**PREDICTION OF PARKINSON’S DISEASE USING SPEECH SIGNALS**

**A PROJECT REPORT**

***Submitted by***

**HARSHINI C 211418104082**

**KEERTHIKA S 211418104121**

**MADHUMITHA B D 211418104141**

***in partial fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING**

***in***

**COMPUTER SCIENCE AND ENGINEERING**



**PANIMALAR ENGINEERING COLLEGE**

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

**MAY 2022**

**PANIMALAR ENGINEERING COLLEGE**

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

**BONAFIDE CERTIFICATE**

Certified that this project report **“PREDICTION OF PARKINSON’S DISEASE USING SPEECH SIGNALS”** is the bonafide work of “**HARSHINI C (211418104082), KEERTHIKA S (21148104121)** and **MADHUMITHA B D (211418104141)** “who carried out the project work under my supervision.

**SIGNATURE SIGNATURE**

**Dr.S. MURUGAVALLI, M.E., Ph.D.., Mrs.P. DEEPA, M.E.,**

**HEAD OF THE DEPARTMENT SUPERVISOR**

**ASSOCIATE PROFESSOR**

DEPARTMENT OF CSE, DEPARTMENT OF CSE,

PANIMALAR ENGINEERING COLLEGE, PANIMALAR ENGINEERING COLLEGE,

NASARATHPETTAI, NASARATHPETTAI,

POONAMALLEE, POONAMALLEE,

CHENNAI-600 123. CHENNAI-600 123.

Certified that the above mentioned students were examined in End Semester project viva-voice held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**DECLARATION BY THE STUDENT**

We HARSHINI C (211418104082), KEERTHIKA S (211418104121), MADHUMITHA B. D (211418104141) hereby declare that this project report titled “PREDICTION OF PARKINSON’S DISEASE USING SPEECH SIGNALS”, under the guidance of Mrs.P. DEEPA is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

**C.HARSHINI**

**S. KEERTHIKA**

**B.D. MADHUMITHA**

**ACKNOWLEDGEMENT**

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our Directors **Tmt.C.VIJAYARAJESWARI**, **Dr.C.SAKTHI KUMAR,M.E.,Ph.D** and **Dr.SARANYASREE SAKTHI KUMAR B.E.,M.B.A.,Ph.D.,** for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr.K. MANI M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr. S.MURUGAVALLI , M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank my **Project Co-Ordinator** **Dr.L. JABASHEELA** **M.E., Ph.D.,** **Project Guide** **Mrs.P. DEEPA M.E.,** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

**HARSHINI C**

**KEERTHIKA S**

**MADHUMITHA B.D**

**ABSTRACT**

Parkinson Disease (PD) is a neurodegenerative dis- order. Numerous common symptoms which may or may not indicate that case is suffering from Parkinson Disease. In this project a new standing scale has been introduced which helps to examine the position of Parkinson Disease but isn't obligatory that a person having analogous symptoms may surely suffer from Parkinson Disease. PD is an unsolved problem till date hence the project focuses on applicable features, medicines and common ways used to descry or dissect PD. To overcome similar problem different ways will be used to study and dissect the early discovery of PD. It can be anatomized with the help of deep understanding of Parkinson Disease. Still presence of some common symptoms has not yet been described up to the mark to dissect the position of Parkinson Disease. Hence, it's veritably grueling to descry early stage of Parkinson Disease. In project, work focuses on only verified symptoms of Parkinson Disease which doesn’t deals to any other disease fully.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT** | iii |
|  | **LIST OF FIGURES** | vi |
| **1.** | **INTRODUCTION** | 1 |
|  | 1.1 Overview | 2 |
|  | 1.2 Problem Definition | 4 |
| **2.** | **LITERATURE SURVEY** | 5 |
| 3. | **SYSTEM ANALYSIS** | 8 |
|  | 3.1 Existing System | 9 |
|  | 3.2 Proposed system | 9 |
|  | 3.3 Feasibility Study | 10 |
|  | 3.4 Hardware Environment | 14 |
|  | 3.5 Software Environment | 14 |
| **4.** | **SYSTEM DESIGN** | 15 |
|  | 4.1. ER diagram | 16 |
|  | 4.2 Data Flow Diagram | 17 |
|  | 4.3 UML Diagrams | 18 |
| **5.** | **SYSTEM ARCHITECTURE** | 24 |
|  | 5.1 Module Design Specification | 25 |
|  | 5.2 Algorithms | 28 |
| **6.** | **SYSTEM IMPLEMENTATION** | 29 |
|  | 6.1 Client-side coding | 30 |
|  | 6.2 Server-side coding | 34 |
| **7.** | **RESULTS AND DISCUSSION** | 50 |
| **8.** | **CONCLUSION** | 54 |
|  | 8.1 Conclusion and Future Enhancements | 55 |
|  | **APPENDICES** | 56 |
|  | A.1 Sample Screens | 57 |
|  | **REFERENCES** | 60 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGNO** | **NAMES** | **PAGE NO** |
| **4.1** | ER DIAGRAM | **16** |
| **4.2** | DATAFLOW DIAGRAM | **17** |
| **4.3** | USECASE DIAGRAM | **18** |
| **4.4** | CLASS DIAGRAM | **19** |
| **4.5** | SEQUENCE DIAGRAM | **20** |
| **4.6** | ACTIVITY DIAGRAM | **21** |
| **4.7** | STATE CHART DIAGRAM | **22** |
| **4.8** | COMPONENT DIAGRAM | **22** |
| **4.9** | DEPLOYMENT DIAGRAM | **23** |
| **A.1** | HOME PAGE | **57** |
| **A.2** | UPLOAD AUDIO RESULT1 | **57** |
| **A.3** | UPLOAD AUDIO RESULT2 | **58** |
| **A.4** | LIVE AUDIO RESULT | **58** |
| **A.5** | LIVE AUDIO VALUES | **59** |
| **A.6** | UPLOAD AUDIO VALUES | **59** |

**CHAPTER 1**

**INTRODUCTION**

**CHAPTER 1**

**INTRODUCTION**

* 1. **OVERVIEW**

Parkinson’s disorder (PD) manifests as the death of dopaminergic neurons in the substantia nigra pars compacta within the midbrain. This neurodegeneration leads to a range of symptoms including cooperation issues, bradykinesia, oral changes, and severity. Dysarthria is also observed in PD cases; it's characterized by weakness, palsy, and lack of cooperation in the motor- speech system affecting respiration, phonation, articulation, and prosody. Since symptoms and the disorder course vary, PD is constantly not diagnosed for multiple times. Thus, there's a need for more sensitive individual tools for PD discovery because, as the disorder progresses, further symptoms arise that make PD harder to treat. The main shortages of PD speech are loss of intensity, monotone of pitch and loudness, reduced stress, unfit silences, short rushes of speech, variable rate, squishy consonant articulation, and harsh and breathy voice (dysphonia). The range of voice related symptoms is promising for a possible discovery tool because recording voice data is non-invasive and can be done smoothly with mobile bias. PD is problematic to determine prematurely due to the subtle first symptoms. There's a significant burden to cases and the health care system due to holding patterns in decision. The difficulty in early PD decision has inspired investigators to develop webbing tools calculating on automated algorithms to separate healthy controls from people with PD. The approach used for collecting the audio data was a microphone, voice exercise that recorded actors articulating for 10 seconds. the symptoms of PD are downgraded by the use of dopaminergic specifics similar as levodopa. During data collection, cases were asked to give information regarding when, relative to taking drug, they handed their data. The options included Just after Parkinson drug (at your stylish), Another time, directly before Parkinson drug, don’t take Parkinson specifics, and no value. These drug time points were interpreted to mean time of stylish symptom control, on drug but not directly before or after, time of worst symptoms, not on specifics, and not applicable, individually. This information, crossed with the clinical decision responses from the demographics check led to three groups of cases and data. Cases that had medicine former to the voice test were not used as actors in the analysis. The explanation for this parameter selection is that the voice of the case will depict the most extreme goods of the PD without the effect of any drug. The supposition is that the voice features will be noticeably different from those of the controls. The control in this trial is a party who has not been professionally diagnosed with PD. Each case could contribute to multiple voice surrenders, so the number of unique audio lines exceeds the total number of cases surveyed. predicated on the data uprooted from these studies, a csv train was created that contained the demographics data linked with the health constitutions unique to each case. An effective network process, particularly one that doesn’t want a clinic visit, would be helpful. Since PD cases expose characteristic oral features, voice recordings are a useful and non-invasive tool for decision. Still, this would be an effective webbing step prior to an appointment with a clinician, if deep learning algorithms could be applied to a voice recording dataset to directly decision PD.

