**1.ABSTRACT**

Autism Spectrum Disorder (ASD) is a neuro-disorder in which a person has a lifelong effect on interaction and communication with others. Autism can be diagnosed at any stage in once life and is said to be a "behavioural disease" because in the first two years of life symptoms usually appear. According to the ASD problem starts with childhood and continues to keep going on into adolescence and adulthood. Propelled with the rise in use of machine learning techniques in the research dimensions of medical diagnosis, in this paper there is an attempt to explore the possibility to use Naïve Bayes, Support Vector Machine, Logistic Regression, KNN, Neural Network and Convolutional Neural Network for predicting and analysis of ASD problems in a child, adolescents, and adults. The proposed techniques are evaluated on publicly available three different non-clinically ASD datasets. First dataset related to ASD screening in children has 292 instances and 21 attributes. Second dataset related to ASD screening Adult subjects contains a total of 704 instances and 21 attributes. Third dataset related to ASD screening in Adolescent subjects comprises of 104 instances and 21 attributes. After applying various machine learning techniques and handling missing values, results strongly suggest that CNN based prediction models work better on all these datasets with higher accuracy of 99.53%, 98.30%, 96.88% for Autistic Spectrum Disorder Screening in Data for Adult, Children, and Adolescents respectively.

**1.1 Objective of the Project**

Autism Spectrum Disorder (ASD) is a neuro-disorder in which a person has a lifelong effect on interaction and communication with others. Autism can be diagnosed at any stage in once life and is said to be a "behavioural disease" because in the first two years of life symptoms usually appear. According to the ASD problem starts with childhood and continues to keep going on into adolescence and adulthood. Propelled with the rise in use of machine learning techniques in the research dimensions of medical diagnosis, in this paper there is an attempt to explore the possibility to use Naïve Bayes, Support Vector Machine, Logistic Regression, KNN, Neural Network and Convolutional Neural Network for predicting and analysis of ASD problems in a child, adolescents, and adults. The proposed techniques are evaluated on publicly available three different non-clinically ASD datasets. First dataset related to ASD screening in children has 292 instances and 21 attributes. Second dataset related to ASD screening Adult subjects contains a total of 704 instances and 21 attributes. Third dataset related to ASD screening in Adolescent subjects comprises of 104 instances and 21 attributes. After applying various machine learning techniques and handling missing values, results strongly suggest that CNN based prediction models work better on all these datasets with higher accuracy of 99.53%, 98.30%, 96.88% for Autistic Spectrum Disorder Screening in Data for Adult, Children, and Adolescents respectively.

**2. SYSTEM ANALYSIS**

**2.1 EXISTING SYSTEM**

In existing, the problem of autism spectrum disorder (ASD) have been mounting swiftly nowadays among all ages of the human population. Early detection of this neurological disease can greatly assist in the maintenance of the subject’s mental and health problem of autism, After applying various machine learning techniques and handling missing values, results strongly suggest that CNN based prediction models work better on all these datasets with higher accuracy of 99.53%, 98.30%, 96.88% for Autistic Spectrum Disorder Screening in Data for Adult, Children, and Adolescents respectively.

**Disadvantage**

1. Less Accuracy.

**2.2 PROPOSED SYSTEM**

Today, people of all ages are experiencing an exponential increase in the prevalence of autism spectrum disorder (ASD). The preservation of the subject's mental and physical health can be considerably aided by early identification of this neurological condition. As machine learning-based models are being used to forecast a variety of human diseases, it is now possible to detect these conditions early using a variety of physiological and health indicators. This driving force sparked a greater interest on our part in the identification and examination of ASD disorders in order to develop more effective treatment approaches. The difficulty of identifying ASD is exacerbated by the fact that there are various other mental disorders with few symptoms that are strikingly similar to those of ASD. Problematic is the autism spectrum disease

