 79% off Coupon under the listed   Price \nW\xe2\x80\xa6.   https://t.co/Uy8DUKd3c3 | 79 off men regular fit short sleeve cotton tshirt clip the extra 79 off coupon under the listed price | ['men',  'regular',  'fit',  'short',  'sleeve',  'cotton',  'tshirt',  'clip',  'extra',  'coupon',  'listed',  'price'] |
| 11. | @bopinion Unlove for passive turned into  a misguided love for #ARK? Cathie's only  feat has been mimicking this reta  \xe2\x80\xa6 https://t.co/OXlVT8UNV4 | unlove for passive turned into a misguided love for ark cathies only feat has been mimicking this reta | ['unlove',  'passive',  'turned',  'misguided',  'love',  'ark',  'cathies',  'feat',  'mimicking',  'reta'] |
| 12. | b'Two years of full Regular recruiting and we are still going strong @BritishArmy. With so many roles and opportunit\xe2\x80\xa6 https://t.co/KGMLtOq7fE' | two years of full regular recruiting and we are still going strong with so many roles and opportunities | ['two',  'years',  'full',  'regular',  'recruiting',  'still',  'going',  'strong',  'many',  'roles',  'opportunities'] |
| 13. | 100 crore target is not  possible sir \nAs   night curfew along with hotel and bar are  closed.\n#MumbaiPolice  to\xe2\x80\xa6  https://t.co/TsejqI3uLJ | 100 crore target is not possible sir as night curfew along with hotel and bar are closed mumbaipolice | ['crore',  'target',  'possible',  'sir',  'night',  'curfew',  'along',  'hotel',  'bar',  'closed',  'mumbaipolice'] |
| 14. | b'@BBCSport People, workers having rights.....absolutely not! They should feel lucky to have a job, even it means ser\xe2\x80\xa6 https://t.co/OBGK9uMUEM | people workers having rights absolutely not they  should feel lucky to have a job even it means ser | ['people',  'workers',  'rightsabsolutely',  'feel',  'lucky',  'job',  'even',  'means'] |
| 15. | I was depressed. He asked me to be happy.I am not depressed anymore. | i was depressed he asked me to be happy i am not depressed anymore | ['depressed',  'asked',  'happy',  'depressed',  'anymore'] |

**TABULAR COLUMN2:**

Input- tokenized data from different set of sarcastic and regular tweet datasets

Output - DATASPLIT(MODULE4) which represents count of the unique words present in the

|  |  |  |
| --- | --- | --- |
| **S.NO** | **WORD CLOUD** | **DATA SPLIT** |
| 1. |  |  |
| 2. |  |  |
| 3. |  |  |
| 4. |  |  |
| 5. |  |  |
|  |  |  |
| 6 |  |  |
| 7 |  |  |