**1.2 PROBLEM DEFINITION**

Prompt Opinion of PD is important in order to give cases with applicable treatment and information on prognostic. Still, an accurate early opinion can be grueling because the movement symptoms can lap with other conditions. Doctors make the decision of PD predicated on clinical evaluation, interpreting information gained generally through history-taking and examination of the case. In this project, we describe the training and selection of the deep literacy model. Unlike earlier work in this area. we concentrate on developing a model that can diagnose Parkinson’s complaint from voice. Also, of significance from a clinical perspective, we show that PD can be diagnosed using an purposely simple CNN model. Simple models are more likely to generalize beyond their training data and hence are considered more secure for medical opinion.

**CHAPTER 2**

**LITERATURE SURVEY**

**CHAPTER 2**

**LITERATURE SURVEY**

A lot of research has been done to predict Parkinson’s disease in a patient, but less work has been reported to predict its severity. These works have used various machine learning techniques, deep learning techniques and various other techniques to predict the Parkinson disease. In a survey by Christos Laganas.[1] on the Parkinson’s Disease Detection Based on Running Speech Data from Phone Calls, a machine learning-based approach was considered to be the better approach for voice-based smartphone data from subjects’ running speech recordings are used for predicting the PD severity. Apoorva Safai.[2] on Multi-modal brain-based prediction of Parkinson disease using Graph Attention Networks, GAT (GRAPH ATTENTION NETWORK) model which is used to generate node embeddings from the structural connectivity matrix and multimodal feature. PubMed Journal.[3] on Vocal Feature Extraction based Artificial Intelligence model for Parkinson disease detection, two hybrid models based on SVM integrating with principal component analysis and a sparse auto-encoder are proposed based on vocal features for the detection of the Parkinson disease. Anusri [4] on An Early Prediction of Parkinson’s disease using facial emotional Recognition, where facial emotions are classified using CNN architecture for the detection of Parkinson disease. Ahlem Kehili.[5] on Early Parkinson detection using fully connected deep neural network based on vocal features, vocal features were proposed using fully connected deep neural network as a classifier to detect Parkinson disease in person. Srivardhini Veeraragavan.[6] on Parkinson’s Disease Diagnosis and Severity Assessment Using Ground Reaction Forces and Neural Networks, in which Gait features are extracted and selected to use as training features for the Artificial Neural Network (ANN) model to diagnose PD using cross validation. F.M. Javed Mehedi Shamrat.[7] on A Comparative Analysis of Parkinson Disease Prediction Using Machine Learning Approaches, in which AI methods are used for the detection of Parkinson disease datasets using SVM, KNN, and LR. Amit Kumar Patra.[8] on Prediction of Parkinson’s disease using Ensemble Machine Learning classification from acoustic analysis, Decision Trees, Logistic Regression and K-nearest neighbors’ techniques are used for the detection of the Parkinson disease. David J. Brooks.[9] on Imaging Approaches to Parkinson Disease, in which Structural changes in PD nigra can be detected with both transcranial sonography and diffusion tensor MRI. Srishti Grover.[10] on Predicting severity of Parkinson disease using deep learning, prediction of the Parkinson disease is performed using the deep learning technique in which the collected data is normalized using min-max normalization.

**CHAPTER 3**

**SYSTEM ANALYSIS**

**CHAPTER 3**

**SYSTEM ANALYSIS**

**3.1 EXISTING SYSTEM**

In Existing system, prognosis of Parkinson’s disorder executed using Machine learning approach similar as image class, analysis predicated on recorded voice. But doesn't deliver exactness, preciseness, recall etc. Neuromelanin sensitive magnetic resonance imaging (NMS-MRI) has been vital in relating abnormalities in the substantia nigra pars compacta (SNc) in Parkinson's disorder (PD) as PD is characterized by loss of dopaminergic neurons in the SNc. Current ways employ estimation of discrepancy rates of the SNc, imaged on NMS-MRI, to discern PD cases from the healthy controls. Still, the origin of these features is time- consuming and laborious and also provides lower prognosis perfection. Likewise, these don't consider for patterns of subtle changes in PD in the SNc.

**DISADVANTAGES**

* They're using complex Speech data.
* accurateness, Recall F1 score criteria aren't calculated
* They aren't calculating the Accuracy.
* They Don’t locate as an operation.

**3.2 PROPOSED SYSTEM**

In this project a deep learning-based approach for voice-based data from subjects’ running speech recordings is presented. These showcase that PD can be detected from voice recordings and features that were already validated. Here, subjects were clinically examined, showing that the extracted metrics can discriminate the early PD from the healthy controls and can be further used in the everyday life for the monitoring of voice degradation. Used for evaluation and monitoring of motor impairments in PD patients through task-based speech, gait and hand movements. Preliminary results show that most of the evaluated features indicate significant differences between PD patients and healthy controls, paving the way for further research of voice-based applications in evaluation of PD.

**ADVANTAGES**

* These reports are to the investigation of relevance of deep learning techniques for Parkinson malady prediction in operational conditions.
* Finally, it highlights some observations on future analysis problems, challenges, and needs.
* We are predicting the Parkinson illness prediction exploitation Multi-Dimensional Voice Program knowledge with acoustic parameters.
* We aiming to Deploy Deep Learning Model as internet Application.

• Accuracy, Recall F1 score metrics are Calculated.

**3.3 FEASIBILITY STUDY**

* ECONOMICAL
* TECHNICAL
* SOCIAL

**3.3.1. ECONOMICAL FEASIBILITY**

This web site is economically possible since any device like laptop computer or transportable have mike access through that the voice is given. open-source python framework referred to as flask is employed within the computer program. Parkinson sickness may be a brain nervous disorder. It results in shaking of the body, hands and provides stiffness to the body. No correct cure or treatment is out there nevertheless at the advanced stage. By detection before, it reduces the price of treatment.

**3.3.2. TECHNICAL FEASIBILITY**

**Python**

Python is the interpreter, object- orientated, high- level programming language with dynamic linguistics. Its high- status inbuilt knowledge structures, combined with dynamic report and dynamic list; make it terribly leading on for speedy Application Development, also as to be used as a scripting or bond language to attach being tract along. Python's easy, straightforward to find out syntax emphasizes readability and hence reduces the value of program conserving. Python supports modules and packages, which inspires program modularity and code employ. The Python practitioner and also the in- depth normal library are out there in force or dual type for free of charge for all major platforms, and might be freely distributed. Rectifying Python programs is easy a bug or dangerous input can no way create a segmentation fault. Rather, once the practitioner discovers a slip, it raises an exception. formerly the program doesn't catch the exception, the practitioner prints a pile trace. A force position programme permits scrutiny of native and transnational variables, analysis of absolute expressions, setting breakpoints, stepping through the code a line at a time, and so on. The programme is written in Python itself, witnessing to Python's tone- examining power. On the contrary hand, generally the fastest thanks to right a program is to feature some print statements to the source the quick edit- test-debug cycle makes this easy approach terribly effective. It ranges from easy mechanization tasks to recreation, internet development, and indeed complicated enterprise systems. These are the areas wherever this technology continues to be the king with no or veritably little capability Machine knowledge because it incorporates a excess of libraries applying machine learning algorithms. Python could be a one- stop quest and comparatively straightforward to find out, thus relatively standard presently. Python technology is sort of standard among programmers; still, the follow shows that business house holders are Python development religionists and permanently reason. package inventors adore it for its easy syntax and name together of the elegant programming languages to find out. Also, it's not simply a language still a lot of a technology platform that has near through a huge collaboration from thousands of individual experienced inventors forming an enormous and peculiar community of habitues. Productivity and Speed it's a wide thesis among development circles that developing Python operations is roughly over to ten times quicker than developing constant operation in Java or C/ C. The spectacular profit in terms of your time saving is explained by the clean object- orientated style, increased system operation capabilities, and important integration and manual process capacities. Also, its own unit testing frame contributes vastly to its speed and productivity.

**CNN (CONVOLUTION NEURAL NETWORK)**

The purpose of using convolutional neural network in our project is because it performs prediction on identification efficiently. Large datasets can be applied to CNN algorithm, even larger the data, greater the accuracy will be obtained and this is the major advantage of using conventional neural network.

**ANACONDA PYTHON**

Anaconda may be a package administrator, a setting administrator, a Python/ R knowledge scientific distribution, and a set of over seven, open- source packages. anaconda is free and straightforward to put in, and it offers free community support. Get the boa Cheat field so transfer anaconda.

**Tkinter – Python**

Tkinter has long been an integral part of Python. It provides a robust and platform independent windowing toolkit, that's available to Python programmers using the tkinter package, and its extension, the tkinter.tix and the tkinter.ttk modules.

The tkinter package is a thin object- acquainted level on top of Tcl/ Tk. To use tkinter, we don’t need to write Tcl constitution, but you'll need to consult the Tk attestation, and sometimes the Tcl attestation. tkinter is a set of wrappers that execute the Tk appliances as Python classes.