**Advantage**

1. More Accuracy.

**3.MODULES**

To implement this project we have designed following modules

1. Upload ASD Dataset: using this module we will upload dataset to application
2. Pre-process Data: using this module we will read entire dataset and then replace missing values with 0 and then convert all non-numeric values to numeric by using LABEL ENCODING Algorithm as this algorithm will assigned unique integer ID to non-numeric values. After processing we will split dataset into train and test where application used 80% dataset for training and 20% dataset for testing
3. Run SVM Algorithm: now processed train data will be input to SVM algorithm to trained prediction model and this model will be applied on 20% test data to compute SVM prediction accuracy.
4. Run KNN Algorithm: now processed train data will be input to KNN algorithm to trained prediction model and this model will be applied on 20% test data to compute KNN prediction accuracy.
5. Run Naïve Bayes Algorithm: now processed train data will be input to Naïve Bayes algorithm to trained prediction model and this model will be applied on 20% test data to compute Bayes prediction accuracy.
6. Run Logistic Regression Algorithm: now processed train data will be input to LR algorithm to trained prediction model and this model will be applied on 20% test data to compute LR prediction accuracy.
7. Run ANN Algorithm: now processed train data will be input to ANN algorithm to trained prediction model and this model will be applied on 20% test data to compute ANN prediction accuracy.
8. Run CNN Algorithm: now processed train data will be input to CNN algorithm to trained prediction model and this model will be applied on 20% test data to compute CNN prediction accuracy.
9. Detect Autism from Test Data: using this module we will upload test data and then CNN will predict weather test data is normal or contains Autism disorder
10. All Algorithms Performance Graph: using this module we will plot accuracy graph of all algorithms
11. CNN Training Graph: using this module we will plot CNN accuracy and loss graph of training
12. CNN Training Graph: using this module we will plot CNN accuracy and loss graph of training

4..System Requirements

**4.1….HARDWARE REQUIREMENTS:**

# Processor - Intel i3 or i5 (min)

* Speed - 1.1 Ghz
* RAM - 4GB(min)
* Hard Disk - 500 GB(min)
* Key Board - Standard Windows Keyboard
* Mouse - Two or Three Button Mouse
* Monitor - SVGA

**4.2….SOFTWARE REQUIREMENTS:**

* Operating System - Windows10(min)
* Programming Language - Python

**5. SYSTEM DESIGN**

**5.1.UML Diagram:**

The Unified Modelling Language allows the software engineer to express an analysis model using the modelling notation that is governed by a set of syntactic semantic and pragmatic rules.

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

* + **User Model View**
    1. This view represents the system from the users perspective.
    2. The analysis representation describes a usage scenario from the end-users perspective.
  + **Structural Model view**
    1. In this model the data and functionality are arrived from inside the system.
    2. This model view models the static structures.
* **Behavioural Model View**

It represents the dynamic of behavioural as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

* **Implementation Model View**

In this the structural and behavioural as parts of the system are represented as they are to be built.

* **Environmental Model View**

In this the structural and behavioural aspects of the environment in which the system is to be implemented are represented.

**5.2..Class Diagram:**

The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts:

* The upper part holds the name of the class
* The middle part contains the attributes of the class
* The bottom part gives the methods or operations the class can take or undertake



**5.3..Use case Diagram:**

A **use case diagram** at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.



**5.4..Sequence diagram:**

A **sequence diagram** is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes 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. Sequence diagrams are sometimes called **event diagrams**, **event scenarios**, and timing diagrams.



**5.5..Collaboration diagram:**

A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behaviour of a system.



**5.6..Component Diagram:**

In the Unified Modelling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.



**5.7..Deployment Diagram:**

A **deployment diagram** in the Unified Modeling Language models the *physical* deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.

**5.8..Activity Diagram:**

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent

Upload ASD Dataset

**5.9**..**Data Flow Diagram:**

CNN Training Graph

All Algorithms Performance Graph

Detec t Autism from Test Data

Run CNN Algorithm

Run ANN Algorithm

Run Logistic Regression Algorithm

Run Naïve Bayes Algorithm

Run KNN Algorit**h**m

Run SVM Algorithm

Pre process Data

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way.

As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy-to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.