**TABULAR COLUMN3:**

**INPUT: Training and Testing Data is given to train the RNN model(MODULE6)**

**OUTPUT: Accuracy and loss values of training and testing data for 15 Epochs**

|  |  |
| --- | --- |
| INPUT(TRAINING FOR 15 EPOCHS) | OUTPUT(ACCURACY AND LOSS VALUES) |
| model.fit(X\_train\_pad,  y\_train, batch\_size=32,   epochs=15,  validation\_data=(X\_test\_pad,  y\_test),verbose=2) | Epoch 1/15  41/41 - 5s - loss: 0.2675 - acc: 0.9722 - val\_loss: 0.1414 - val\_acc: 0.9690  Epoch 2/15  41/41 - 2s - loss: 0.1087 - acc: 0.9776 - val\_loss: 0.1390 - val\_acc: 0.9690  Epoch 3/15  41/41 - 2s - loss: 0.1088 - acc: 0.9776 - val\_loss: 0.1396 - val\_acc: 0.9690  Epoch 4/15  41/41 - 2s - loss: 0.1081 - acc: 0.9776 - val\_loss: 0.1377 - val\_acc: 0.9690  Epoch 5/15  41/41 - 2s - loss: 0.0927 - acc: 0.9745 - val\_loss: 0.0944 - val\_acc: 0.9814  Epoch 6/15  41/41 - 2s - loss: 0.0758 - acc: 0.9814 - val\_loss: 0.1006 - val\_acc: 0.9783  Epoch 7/15  41/41 - 2s - loss: 0.0653 - acc: 0.9838 - val\_loss: 0.0945 - val\_acc: 0.9814  Epoch 8/15  41/41 - 2s - loss: 0.0633 - acc: 0.9869 - val\_loss: 0.0779 - val\_acc: 0.9752  Epoch 9/15  41/41 - 2s - loss: 0.0518 - acc: 0.9884 - val\_loss: 0.0924 - val\_acc: 0.9814  Epoch 10/15  41/41 - 2s - loss: 0.0440 - acc: 0.9876 - val\_loss: 0.0996 - val\_acc: 0.9814  Epoch 11/15  41/41 - 2s - loss: 0.0427 - acc: 0.9907 - val\_loss: 0.0993 - val\_acc: 0.9783  Epoch 12/15  41/41 - 2s - loss: 0.0479 - acc: 0.9892 - val\_loss: 0.0944 - val\_acc: 0.9783  Epoch 13/15  41/41 - 2s - loss: 0.0343 - acc: 0.9938 - val\_loss: 0.0975 - val\_acc: 0.9783  Epoch 14/15  41/41 - 2s - loss: 0.0376 - acc: 0.9923 - val\_loss: 0.0938 - val\_acc: 0.9752  Epoch 15/15  41/41 - 2s - loss: 0.0352 - acc: 0.9938 - val\_loss: 0.0812 - val\_acc: 0.9814 |

**TABULAR COLUMN4:**

**INPUT: Training and Testing Data is given to train the CNN model(MODULE7)**

**OUTPUT: Accuracy and loss values of training and testing data for 15 Epochs**

|  |  |
| --- | --- |
| **Training for 15 Epochs**  **(INPUT)** | **Accuracy and loss values(OUTPUT)** |
| model\_conv.fit(  X\_train\_pad,   y\_train,batch\_size=32,  epochs=15,  validation\_data=  (X\_test\_pad,   y\_test),   verbose=2) | Epoch 1/15  41/41 - 3s - loss: 0.2101 - accuracy: 0.9520 - val\_loss: 0.1065 - val\_accuracy: 0.9783  Epoch 2/15  41/41 - 0s - loss: 0.1099 - accuracy: 0.9753 - val\_loss: 0.0996 - val\_accuracy: 0.9783  Epoch 3/15  41/41 - 0s - loss: 0.0939 - accuracy: 0.9753 - val\_loss: 0.0769 - val\_accuracy: 0.9783  Epoch 4/15  41/41 - 0s - loss: 0.0665 - accuracy: 0.9814 - val\_loss: 0.0671 - val\_accuracy: 0.9814  Epoch 5/15  41/41 - 0s - loss: 0.0556 - accuracy: 0.9861 - val\_loss: 0.0619 - val\_accuracy: 0.9845  Epoch 6/15  41/41 - 1s - loss: 0.0291 - accuracy: 0.9938 - val\_loss: 0.0689 - val\_accuracy: 0.9814  Epoch 7/15  41/41 - 0s - loss: 0.0189 - accuracy: 0.9954 - val\_loss: 0.0766 - val\_accuracy: 0.9814  Epoch 8/15  41/41 - 1s - loss: 0.0128 - accuracy: 0.9969 - val\_loss: 0.0813 - val\_accuracy: 0.9876  Epoch 9/15  41/41 - 1s - loss: 0.0115 - accuracy: 0.9961 - val\_loss: 0.1035 - val\_accuracy: 0.9752  Epoch 10/15  41/41 - 1s - loss: 0.0081 - accuracy: 0.9969 - val\_loss: 0.0816 - val\_accuracy: 0.9845  Epoch 11/15  41/41 - 1s - loss: 0.0040 - accuracy: 0.9985 - val\_loss: 0.0875 - val\_accuracy: 0.9845  Epoch 12/15  41/41 - 1s - loss: 0.0122 - accuracy: 0.9954 - val\_loss: 0.1160 - val\_accuracy: 0.9659  Epoch 13/15  41/41 - 0s - loss: 0.0109 - accuracy: 0.9977 - val\_loss: 0.0763 - val\_accuracy: 0.9783  Epoch 14/15  41/41 - 0s - loss: 0.0039 - accuracy: 0.9985 - val\_loss: 0.0782 - val\_accuracy: 0.9814  Epoch 15/15  41/41 - 0s - loss: 0.0141 - accuracy: 0.9946 - val\_loss: 0.0869 - val\_accuracy: 0.9814 |