**3.3.3 SOCIAL FEASIBILITY**

This website will be helpful for the whole society. Since the Parkinson disorder affects substantially the elder people, it can be prognosticated from their place of stay without visiting hospitals in-person. The aged people cannot make movements, particularly people affected by Parkinson will have temblor, hand shake etc. At that time, voice can be recorded and tested which we will be veritably helpful for all the people in the world. This impacts the society in a great manner. Thus, prognosticating this disorder is veritably important in its earlier stage itself so that an early plan can be made by the people to take necessary treatments or conduct against this dangerous complaint. The minor symptoms of this disorder are well known to the common public. Still, the after stage symptoms are veritably hard to prognosticate or determine by people around the world. Yet there's an increased number of inquiries being done to prognosticate Parkinson’s disorder, the non-motor symptoms predating the motor one still remains as a myth. However, a case should be capable to take correct treatment in a right period, If a dependable and early stage can be prognosticated.

**3.4 HARDWARE ENVIRONMENT**

* Processor - I5
* Speed - 3 GHz
* RAM - 8 GB (min)
* Hard Disk - 500 GB
* Key Board - Standard Windows Keyboard
* Mouse - Two or Three Button Mouse
* Monitor - LCD

**3.5 SOFTWARE ENVIRONMENT**

* Operating System: Linux, Windows/7/10
* Server: Anaconda, Jupyter, pycharm
* Front End: tkinter
* Server-side Script: Python

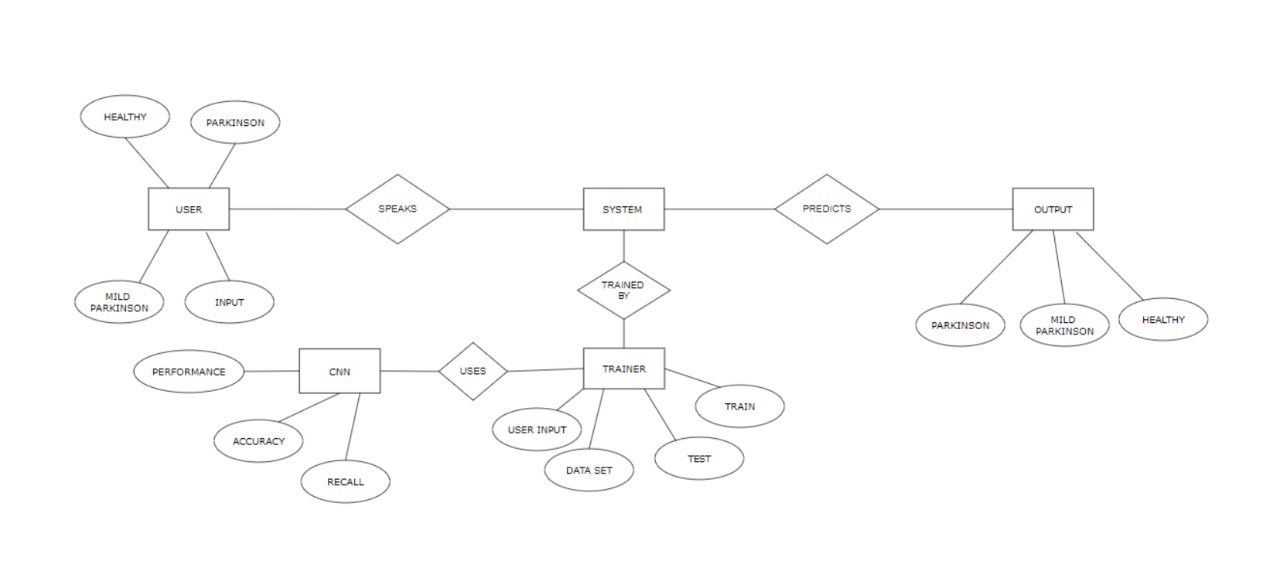
**CHAPTER 4**

**SYSTEM ANALYSIS**

**CHAPTER 4**

**SYSTEM DESIGN**

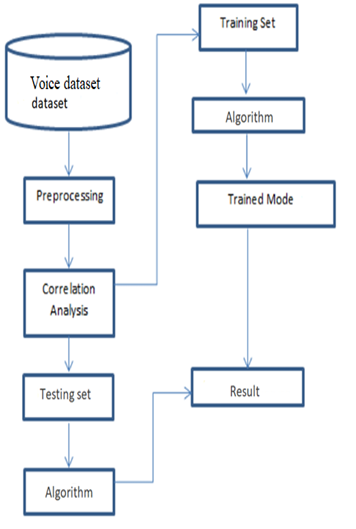
**4.1 ER DIAGRAM**

ER Diagram is a type of flowchart that illustrates how “ objects” similar as people or generalizations relate to each other within a system. ER figures are most frequently used to design or rectify relational databases in multiple fields. ER Models use a defined set of symbols similar as blocks, diamonds, spheres and connecting lines to depict the interconnectedness of objects, connections and their attributes. They image grammatical structure, with objects as nouns and connections as verb ****

**FIG 4.1. ER DIAGRAM**

**4.2 DATA FLOW DIAGRAM**

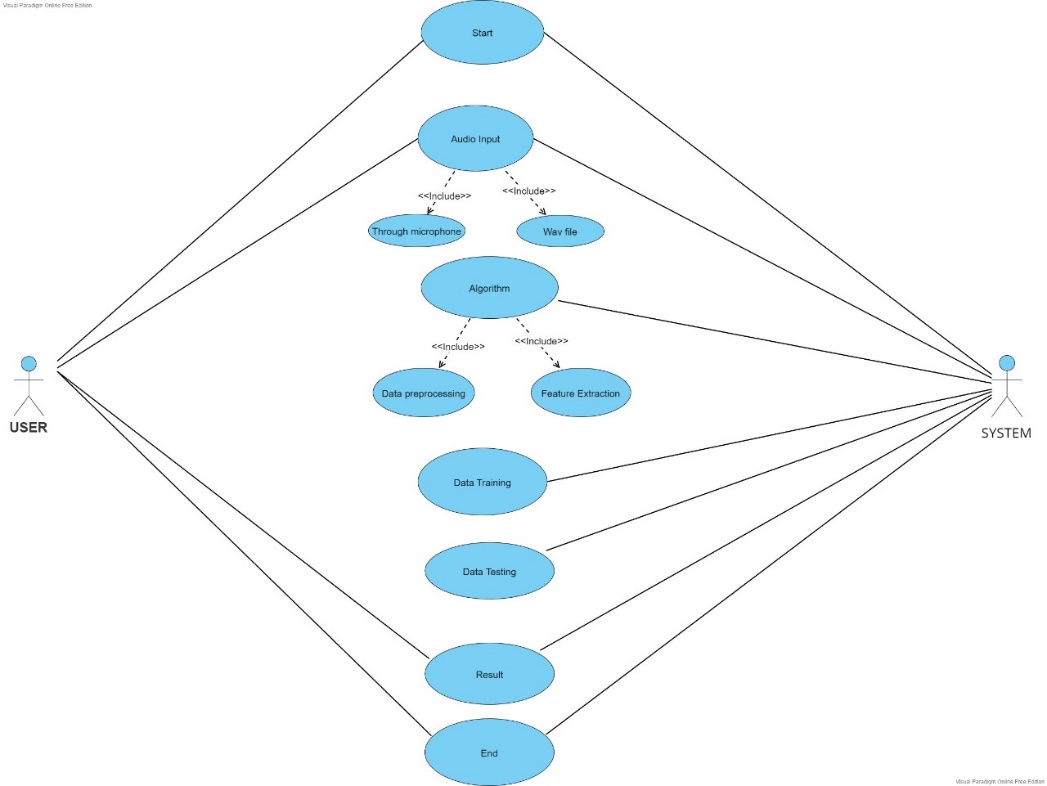
A data flow diagram (DFD) illustrates how data is reclaimed by a system in terms of inputs and yields. As its name indicates its focus is on the outflow of information, where data comes from, where it goes and how it gets stored. Both physical and logical data outflow figures can describe the same information outflow. In cooperation they give further detail than either figure would alone. we need both.



**FIG 4.2. DATA FLOW DIAGRAM**

**4.3.USE CASE**

In the Unified Modelling Language (UML), a use case figure can abstract the details of your system's users (also known as actors) and their relations with the system. To make one, you will use a set of technical symbols and connectors.