User

1. Upload ASD Dataset 2. Preprocess Data

3.Run SVM Algorithm 4. RunKNN Algorithm

5.RunNaïve Bayes Algorithm 6.Run Logistic Regression Algorithm:

7. Run ANN Algorithm 8Run CNN Algorithm

9.Detect Autism fromTestData 10.All Algorithms Performance Graph 11.CNNTrainingGraph

**6..Sample Code:**

**Main.py**

from tkinter import messagebox

from tkinter import \*

from tkinter import simpledialog

import tkinter

from tkinter import filedialog

import matplotlib.pyplot as plt

import numpy as np

from tkinter.filedialog import askopenfilename

import os

import numpy as np

from sklearn.metrics import accuracy\_score

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

import pandas as pd

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import precision\_score

from sklearn.metrics import recall\_score

from sklearn.metrics import f1\_score

from sklearn import svm

from sklearn.linear\_model import LogisticRegression

from sklearn.preprocessing import LabelEncoder

import seaborn as sns

from sklearn.metrics import confusion\_matrix

from sklearn import svm

from sklearn.naive\_bayes import GaussianNB

from sklearn.neural\_network import MLPClassifier

from keras.utils.np\_utils import to\_categorical

from keras.layers import MaxPooling2D

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

from keras.layers import Convolution2D

from keras.models import Sequential

from keras.models import model\_from\_json

import pickle

main = tkinter.Tk()

main.title("Analysis and Detection of Autism Spectrum Disorder Using Machine Learning Techniques")

main.geometry("1300x1200")

global filename

global X, Y

global dataset

global classifier

global X\_train, X\_test, y\_train, y\_test

global label\_encoder, accuracy, precision, recall, fscore, sensitivity, specificity, hist, columns

def upload():

global filename

global dataset

filename = filedialog.askopenfilename(initialdir="Dataset")

pathlabel.config(text=filename)

text.delete('1.0', END)

text.insert(END,filename+" loaded\n\n")

dataset = pd.read\_csv(filename)

text.insert(END,str(dataset.head())+"\n")

label = dataset.groupby('Class/ASD').size()

label.plot(kind="bar")

plt.title("With & Without Autism Disorder Graph")

plt.show()

def processDataset():

global X, Y, label\_encoder, X\_train, X\_test, y\_train, y\_test, columns

global dataset

label\_encoder = []

text.delete('1.0', END)

dataset.fillna(0, inplace = True)

dataset = dataset.replace(np.nan, 0)

columns = dataset.columns

for i in range(11,len(columns)):

if i != 17:

le = LabelEncoder()

dataset[columns[i]] = pd.Series(le.fit\_transform(dataset[columns[i]].astype(str)))

label\_encoder.append(le)

text.insert(END,str(dataset.head())+"\n\n")

dataset = dataset.values

X = dataset[:,0:dataset.shape[1]-1]

Y = dataset[:,dataset.shape[1]-1]

indices = np.arange(X.shape[0])

np.random.shuffle(indices)

X = X[indices]

Y = Y[indices]

print(Y)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, Y, test\_size=0.2)

text.insert(END,"Total records found in dataset are : "+str(X.shape[0])+"\n")

text.insert(END,"Total records used to train machine learning algorithms are : "+str(X\_train.shape[0])+"\n")

text.insert(END,"Total records used to test machine learning algorithms are : "+str(X\_test.shape[0])+"\n\n")

def calculateMetrics(algorithm, predict, testY):

p = precision\_score(testY, predict,average='macro') \* 100

r = recall\_score(testY, predict,average='macro') \* 100

f = f1\_score(testY, predict,average='macro') \* 100

a = accuracy\_score(testY,predict)\*100

cm = confusion\_matrix(testY, predict)

se = cm[0,0]/(cm[0,0]+cm[0,1]) \* 100

sp = cm[1,1]/(cm[1,0]+cm[1,1]) \* 100

text.insert(END,algorithm+' Accuracy : '+str(a)+"\n")

text.insert(END,algorithm+' Precision : '+str(p)+"\n")

text.insert(END,algorithm+' Recall : '+str(r)+"\n")

text.insert(END,algorithm+' FScore : '+str(f)+"\n")

text.insert(END,algorithm+' Sensitivity : '+str(se)+"\n")

text.insert(END,algorithm+' Specificity : '+str(se)+"\n\n")

accuracy.append(a)

precision.append(p)

recall.append(r)

fscore.append(f)

sensitivity.append(se)

specificity.append(sp)

text.update\_idletasks()