**TABULAR COLUMN5:**

**FINAL RESULT:**

**INPUT: Sample Text**

**OUTPUT: Predicts whether the text is sarcastic or not**

|  |  |  |
| --- | --- | --- |
| S.NO | INPUT | OUTPUT |
| 1. | I don't need a license from the state  to drive a car. I already know how to  drive. The state can't tell me what to do. | It's a sarcasm. |
| 2. | Today's profit \xf0\x9f\x92\xb8\xf0\x9f\x8e  \xaf\xf0\x9f\x93\x88\n.\n.\n.\n.\n.\n.\n.\n.\n.  \n#profit #regular #everyday #forex  #forextrading #forexsignals #forextrader   \xe2\x80\xa6 https://t.co/K0sL7jXaTG | It's not a sarcasm. |
| 3. | @bhatiamanu What is that #Sarcasm ?? I literally read it as scam \xf0\x9f\xa4\xa3. May be bcos of spending too much time on twitter | It's a sarcasm. |
| 4. | b'It\'s said that #sarcasm is the lowest form of #wit. To that I say ""No, really? Hey, somebody give this person a #Nobel #peace #prize! | It's a sarcasm. |
| 5. | b""\xe3\x80\x90New Arrival\xe3\x80\x91 Atolo Pen's Chinese Literary Copy Book just arrival! Most of them are classic poems. Check it out! | It's not a sarcasm |
| 6. | b""@RichLowry Yes and no one died in #PuertoRico after the hurricane and over 500K people didn't die from #COVID19. It\xe2\x80\xa6 https://t.co/MuDoVCNESG" | It's a sarcasm. |
| 7. | @LeaderMcConnell lied, again. Call me  shocked. Republicans whole platform is  gaslighting!\n\nWow. | It's a sarcasm |
| 8. | The #Original #DonutShop #Regular   #MediumRoast #Coffee 18 ct. #Keurig #Kcups   BB 10/21! #bargains #beverages\xe2\x80\xa6   https://t.co/PJaQrqkuAO | It's not a sarcasm. |
| 9. | Surprized everytime how lockdown solves so many problems. #sarcasm \n#NoCovidJETZT BS is trending again. | It's a sarcasm |
| 10. | 79% off #Men #Regular #Fit #Short  #Sleeve #Cotton #T-Shirt \n\nClip the  Extra 79% off Coupon under the listed   Price \nW\xe2\x80\xa6.   https://t.co/Uy8DUKd3c3 | It's not a sarcasm. |
| 11. | @bopinion Unlove for passive turned into  a misguided love for #ARK? Cathie's only  feat has been mimicking this reta  \xe2\x80\xa6 https://t.co/OXlVT8UNV4 | It's a sarcasm |
| 12. | b'Two years of full Regular recruiting and we are still going strong @BritishArmy. With so many roles and opportunit\xe2\x80\xa6 https://t.co/KGMLtOq7fE' | It's not a sarcasm |
| 13. | 100 crore target is not possible sir \nAs   night curfew along with hotel and bar are  closed.\n#MumbaiPolice  to\xe2\x80\xa6  https://t.co/TsejqI3uLJ | It's a sarcasm |
| 14. | b'@BBCSport People, workers having rights.....absolutely not! They should feel lucky to have a job, even it means ser\xe2\x80\xa6 https://t.co/OBGK9uMUEM | It's a sarcasm |
| 15. | I was depressed. He asked me to be happy.   I am not depressed anymore. | It's a sarcasm |

1. **EVALUATION METRICS**

**Accuracy, Precision, recall and f1 - score.these can be caluculated by using Following formulae:**

**Accuracy:** It shows the overall accuracy of the instances which are correctly classified to the total number of the instances. It is calculated by the following formula:

**ACCURACY=TP+TN/TP+TN+P+N**

Where, TP = true positive, TN = true negative, FP = false positive, FN = false negative.