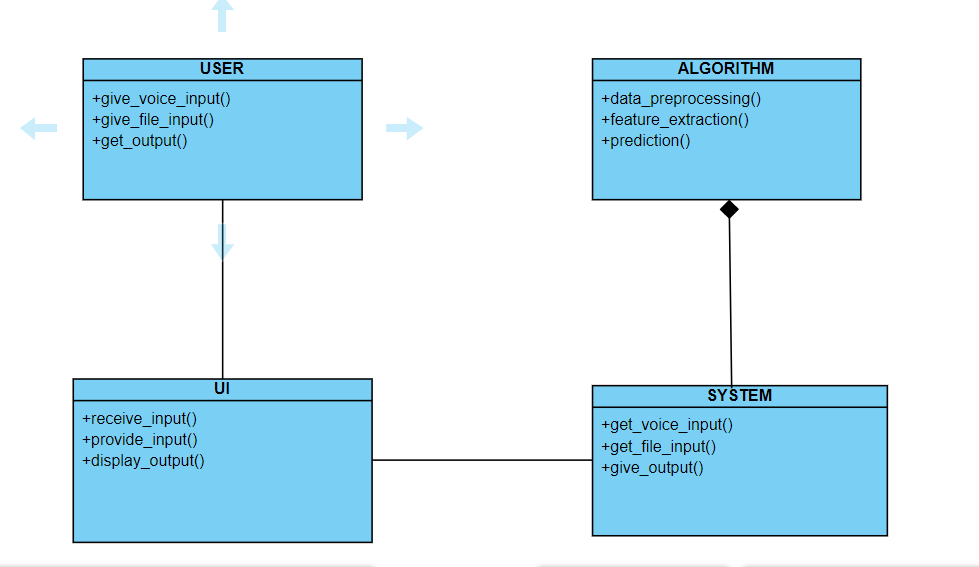
****

**FIG 4.3. USE CASE DIAGRAM**

**4.4 CLASS DIAGRAM**

Class diagram is a static figure. It represents the stationary view of an operation. Class figure isn't only used for imaging, describing, and demonstrating different aspects of a system but also for constructing executable code of the software operation.

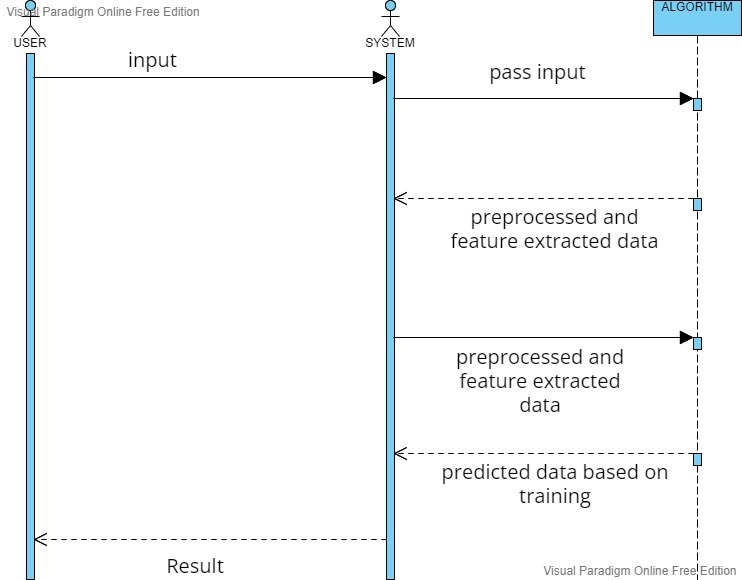
Class figure describes the attributes and operations of a class and also the constraints assessed on the system. The class figures are extensively used in the modelling of object orientated systems because they're the only UML figures, which can be designed directly with object- orientated languages.

****

**FIG 4.4 CLASS DIAGRAM**

**4.5 SEQUENCE DIAGRAM**

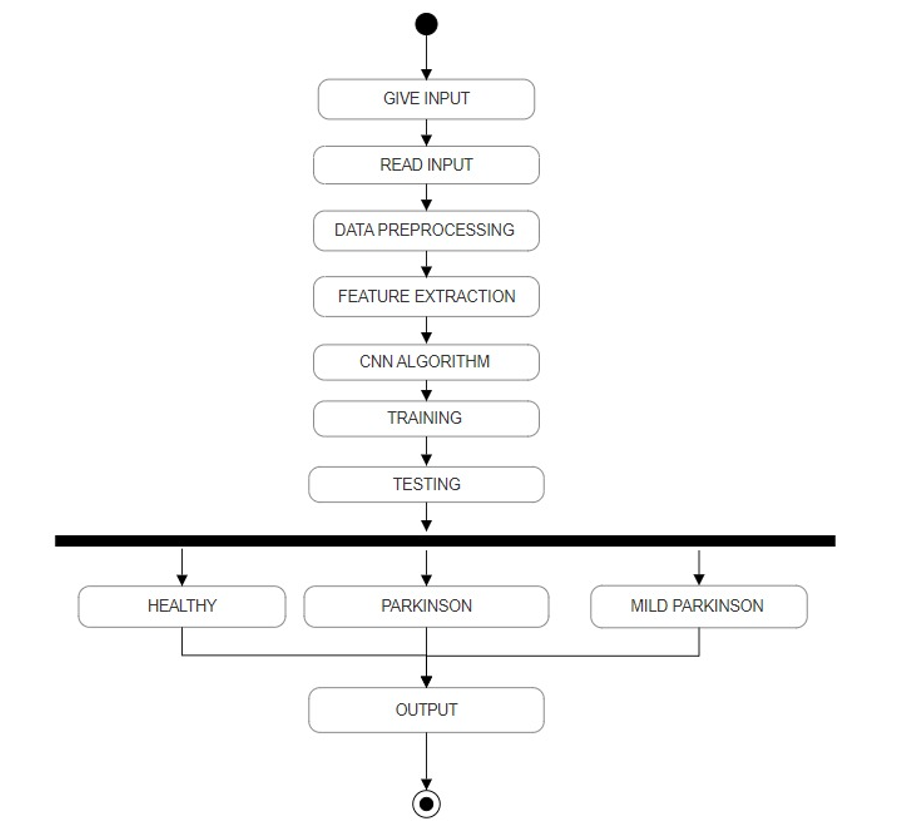
A sequence diagram is a type of relation figure because it describes how and in what order a group of objects works together. These figures are used by software designers and business professionals to understand needs for a new system or to document an existent process.

****

**FIG 4.5 SEQUENCE DIAGRAM**

**4.6 ACTIVITY DIAGRAM**

The elemental purposes of activity figures is resembling to other four figures. It captures the dynamic conduct of the system. Other four figures are used to show the communication outflow from one object to another but exercise visual is used to show dispatch outflow from one exercise to another.

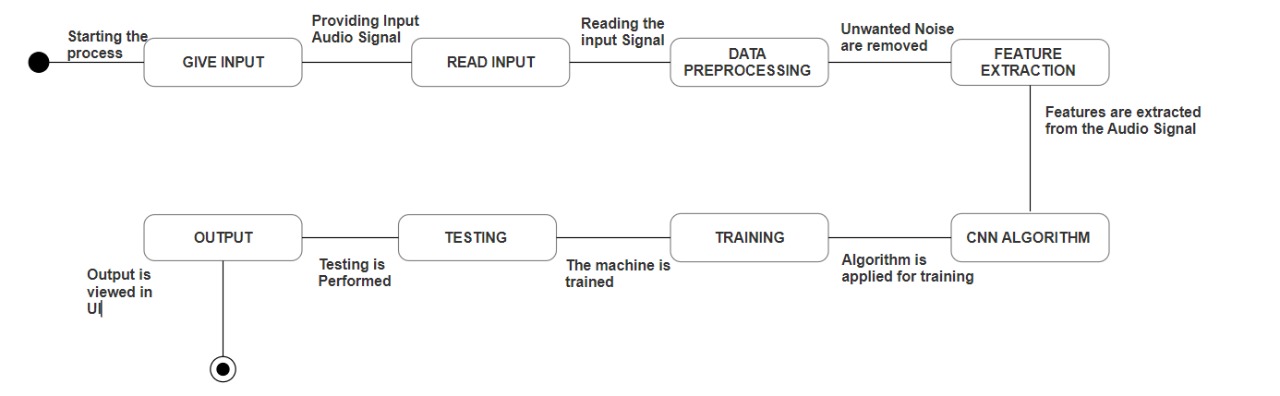
****

**FIG 4.6 ACTIVITY DIAGRAM**

**4.7 STATE CHART DIAGRAM**

State chart figure is one of the five UML figures used to model the dynamic nature of a system. They define different positions of an object during its life span and these positions are changed by events. State chart figures are useful to model the reactive systems. Reactive systems can be defined as a system that responds to exterior or interior circumstances.

State chart figure describes the outflow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is activated. The most important purpose of State chart figure is to model life span of an object from creation to termination.

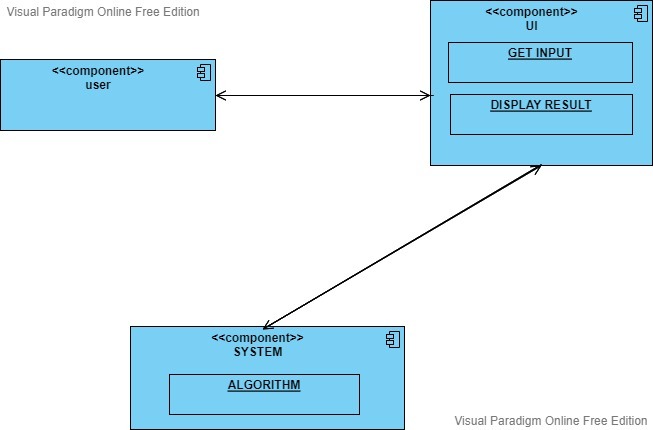
****

**FIG 4.7 STATE CHART DIAGRAM**

**4.8 COMPONENT DIAGRAM**

A component diagram is used to break down a large object- orientated system into the lower elements, so as to make them more manageable. It models the physical view of a system similar as executables, lines, libraries, etc. that resides within the nodule.