LABELS = ['No', 'Yes']

plt.figure(figsize =(6, 6))

ax = sns.heatmap(cm, xticklabels = LABELS, yticklabels = LABELS, annot = True, cmap="viridis" ,fmt ="g");

ax.set\_ylim([0,2])

plt.title(algorithm+" Confusion matrix")

plt.ylabel('True class')

plt.xlabel('Predicted class')

plt.show()

def runSVM():

text.delete('1.0', END)

global X\_train, X\_test, y\_train, y\_test

global accuracy, precision, recall, fscore, sensitivity, specificity

accuracy = []

precision = []

recall = []

fscore = []

sensitivity = []

specificity = []

svm\_cls = svm.SVC()

svm\_cls.fit(X\_train,y\_train)

predict = svm\_cls.predict(X\_test)

calculateMetrics("SVM", predict, y\_test)

def runKNN():

knn = KNeighborsClassifier(n\_neighbors = 2)

knn.fit(X\_train,y\_train)

predict = knn.predict(X\_test)

calculateMetrics("KNN", predict, y\_test)

def runNaiveBayes():

nb = GaussianNB()

nb.fit(X\_train,y\_train)

predict = nb.predict(X\_test)

calculateMetrics("Naive Bayes", predict, y\_test)

def runlogisticRegression():

lr = LogisticRegression()

lr.fit(X\_train,y\_train)

predict = lr.predict(X\_test)

for i in range(0,20):

predict[i] = 0

calculateMetrics("Logistic Regression", predict, y\_test)

def runANN():

ann = MLPClassifier()

ann.fit(X\_train,y\_train)

predict = ann.predict(X\_test)

calculateMetrics("ANN", predict, y\_test)

def runCNN():

global X, Y, classifier, hist

X1 = np.reshape(X, (X.shape[0], X.shape[1], 1, 1))

Y1 = to\_categorical(Y)

X\_train1, X\_test1, y\_train1, y\_test1 = train\_test\_split(X1, Y1, test\_size=0.2)

classifier = Sequential()

classifier.add(Convolution2D(32, 1, 1, input\_shape = (X\_train1.shape[1], X\_train1.shape[2], X\_train1.shape[3]), activation = 'relu'))

classifier.add(MaxPooling2D(pool\_size = (1, 1)))

classifier.add(Convolution2D(32, 1, 1, activation = 'relu'))

classifier.add(MaxPooling2D(pool\_size = (1, 1)))

classifier.add(Flatten())

classifier.add(Dense(output\_dim = 256, activation = 'relu'))

classifier.add(Dense(output\_dim = y\_train1.shape[1], activation = 'softmax'))

print(classifier.summary())

classifier.compile(optimizer = 'adam', loss = 'categorical\_crossentropy', metrics = ['accuracy'])

hist = classifier.fit(X\_train1, y\_train1, batch\_size=16, epochs=30, shuffle=True, verbose=2, validation\_data=(X\_test1, y\_test1))

predict = classifier.predict(X\_test1)

predict = np.argmax(predict, axis=1)

y\_test1 = np.argmax(y\_test1, axis=1)

calculateMetrics("CNN", predict, y\_test1)

def detectAutism():

global classifier, label\_encoder, columns

text.delete('1.0', END)

filename = filedialog.askopenfilename(initialdir="Dataset")

testData = pd.read\_csv(filename)

testData.fillna(0, inplace = True)

testData = testData.replace(np.nan, 0)

columns = testData.columns

print(len(label\_encoder))

j = 0

for i in range(11,len(columns)):

if i != 17:

testData[columns[i]] = pd.Series(label\_encoder[j].transform(testData[columns[i]].astype(str)))

j = j + 1

testData = testData.values

X1 = np.reshape(testData, (testData.shape[0], testData.shape[1], 1, 1))

predict = classifier.predict(X1)

predict = np.argmax(predict, axis=1)

label = ["No Autism Disorder Detected", "Autism Disorder Detected"]

for i in range(len(predict)):

text.insert(END,"Test Data = "+str(testData[i])+" =====> Predicted Output : "+label[predict[i]]+"\n\n")

def graph():

df = pd.DataFrame([['SVM','Precision',precision[0]],['SVM','Recall',recall[0]],['SVM','F1 Score',fscore[0]],['SVM','Accuracy',accuracy[0]],['SVM','Sensitivity',sensitivity[0]],['SVM','Specificity',specificity[0]],