**Precision:** It represents the percentage of relevant searched sarcastic tweets. That is, it measures the amount of tweets categorized as sarcasm against the total

amount of tweets classified as sarcasm. It is calculated by the following formula:

**PRECISION=TP/TP+P**

**Recall:** It represents the percentage of relevant sarcastic tweets that have been searched. That is, against the total amount of sarcastic tweets, measured the

amount of tweets that are normally classified as sarcastic. It is calculated by the following formula:

**RECALL=TP/TP+FN**

Finally, **the F1 score** is a measure of accuracy that can be interpreted as a weighted average of accuracy and recall:

**F1 SCORE=2PRECISON.RECALL/PRECISION+RECALL**

**1)RNN:**

**Training performance metrics**

We have calculated the following metrics

1. Accuracy =0.993511

2.precison=0.993421

3.recall =1.00000

4.f1-score=0.996700

And also we have calculated confusion matrix

[[19 9]

[0 1359]]

Here we can observe that number of true positive =19, number of false postive =9, number of false negative =0 and number of true negative =1359

**Validation performance metrics:-**

We have calculated the following metrics

1. Accuracy = 0.985549

2.precison=0.988270

3.recall =0.997041

4.f1-score=0.992636

And also we have calculated confusion matrix

[[ 4 4]

[ 1 337]]

Here we can observe that number of true positive =4, number of false postive =4, number of false negative =1 and number of true negative =337

**CNN:**

**Training performance metrics**

We have calculated the following metrics

1. Accuracy =0.999279

2.precison= 0.999265

3.recall = 1.000000

4.f1-score=0.999632

And also we have calculated confusion matrix

[[ 27 1]

[ 0 1359]]

Here we can observe that number of true positive =27, number of false postive =1, number of false negative =0 and number of true negative =1359

**Validation performance metrics**

We have calculated the following metrics

1. Accuracy =0.988439

2.precison= 0.991176

3.recall = 0.997041

4.f1-score=0.994100

And also we have calculated confusion matrix

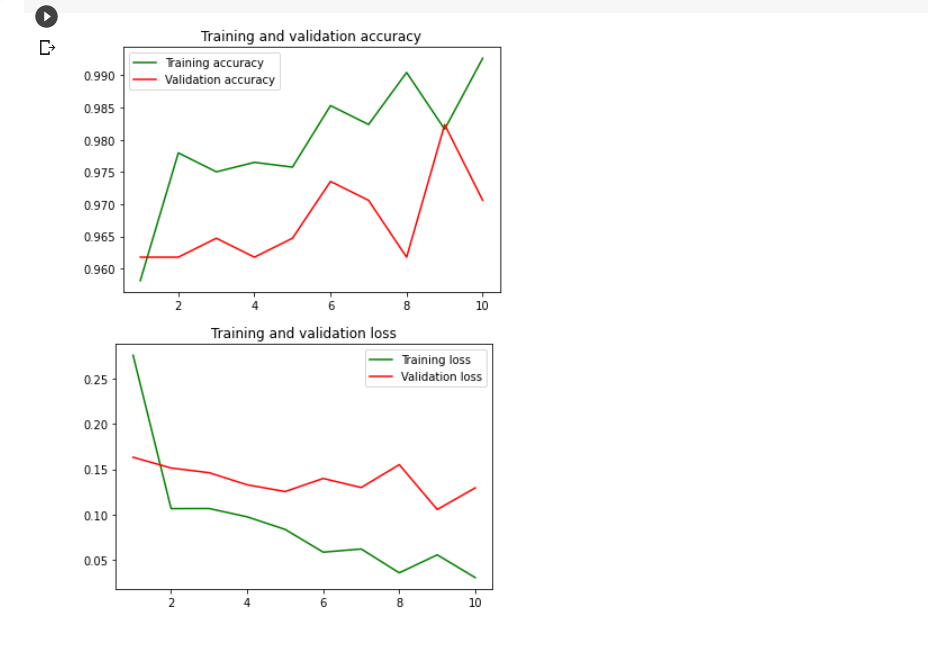
[[ 5 3]