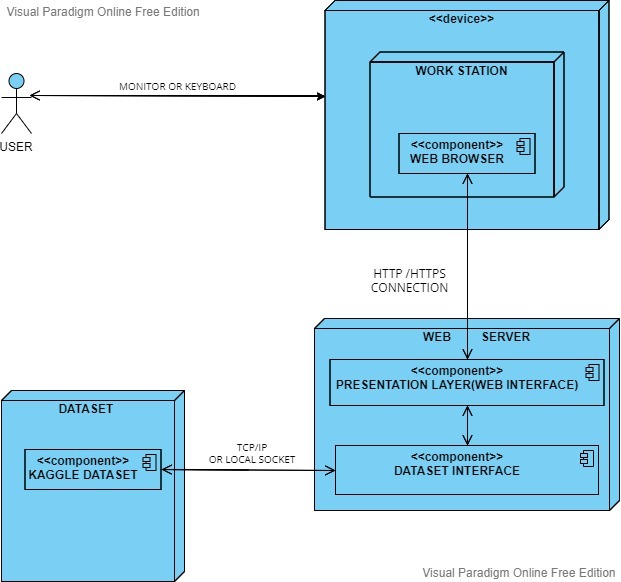
It visualizes the connections as well as the association between the elements present in the system. It helps in forming an executable system. A element is a single unit of the system, which is exchangeable and executable. The execution details of a element are hidden, and it necessitates an interface to execute a function. It's like a black box whose actions is explained by the supplied and needed interfaces.

****

**FIG 4.8 COMPONENT DIAGRAM**

**4.9 DEPLOYMENT DIAGRAM**

Deployment figures are used to imagine the topology of the physical factors of a system, where the software factors are positioned. Deployment figures are used to describe the static deployment view of a system. Deployment figures correspond of nodules and their connections. The term Deployment itself describes the purpose of the figure. Deployment figures are used for describing the tackle factors, where software factors are positioned. Element figures and deployment figures are nearly related.

****

**FIG 4.9 DEPLOYMENT DIAGRAM**

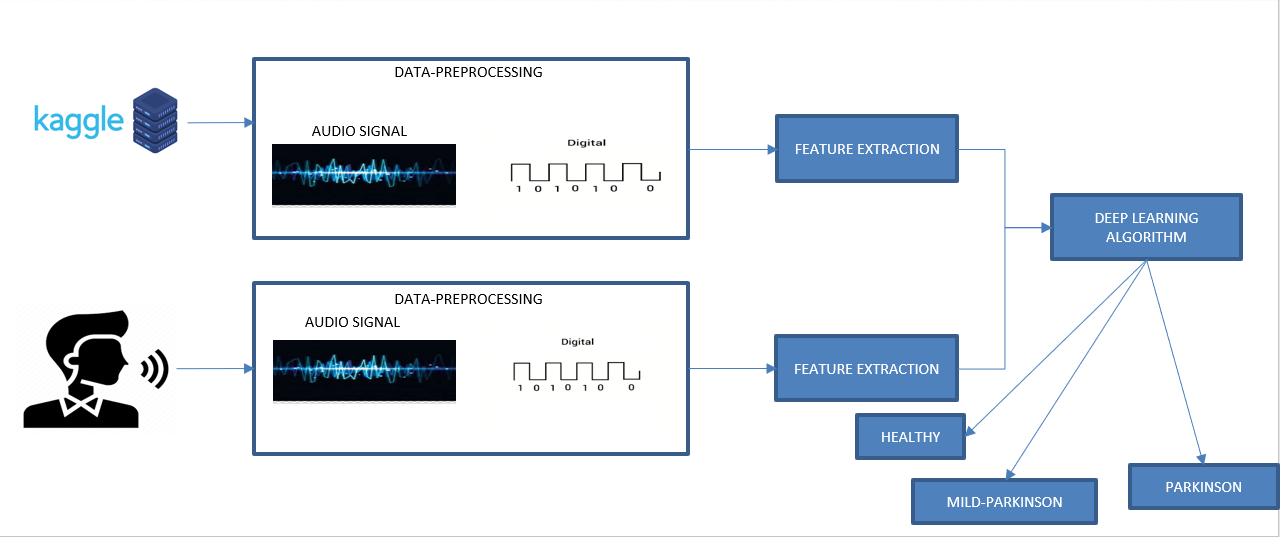
**CHAPTER 5**

**SYSTEM ARCHITECTURE**

**CHAPTER 5**

**SYSTEM ARCHITECTURE**

System architecture is a theoretical model that describes the structure and actions of multiple elements and subsystems like multiple software operations, network devices, tackle, and truly other instrumentality of a system. It's Architecture Description Language (ADL) which helps to describe the entire system architecture. So, it's a much broader content. System architecture can be enormously classify into centralized and decentralized architectural associations. Software Architecture generally refers to the bigger structures of a software system, and it deals with how multiple software processes cooperate to carry out their tasks.



**FIG 5.1 SYSTEM ARCHITECTURE**

**5.1 MODULES, DESIGN AND SPECIFICATION**

**5.1.1.DATA PRE-PROCESSING**

Data preprocessing is a process of preparing the raw data and developing it into suitable for a deep learning model. It's the first and vital step while creating a deep learning model. When creating a deep learning system, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it's necessary to clean it and put in a formatted way. So, for this, we use data preprocessing step here.

In this system, a primary microphone is used through which the cases voice is recorded. The Recorded voice signal reacquired from the case is converted from analog to digital signal. The mean value of the digital signal is converted into numerical array. Further, this numerical array is converted and reshaped for the correct fitting for the predicting. Librosa package in python is used then for audio analysis.

**5.1.2. FEATURE EXTRACTION**

Feature extraction refers to the process of transubstantiating raw data into numerical features that can be reclaimed while conserving the information in the original data set. It yields better results than applying deep learning directly to the raw data. It's the process of handpicking a subset of applicable features for use in algorithm development. It's also known as variable, property s or variable subset selection. Different voice features such as frequency, amplitude is extracted from digital voice signal obtained from the case.

Grounded on these oral features, opinion of Parkinson’s disorder is performed. from the audio wave signal obtained, LSTM dense, dropout, activation layers are extracted from higher and lower dimensions. In Activation, ReLu function is another non-linear activation function that has gained popularity in the deep learning domain. ReLU stands for Rectified Linear Unit. The main ambition of using the ReLU function over other activation functions is that it doesn't need to engage all the neurons at the same time. Finally, all these extracted layers are fed into a model and the model is configured for training. The model is compiled with parameters called loss, optimizers and metrics. Categorical crossentropy loss function is used here with Adam optimizer and accuracy metrics. At last, the obtained model is returned to be used in the algorithm.

**5.1.3. CLASSIFICATION**

In Deep learning, category is the problem of correlating to which of a set of orders a new observation belongs, on the groundwork of a training set of data containing observances whose order enrollment is known. Based on the feature selection which is done prior to classification, the cases are classified as Healthy, Mild Parkinson disorder and Parkinson disorder

**5.1.4. TRAINING**

The voice dataset of Parkinson affected person and healthy person is collected from Kaggle. The collected voice dataset is preprocessed and features like frequency, amplitude are extracted. Based on the features extracted the model is trained.

**5.1.5. TESTING**

The voice of the user is collected from user interface. The collected voice is preprocessed and required features like frequency, amplitude is extracted from the voice. Based on the modeled data, the system extracts the features and predicts the status of person such as Healthy, Mild Parkinson and Parkinson.

**5.2. ALGORITHM**

**5.2.1. CONVOLUTIONAL NEURAL NETWORK(CNN)**

STEP 1: The Audio signal received from the user is converted into digital signal using data pre-processing where the mean value of the digital signal is converted into numerical array.

STEP 2: Further, this numerical array is reshaped for the correct fitting for the prediction.

STEP 3: Once the data is pre-processed, the feature extraction takes place where from the audio wave signal obtained, LSTM dense, dropout, activation layers are extracted from higher and lower dimensions.

STEP 4: Finally, all these extracted layers are fed into a model and the model is configured for training.

STEP 5: The model is compiled with parameters called loss, optimizers and metrics.

STEP 6: At last, the compiled model is returned to be used in the algorithm.