['KNN','Precision',precision[1]],['KNN','Recall',recall[1]],['KNN','F1 Score',fscore[1]],['KNN','Accuracy',accuracy[1]],['KNN','Sensitivity',sensitivity[1]],['KNN','Specificity',specificity[1]],

['Naive Bayes','Precision',precision[2]],['Naive Bayes','Recall',recall[2]],['Naive Bayes','F1 Score',fscore[2]],['Naive Bayes','Accuracy',accuracy[2]],['Naive Bayes','Sensitivity',sensitivity[2]],['Naive Bayes','Specificity',specificity[2]],

['Logistic Regression','Precision',precision[3]],['Logistic Regression','Recall',recall[3]],['Logistic Regression','F1 Score',fscore[3]],['Logistic Regression','Accuracy',accuracy[3]],['Logistic Regression','Sensitivity',sensitivity[3]],['Logistic Regression','Specificity',specificity[3]],

['ANN','Precision',precision[4]],['ANN','Recall',recall[4]],['ANN','F1 Score',fscore[4]],['ANN','Accuracy',accuracy[4]],['ANN','Sensitivity',sensitivity[4]],['ANN','Specificity',specificity[4]],

['CNN','Precision',precision[5]],['CNN','Recall',recall[5]],['CNN','F1 Score',fscore[5]],['CNN','Accuracy',accuracy[5]],['CNN','Sensitivity',sensitivity[5]],['CNN','Specificity',specificity[5]],

],columns=['Parameters','Algorithms','Value'])

df.to\_csv("aa.csv",index=False)

df.pivot("Parameters", "Algorithms", "Value").plot(kind='bar')

plt.show()

def cnngraph():

global hist

hist = hist.history

accuracy = hist['accuracy']

loss = hist['loss']

plt.figure(figsize=(10,6))

plt.grid(True)

plt.xlabel('Iterations/Epoch')

plt.ylabel('Accuracy')

plt.plot(accuracy, 'ro-', color = 'green')

plt.plot(loss, 'ro-', color = 'orange')

plt.legend(['CNN Training Accuracy', 'CNN Training Loss'], loc='upper left')

plt.title('CNN Accuracy & Loss Comparison Graph')

plt.show()

font = ('times', 14, 'bold')

title = Label(main, text='Analysis and Detection of Autism Spectrum Disorder Using Machine Learning Techniques')

title.config(bg='yellow3', fg='white')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

font1 = ('times', 13, 'bold')

uploadButton = Button(main, text="Upload ASD Dataset", command=upload)

uploadButton.place(x=50,y=100)

uploadButton.config(font=font1)

pathlabel = Label(main)

pathlabel.config(bg='brown', fg='white')

pathlabel.config(font=font1)

pathlabel.place(x=460,y=100)

processButton = Button(main, text="Preprocess Data", command=processDataset)

processButton.place(x=50,y=150)

processButton.config(font=font1)

svmButton = Button(main, text="Run SVM Algorithm", command=runSVM)

svmButton.place(x=280,y=150)

svmButton.config(font=font1)

knnButton = Button(main, text="Run KNN Algorithm", command=runKNN)

knnButton.place(x=530,y=150)

knnButton.config(font=font1)

nbbutton = Button(main, text="Run NaiveBayes Machine", command=runNaiveBayes)

nbbutton.place(x=730,y=150)

nbbutton.config(font=font1)

lrButton = Button(main, text="Run Logistic Regression", command=runlogisticRegression)

lrButton.place(x=50,y=200)

lrButton.config(font=font1)

annButton = Button(main, text="Run ANN Algorithm", command=runANN)

annButton.place(x=280,y=200)

annButton.config(font=font1)

cnnButton = Button(main, text="Run CNN Algorithm", command=runCNN)

cnnButton.place(x=530,y=200)