[ 1 337]]

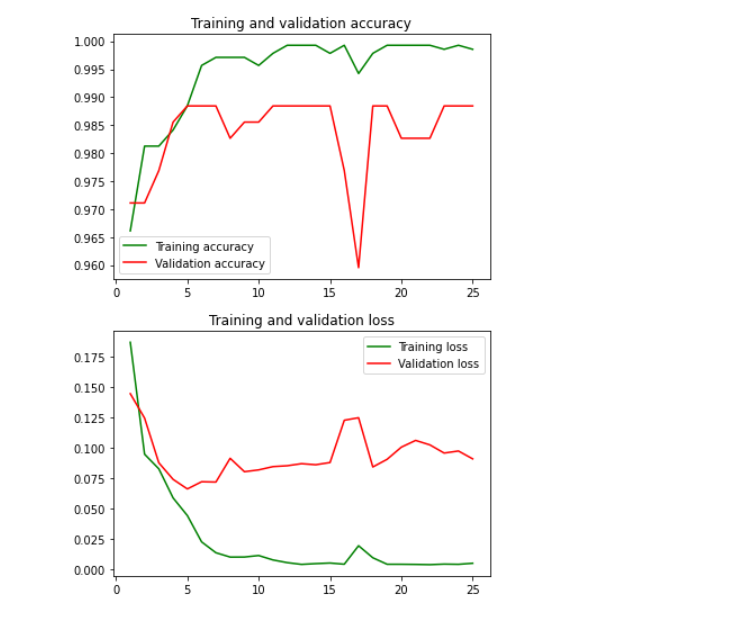
Here we can observe that number of true positive =5, number of false postive =3, number of false negative =1 and number of true negative =337

**GRAPHS**

**ACCURACY & LOSS OF RNN:-**

****

**ACCURACY AND LOSS OF CNN**

****

**11** )**CONCLUSION AND FUTURE DIRECTION**

This model detects the sarcasm in the user's tweets and additionally recommends the best classification method using twitter data .This machine learning analysis can provide the statistical proofs for the best classification.The future direction includes detecting the user's features since we are using user's data as training set. Also the system can be implemented for multi-label data.Using sarcasm detection,the e-commerce websites can make their system more user-personalized.

**12)REFERENCES**

[1]Patrick Adolf Telnoni, Reza Budiawan, Mutia Qana’a.” Comparison of Machine Learning Classification Method on Text-based Case in Twitter”.In 978-1-7281-4880-9/19/$31.00 2019 IEEE

[2]Neha Pawar and Sukhada Bhingarkar.” Machine Learning based Sarcasm Detection on Twitter Data”.In the Fifth International Conference on Communication and Electronics Systems (ICCES 2020) IEEE Conference Record # 48766; IEEE Xplore ISBN: 978-1-7281-5371-1

[3]Azilawati Azizan,Nurul Najwa SK Abdul Jamal,Mohammad Nasir Abdullah,Masurah Mohamad,Nurkhairizan Khairudin."Lexicon-Based Sentiment Analysis for Movie Review Tweets".In 978-1-7281-3041-5/19/$31.00 2019 IEEE

[4] Soujanya Poria ,Erik Cambria ,Devamanyu Hazarika, Prateek Vij” A Deeper Look into Sarcastic Tweets Using Deep Convolutional Neural Networks” Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 1601–1612, Osaka, Japan, December 11-17 2016.