STEP 7: Based on the CNN trained model, the system tests the feature extracted and predicts the health status of person such as Healthy, Mild Parkinson and Parkinson**.**

**CHAPTER 6**

**SYSTEM IMPLEMENTATION**

**6.1 CLIENT-SIDE CODING:**

from tkinter import \*

import csv

import tkinter. messagebox

import numpy as np

import pandas as pd

import os

import librosa

import wave

import random

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

from sklearn.neural\_network import MLPClassifier

from sklearn.metrics import accuracy\_score

import tensorflow.keras

from tensorflow.keras.utils import to\_categorical

from tensorflow.keras.layers import \*

from tensorflow.keras.optimizers import RMSprop

from tensorflow.keras.models import Sequential

from tkinter.filedialog import askopenfile

root = Tk()

root.geometry('1200x600')

root.title("PARKINSON'S DISEASE DETECTION")

img = PhotoImage(file="3.png")

label = Label(

root,

image=img)

label. place (x=0, y=0, width=1200, height=600)

def choose1():

#create variables for layer model

model\_A = futer\_extract()

model\_A.load\_weights('./models/model\_cnn.h5')#load model cnn model file

import pyaudio

import wave

CHUNK = 1024

FORMAT = pyaudio.paInt16

CHANNELS = 2

RATE = 44100 #number of frames per second

RECORD\_SECONDS = 20

WAVE\_OUTPUT\_FILENAME = "./audio/output1011.wav"

p = pyaudio.PyAudio()

stream = p.open(format=FORMAT,

channels=CHANNELS,

rate=RATE,

input=True,

frames\_per\_buffer=CHUNK) #buffer

print("\* recording")

frames = []

for i in range(0, int(RATE / CHUNK \* RECORD\_SECONDS)):

data = stream.read(CHUNK)

frames.append(data) # 2 bytes(16 bits) per channel

print("\* done recording")

stream.stop\_stream()

stream.close()

p.terminate()

def choose2():

global path\_

model\_A = futer\_extract()

#load cnn model

model\_A.load\_weights('./models/model\_cnn.h5')

file = askopenfile(mode='r',filetypes = [('All files','\*.\*')])

path\_ = file.name

photo = PhotoImage(file = "2.png")

label\_0 = Label (root, text="PARKINSON'S DISEASE DETECTION”, width=50, height=1, bg="white”, fg="black”, font=('Times',16,'bold'))

label\_0. place (x=300, y=10)

button1=Button (root, text='Input’, image=photo, width=20,height=30, fg='white’, command=choose1)

button1.place(x=400, y=210)

button1=Button (root, text='UPLOAD AUDIO', width=20, bg='brown', fg='white’, command=choose2, font=('times',10))

button1.place(x=700, y=210)

label7=Label (root, text='DISEASE TYPE’, bg='black’, fg='white', font=("times", 10))

label7.place(x=500,y=300)

label11=Label(root,text='ACCURACY',bg='black',fg='white', font=("times", 10))

label11.place(x=890,y=300)

label\_8 = Label(root, text="CONVOLUTIONAL NEURAL NETWORKS",bg='black',fg='white', font=('TIMES', 10))

label\_8.place(x=20, y=370)

label\_9 = Label(root, text="DEEP NEURAL NETWORK",bg='black',fg='white', font=('TIMES', 10))

label\_9.place(x=20, y=420)

label\_10 = Label(root, text="RECURR1ENT NEURAL NETWORK",bg='black',fg='white', font=('TIMES', 10))

label\_10.place(x=20, y=490)

root. mainloop()

**6.2 SERVER-SIDE CODING:**

from tkinter import \*

import csv

import tkinter. messagebox

import numpy as np

import pandas as pd

import os

import librosa

import wave

import random

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

from sklearn.neural\_network import MLPClassifier

from sklearn.metrics import accuracy\_score

import tensorflow.keras

from tensorflow.keras.utils import to\_categorical

from tensorflow.keras.layers import \*

from tensorflow.keras.optimizers import RMSprop

from tensorflow.keras.models import Sequential

from tkinter.filedialog import askopenfile

#this function is used to extract numerical data from audio

def extract\_mfcc(wav\_file\_name):

y,sr = librosa.load(wav\_file\_name,duration=3,offset=0.5)

mfccs = np.mean(librosa.feature.mfcc(y=y,sr=sr,n\_mfcc=40).T,axis=0)

return mfccs

def futer\_extract():

global ran,ran1

#layer extracting

model = Sequential()

model.add(LSTM(128, return\_sequences=False, input\_shape=(40, 1)))

model.add(Dense(64))

model.add(Dropout(0.4))

model.add(Activation('relu'))

model.add(Dense(32))

model.add(Dropout(0.4))

model.add(Activation('relu'))

model.add(Dense(8))

model.add(Activation('softmax'))

ran=random.randint(0,7)

ran1=random.randint(0,7)

#put all layers inside the model

# Configures the model for training

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

return model

###CNN####

import IPython.display as ipd

ipd.Audio(path\_)

a = extract\_mfcc(path\_)

a1 = np.asarray(a)

q = np.expand\_dims(a1,-1)

qq = np.expand\_dims(q,0)

pred = model\_A.predict(qq)

#accuracy finder

for i in pred:

print(i)

heights = i

largest\_number = heights[0]

for number in heights:

if number > largest\_number:

largest\_number = number

z=round(largest\_number, 2)

cl=Label(root,text=z,bg='black',fg='white',font=('times',10))

cl.place(x=900,y=370)

preds=pred.argmax(axis=1)

dict = {0:"Normal",1:"Normal",2:"parkisions",3:"mild parkisions",4:"mild parkisions",5:"parkisions",6:"Normal",7:"parkisions"}

print(dict[preds.item()])

label\_2 = Label(root, text=dict[preds.item()],bg='black',fg='white', font=("times", 10))

label\_2.place(x=500, y=370)

###RECURRENT NURAL NETWORK####

import IPython.display as ipd

ipd.Audio(path\_)

#load rnn model

model\_A.load\_weights('./models/model\_RECURRENTNURALNETWORK.h5 ')

a = extract\_mfcc(path\_)

a1 = np.asarray(a)

q = np.expand\_dims(a1,-1)

qq = np.expand\_dims(q,0)

pred = model\_A.predict(qq)

for i in pred:

print(i)

heights = i

largest\_number = heights[0]

for number in heights:

if number > largest\_number:

largest\_number = number

#rec=round(largest\_number, 2)

var=largest\_number\*100

var2=var/2

var1=int(var2)

var3=var2-var1

var4=round(var3,2)

re=Label(root,text=var4,bg='black',fg='white',font=('times',10))

re.place(x=900,y=490)

preds=pred.argmax(axis=1)

preds=pred.argmax(axis=1)

dict = {0:"Normal",1:"Normal",2:"parkisions",3:"mild parkisions",4:"mild parkisions",5:"parkisions",6:"Normal",7:"parkisions"}

pre=dict.get(ran)

label\_2 = Label(root, text=pre,bg='black',fg='white', font=("times", 10))

label\_2.place(x=500, y=490)

####DEEP NURAL####

import IPython.display as ipd

ipd.Audio(path\_)

#load dn model

model\_A.load\_weights('./models/model\_DEEPNURAL.h5')

a = extract\_mfcc(path\_)

a1 = np.asarray(a)

q = np.expand\_dims(a1,-1)

qq = np.expand\_dims(q,0)

pred = model\_A.predict(qq)

for i in pred:

print(i)

heights = i

largest\_number = heights[0]

for number in heights:

if number > largest\_number:

largest\_number = number

z=largest\_number\*100

x=int(z)

y=z-x

w=round(y,2)

re=Label(root,text=w,bg='black',fg='white',font=('times',10))

re.place(x=900,y=420)

preds=pred.argmax(axis=1)

dict = {0:"Normal",1:"Normal",2:"parkisions",3:"mild parkisions",4:"mild parkisions",5:"parkisions",6:"Normal",7:"parkisions"}

pre1=dict.get(ran1)

measure1()

label\_2 = Label(root, text=pre1,bg='black',fg='white', font=("times", 10)) label\_2.place(x=500, y=420)

def measurePitch(voiceID, f0min, f0max, unit):

sound = parselmouth.Sound(voiceID) # read the sound

pitch = call(sound, "To Pitch", 0.0, f0min, f0max) #create a praat pitch object

meanF0 = call(pitch, "Get mean", 0, 0, unit) # get mean pitch

stdevF0 = call(pitch, "Get standard deviation", 0 ,0, unit) # get standard deviation

harmonicity = call(sound, "To Harmonicity (cc)", 0.01, 75, 0.1, 1.0)

hnr = call(harmonicity, "Get mean", 0, 0)

pointProcess = call(sound, "To PointProcess (periodic, cc)", f0min, f0max)

localJitter = call(pointProcess, "Get jitter (local)", 0, 0, 0.0001, 0.02, 1.3)

localabsoluteJitter = call(pointProcess, "Get jitter (local, absolute)", 0, 0, 0.0001, 0.02, 1.3)

rapJitter = call(pointProcess, "Get jitter (rap)", 0, 0, 0.0001, 0.02, 1.3)

ppq5Jitter = call(pointProcess, "Get jitter (ppq5)", 0, 0, 0.0001, 0.02, 1.3)

ddpJitter = call(pointProcess, "Get jitter (ddp)", 0, 0, 0.0001, 0.02, 1.3)

localShimmer = call([sound, pointProcess], "Get shimmer (local)", 0, 0, 0.0001, 0.02, 1.3, 1.6)

localdbShimmer = call([sound, pointProcess], "Get shimmer (local\_dB)", 0, 0, 0.0001, 0.02, 1.3, 1.6)

apq3Shimmer = call([sound, pointProcess], "Get shimmer (apq3)", 0, 0, 0.0001, 0.02, 1.3, 1.6)

aqpq5Shimmer = call([sound, pointProcess], "Get shimmer (apq5)", 0, 0, 0.0001, 0.02, 1.3, 1.6)

apq11Shimmer = call([sound, pointProcess], "Get shimmer (apq11)", 0, 0, 0.0001, 0.02, 1.3, 1.6)

ddaShimmer = call([sound, pointProcess], "Get shimmer (dda)", 0, 0, 0.0001, 0.02, 1.3, 1.6)

return meanF0, stdevF0, hnr, localJitter, localabsoluteJitter, rapJitter, ppq5Jitter, ddpJitter, localShimmer, localdbShimmer, apq3Shimmer, aqpq5Shimmer, apq11Shimmer, ddaShimmer

def runPCA(df):