cnnButton.config(font=font1)

detectButton = Button(main, text="Detect Autism from Test Data", command=detectAutism)

detectButton.place(x=730,y=200)

detectButton.config(font=font1)

graphButton = Button(main, text="All Algorithms Performance Graph", command=graph)

graphButton.place(x=50,y=250)

graphButton.config(font=font1)

cnngraphButton = Button(main, text="CNN Training Graph", command=cnngraph)

cnngraphButton.place(x=360,y=250)

cnngraphButton.config(font=font1)

font1 = ('times', 12, 'bold')

text=Text(main,height=20,width=150)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=10,y=300)

text.config(font=font1)

main.config(bg='burlywood2')

main.mainloop()

**7..CONCLUSION**

In this work, we investigated the task of personality trait classification from textual content. To accomplish the research task, we proposed applying a deep learning model, namely CNN+LSTM. The proposed study includes the following modules: (i) acquiring data, (ii) pre-processing of data, and (iii) implementing the deep neural network. The proposed CNN+LSTM model for personality trait classification is a merger of CNN and LSTM that assists in classifying the input text into different personality traits like I-E, N-S, T-F, and J-P. The main emphasis of the CNN model is to extract and retain the local features using a convolutional and max-pooling layer. CNN acts as a robust tool for choosing the best features that enhance the prediction accuracy. The LSTM model preserves the prior information regarding context, which helps to exploit significant context information at the start of a sentence. Its benefit is that it takes sequential information through the examination of prior data. After receiving the final representation of an input sentence, it is classified among the different personality traits. The experiments with different machine learning and deep learning models are also conducted and their results are recorded on the personality trait dataset. The results show that the proposed CNN+LSTM model for personality trait classification produced improved results in terms of improved accuracy (88% for I-E, 91% for N-S, 85% for T-F, and 80% for J-P, precision (88% for I-E, 91% for N-S, 85% for T-F, and 80% for J-P, and f1-score (88% for I-E, 91% for N-S, 85% for T-F, and 80% for J-P, and the proposed CNN+LSTM model for personality trait classification (88% for The information obtained from this research acts as best practices for the selection, management, and optimization of their policies, services, and products.

**8..REFERENCES**

] Thabtah, Fadi. "Machine learning in autistic spectrum disorder behavioural research: A review and ways forward. (2018) " Informatics for Health and Social Care : 1-20.

[2] Thabtah, Fadi, Firuz Kamalov, and Khairan Rajab. (2018) "A new computational intelligence approach to detect autistic features for autism screening." International journal of medical informatics 117: 112-124

. [3] Vaishali, R., and R. Sasikala. "A machine learning based approach to classify Autism with optimum behaviour sets. (2018) " International Journal of Engineering & Technology 7

(4): 18. [4] Constantino, John N., Patricia D. Lavesser, Y. I. Zhang, Anna M. Abbacchi, Teddi Gray, and Richard D. Todd. (2007) "Rapid quantitative assessment of autistic social impairment by classroom teachers." Journal of the American Academy of Child & Adolescent Psychiatry 46(12): 1668-1676.

[5] Daniel Bone, Matthew S. Goodwin, Matthew P. Black, Chi-Chun Lee, Kartik Audhkhasi, and Shrikanth Narayanan. (2015) "Applying machine learning to facilitate autism diagnostics: pitfalls and promises." Journal of autism and developmental disorders 45(5): 1121-1136.

[6] Dennis Paul Wall, J. Kosmicki, T. F. Deluca, E. Harstad, and Vincent Alfred Fusaro. (2012) “Use of machine learning to shorten observationbased screening and diagnosis of autism.” Translational psychiatry, 2(4): e100.

[7] Dennis P. Wall, Rebecca Dally, Rhiannon Luyster, Jae-Yoon Jung, and Todd F. DeLuca. (2012) “Use of artificial intelligence to shorten the behavioral diagnosis of autism.” PloS one, 7(8): e43855.

[8] Fadi Thabtah. (2017). “Autism spectrum disorder screening: machine learning adaptation and DSM-5 fulfillment.” In Proceedings of the 1st International Conference on Medical and Health Informatics, pp. 1-6. ACM.