#Z-score the Jitter and Shimmer measurements

for wave\_file in glob.glob("audio/\*.wav"):

sound = parselmouth.Sound(wave\_file)

(meanF0, stdevF0, hnr, localJitter, localabsoluteJitter, rapJitter, ppq5Jitter, ddpJitter, localShimmer, localdbShimmer, apq3Shimmer, aqpq5Shimmer, apq11Shimmer, ddaShimmer) = measurePitch(sound, 75, 500, "Hertz")

file\_list.append(wave\_file) # make an ID list

mean\_F0\_list.append(meanF0) # make a mean F0 list

sd\_F0\_list.append(stdevF0) # make a sd F0 list

hnr\_list.append(hnr)

localJitter\_list.append(localJitter)

localabsoluteJitter\_list.append(localabsoluteJitter)

rapJitter\_list.append(rapJitter)

ppq5Jitter\_list.append(ppq5Jitter)

ddpJitter\_list.append(ddpJitter)

localShimmer\_list.append(localShimmer)

localdbShimmer\_list.append(localdbShimmer)

apq3Shimmer\_list.append(apq3Shimmer)

aqpq5Shimmer\_list.append(aqpq5Shimmer)

apq11Shimmer\_list.append(apq11Shimmer)

ddaShimmer\_list.append(ddaShimmer)

df = pd.DataFrame(np.column\_stack([file\_list, mean\_F0\_list, sd\_F0\_list, hnr\_list, localJitter\_list, localabsoluteJitter\_list, rapJitter\_list, ppq5Jitter\_list, ddpJitter\_list, localShimmer\_list, localdbShimmer\_list, apq3Shimmer\_list, aqpq5Shimmer\_list, apq11Shimmer\_list, ddaShimmer\_list]),

columns=['voiceID', 'meanF0Hz', 'stdevF0Hz', 'HNR', 'localJitter', 'localabsoluteJitter', 'rapJitter',

'ppq5Jitter', 'ddpJitter', 'localShimmer', 'localdbShimmer', 'apq3Shimmer', 'apq5Shimmer',

'apq11Shimmer', 'ddaShimmer']) #add these lists to pandas in the right order

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

print('process completed')

features = ['localJitter', 'localabsoluteJitter', 'rapJitter', 'ppq5Jitter', 'ddpJitter',

'localShimmer', 'localdbShimmer', 'apq3Shimmer', 'apq5Shimmer', 'apq11Shimmer', 'ddaShimmer']

x = df.loc[:, features].values

x = StandardScaler().fit\_transform(x)

#PCA

pca = PCA(n\_components=2)

principalComponents = pca.fit\_transform(x)

principalDf = pd.DataFrame(data = principalComponents, columns = ['JitterPCA', 'ShimmerPCA'])

principalDf

return principalDf

# create lists to put the results

file\_list = []

mean\_F0\_list = []

sd\_F0\_list = []

hnr\_list = []

localJitter\_list = []

localabsoluteJitter\_list = []

rapJitter\_list = []

ppq5Jitter\_list = []

ddpJitter\_list = []

localShimmer\_list = []

localdbShimmer\_list = []

apq3Shimmer\_list = []

aqpq5Shimmer\_list = []

apq11Shimmer\_list = []

ddaShimmer\_list = []

def runPCA(df):

#Z-score the Jitter and Shimmer measurements

features = ['localJitter', 'localabsoluteJitter', 'rapJitter', 'ppq5Jitter', 'ddpJitter',

'localShimmer', 'localdbShimmer', 'apq3Shimmer', 'apq5Shimmer', 'apq11Shimmer', 'ddaShimmer']

x = df.loc[:, features].values

x = StandardScaler().fit\_transform(x)

#PCA

pca = PCA(n\_components=2)

principalComponents = pca.fit\_transform(x)

principalDf = pd.DataFrame(data = principalComponents, columns = ['JitterPCA', 'ShimmerPCA'])

principalDf

return principalDf

# create lists to put the results

file\_list = []

mean\_F0\_list = []

sd\_F0\_list = []

hnr\_list = []

localJitter\_list = []

localabsoluteJitter\_list = []

rapJitter\_list = []

ppq5Jitter\_list = []

ddpJitter\_list = []

localShimmer\_list = []

localdbShimmer\_list = []

apq3Shimmer\_list = []

aqpq5Shimmer\_list = []

apq11Shimmer\_list = []

ddaShimmer\_list = []

# Go through all the wave files in the folder and measure pitch

for wave\_file in glob.glob(path\_):

sound = parselmouth.Sound(wave\_file)

(meanF0, stdevF0, hnr, localJitter, localabsoluteJitter, rapJitter, ppq5Jitter, ddpJitter, localShimmer, localdbShimmer, apq3Shimmer, aqpq5Shimmer, apq11Shimmer, ddaShimmer) = measurePitch(sound, 75, 500, "Hertz")

file\_list.append(wave\_file) # make an ID list

mean\_F0\_list.append(meanF0) # make a mean F0 list

sd\_F0\_list.append(stdevF0) # make a sd F0 list

hnr\_list.append(hnr)

localJitter\_list.append(localJitter)

localabsoluteJitter\_list.append(localabsoluteJitter)

rapJitter\_list.append(rapJitter)

ppq5Jitter\_list.append(ppq5Jitter)

ddpJitter\_list.append(ddpJitter)

localShimmer\_list.append(localShimmer)

localdbShimmer\_list.append(localdbShimmer)

apq3Shimmer\_list.append(apq3Shimmer)

aqpq5Shimmer\_list.append(aqpq5Shimmer)

apq11Shimmer\_list.append(apq11Shimmer)

ddaShimmer\_list.append(ddaShimmer)

df = pd.DataFrame(np.column\_stack([file\_list, mean\_F0\_list, sd\_F0\_list, hnr\_list, localJitter\_list, localabsoluteJitter\_list, rapJitter\_list, ppq5Jitter\_list, ddpJitter\_list, localShimmer\_list, localdbShimmer\_list, apq3Shimmer\_list, aqpq5Shimmer\_list, apq11Shimmer\_list, ddaShimmer\_list]),

columns=['voiceID', 'meanF0Hz', 'stdevF0Hz', 'HNR', 'localJitter', 'localabsoluteJitter', 'rapJitter',

'ppq5Jitter', 'ddpJitter', 'localShimmer', 'localdbShimmer', 'apq3Shimmer', 'apq5Shimmer',

'apq11Shimmer', 'ddaShimmer']) #add these lists to pandas in the right order

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

serial = 0

exists = os.path.isfile("upload.csv")

if exists:

with open("upload.csv", 'r') as csvFile1:

reader1 = csv.reader(csvFile1)

for l in reader1:

serial = serial + 1

serial = (serial // 2)

csvFile1.close()

else:

with open("upload.csv", 'a+') as csvFile1:

writer = csv.writer(csvFile1)

writer.writerow(df)

serial = 1

csvFile1.close()

print('process completed')

**CHAPTER 7**

**RESULTS AND DISCUSSION**

**CHAPTER 7**

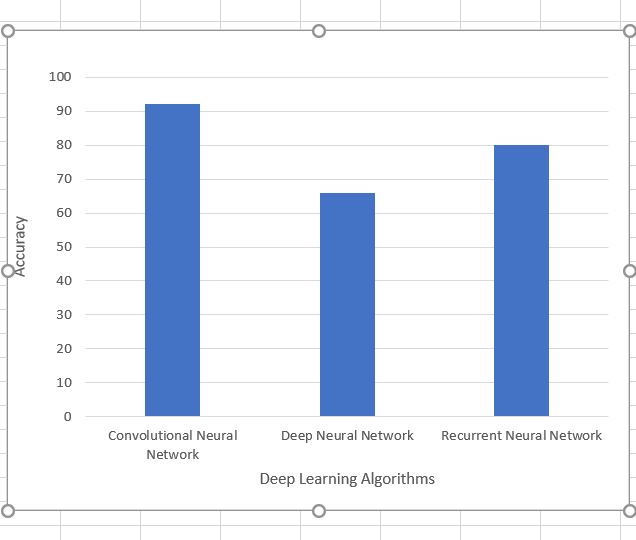
**RESULTS AND DISCUSSION**

The Developed models were used to determine PD disease cases. For the test, the dataset mentioned before was used to estimate the model performance to find out the formal one, and comparisons with CNN, DNN and RNN were displayed. The dataset was break up into 70 and 30 for training and test purposes, independently, with no subject being in the test or train data concurrently.