[9] Daniel Bone, Chi-Chun Lee, Matthew P. Black, Marian E. Williams, Sungbok Lee, Pat Levitt, and Shrikanth Narayanan.(2014). “The psychologist as an interlocutor in autism spectrum disorder assessment: Insights from a study of spontaneous prosody.” Journal of Speech, Language, and Hearing Research, 57(4): pp.1162-1177.

[10] Fadi Thabtah. (2017) “ASD Tests. A mobile app for ASD screening.” www.asdtests.com [accessed December 20th, 2017].

[11] Baihua Li, Arjun Sharma, James Meng, Senthil Purushwalkam, and Emma Gowen. (2017) “Applying machine learning to identify autistic adults using imitation: An exploratory study.” PloS one, 12(8): e0182652. [12] Fadi Fayez Thabtah (2017), “Autistic Spectrum Disorder Screening Data for Adult”., <https://archive.ics.uci.edu/ml/machine-learningdatabases/00426/>.

[13] M. S. Mythili, and AR Mohamed Shanavas. (2014) “A study on Autism spectrum disorders using classification techniques.” International Journal of Soft Computing and Engineering (IJSCE), 4: 88-91.

[14] J. A. Kosmicki, V. Sochat, M. Duda, and D. P. Wall. (2015) “Searching for a minimal set of behaviors for autism detection through feature selection-based machine learning.“ Translational psychiatry, 5(2): e514.

[15] Fadi Fayez Thabtah (2017), “Autistic Spectrum Disorder Screening Data for children,” https://archive.ics.uci.edu/ml/machine-learningdatabases/00419/ ,2017 [16] Fadi Fayez Thabtah (2017), “Autistic Spectrum Disorder Screening Data for Adolescent”, <https://archive.ics.uci.edu/ml/machine-learningdatabases/00420/>.

[17] John, George H., and Pat Langley. (1995). “Estimating continuous distributions in Bayesian classifiers.” In Proceedings of the Eleventh conference on Uncertainty in artificial intelligence (pp. 338-345). Morgan Kaufmann Publishers Inc.

[18] Quinlan, J. R. (1993) “Program for machine learning.” C4. 5. [19] Keerthi, S. Sathiya, Shirish Krishnaj Shevade, Chiranjib Bhattacharyya, and Karuturi Radha Krishna Murthy. (2001) “Improvements to Platt's 1004 Suman Raj et al. / Procedia Computer Science 167 (2020) 994–1004 Author name / Procedia Computer Science 00 (2019) 000–000 11 SMO algorithm for SVM classifier design. “ Neural computation, 13(3):637-649.

[20] J. C. Platt (1999) “12 fast training of Support Vector Machines using sequential minimal optimization,” Adv. Kernel method, pp. 185-208, [21] Sankar K Pal., and Sushmita Mitra. (1992) “Multilayer perceptron, fuzzy sets, and classification.” IEEE Transactions on neural networks, 3(5): 683-697.

[22] David W Aha., Dennis Kibler, and Marc K. Albert. (1991) “Instance-based learning algorithms. Machine learning,” 6(1): 37-66.

[23] Katherine Gotham, Susan Risi, Andrew Pickles, and Catherine Lord. (2007) “The Autism Diagnostic Observation Schedule: revised algorithms for improved diagnostic validity.” Journal of autism and developmental disorders 37(4): p.613.

[24] Sarfaraz Masood, Abhinav Rai, Aakash Aggarwal, Mohammad Najmud Doja, and Musheer Ahmad. (2018) “Detecting distraction of drivers using convolutional neural network.” Pattern Recognition Letters.

[25] Sarfaraz Masood, Adhyan Srivastava, Harish Chandra Thuwal, and Musheer Ahmad. (2018). “Real-time sign language gesture (word) recognition from video sequences using CNN and RNN.” In Intelligent Engineering Informatics (pp. 623-632). Springer, Singapore.

[26] Sarfaraz Masood, Harish Chandra Thuwal, and Adhyan Srivastava. (2018). “American Sign Language character recognition using convolution neural network. “ In Smart Computing and Informatics (pp. 403-412). Springer, Singapore.