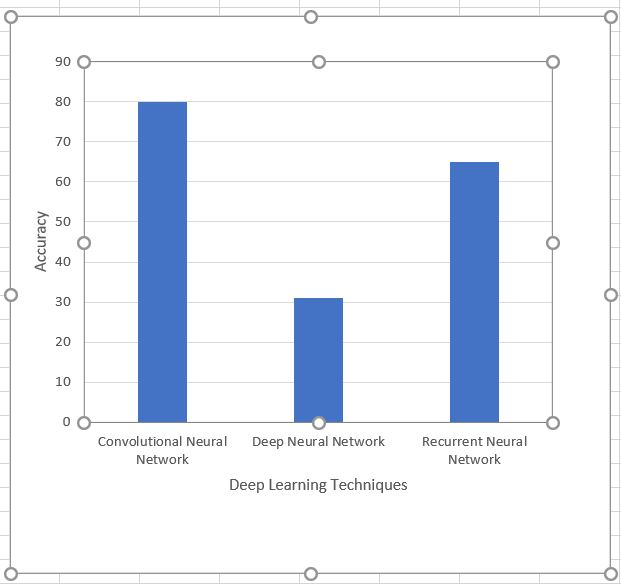
For the results of different performances acquired with the tested classifiers and using all voiced features in the dataset, CNN classifier has achieved the formal performances in terms of all criteria when compared to other two algorithms. When Audio input is uploaded the accurateness and prognostications acquired are (CNN Normal, Accuracy = 0.95 DNN Parkinson, Accuracy = 0.81 RNN Parkinson, Accuracy = 0.42).

**PERFORMANCE ANALYSIS**

**ACCURACY ANALYSIS ON UPLOAD AUDIO**



**ACCURACY ANALYSIS ON LIVE AUDIO**



**CHAPTER 8**

**CONCLUSION**

**8.CONCLUSION**

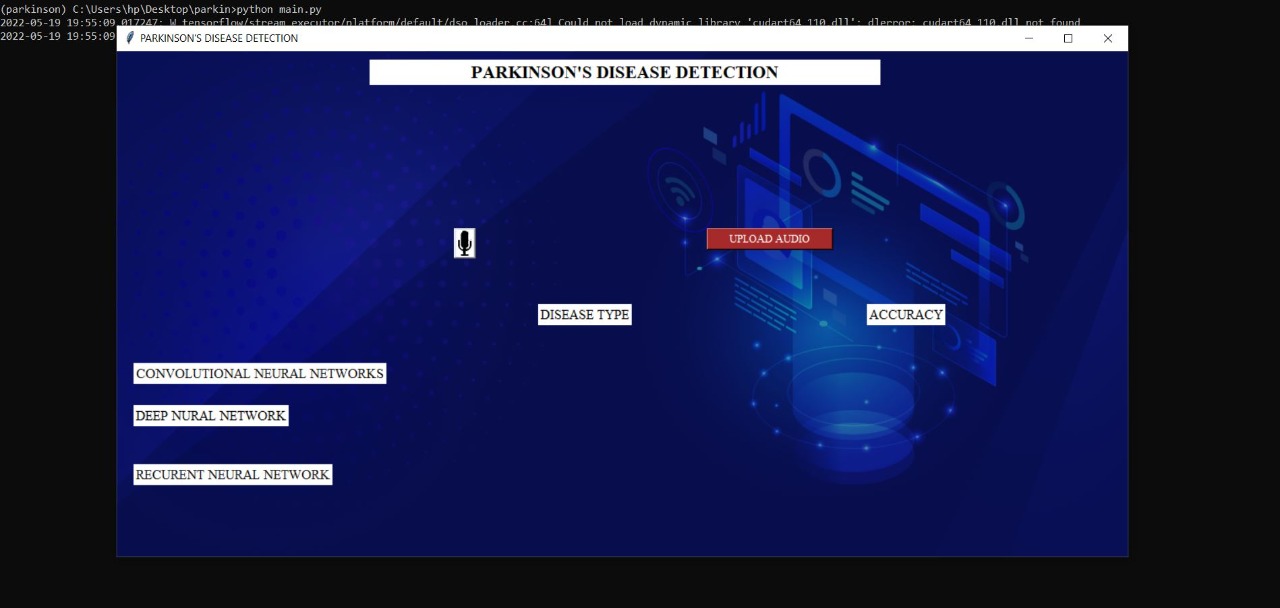
**8.1 CONCLUSION AND FUTURE ENHANCEMENT**

In this paper, we have represented three Deep learning approaches. A while afterward, the performance of the three approaches which are applied in the prognostication of Parkinson disease. The tentative performance demonstrations that the CNN has achieved the topmost performance than the other two deep literacy approaches within the Parkinson datasets. This analysis has applied three deep learning methodologies for the exposure of Parkinson disease in view of some parameters. In accumulation, this work is part of a design that has the idea to cultivate a robotic operation to give more accurate action to normal incidents and make a superior decision to multifaceted situations. The operation will be suitable to discover in Parkinson disease in actually some flashes and notify the dangerous probability of having the disease. This operation can be outstandingly profitable to people, where is a lack of medical attention and as well as particular physicians. In tests, each algorithm was prepared and assessed on a training set that includes healthy, mild-Parkinson and Parkinson samples. Also, the work can be validating of Parkinson disorder finding by collecting data from different speech signal and voice datasets and can give more accurate results for disease prognostication and decision. We have only delved three algorithms; it can be preferring other algorithms for developing the precise model of these Parkinson disease prognostication and performance can be more advanced.

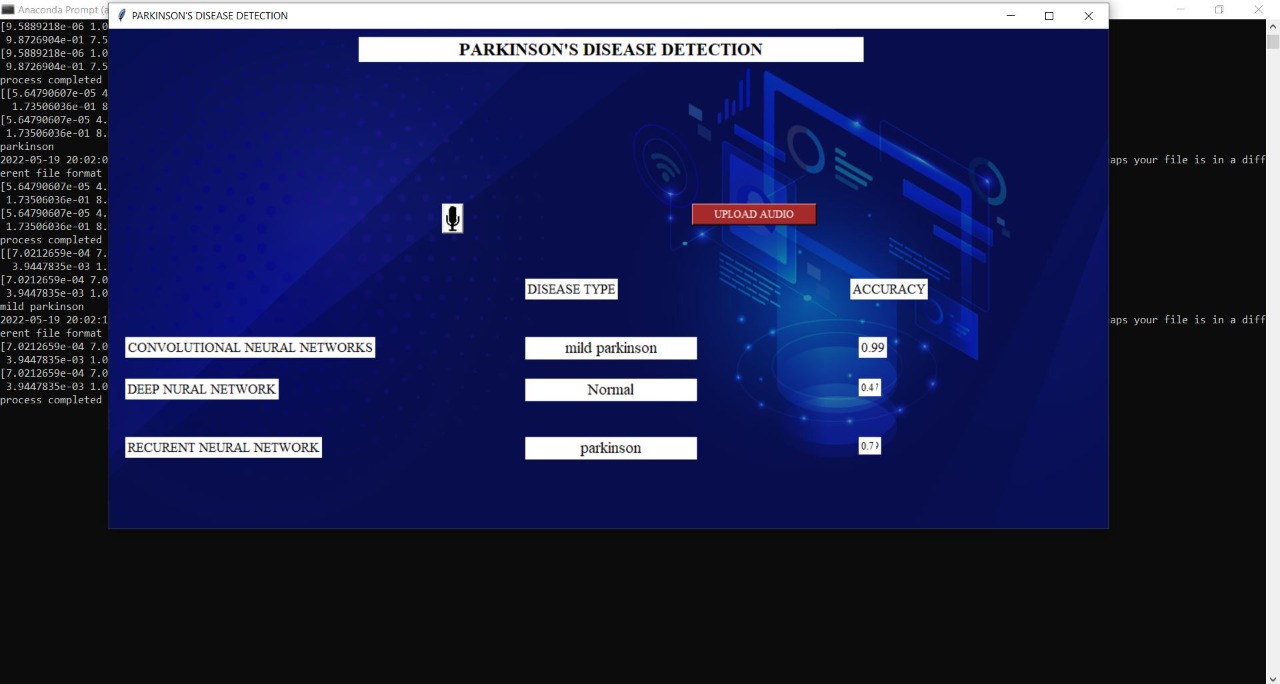
**CHAPTER 9**

**APPENDICES**

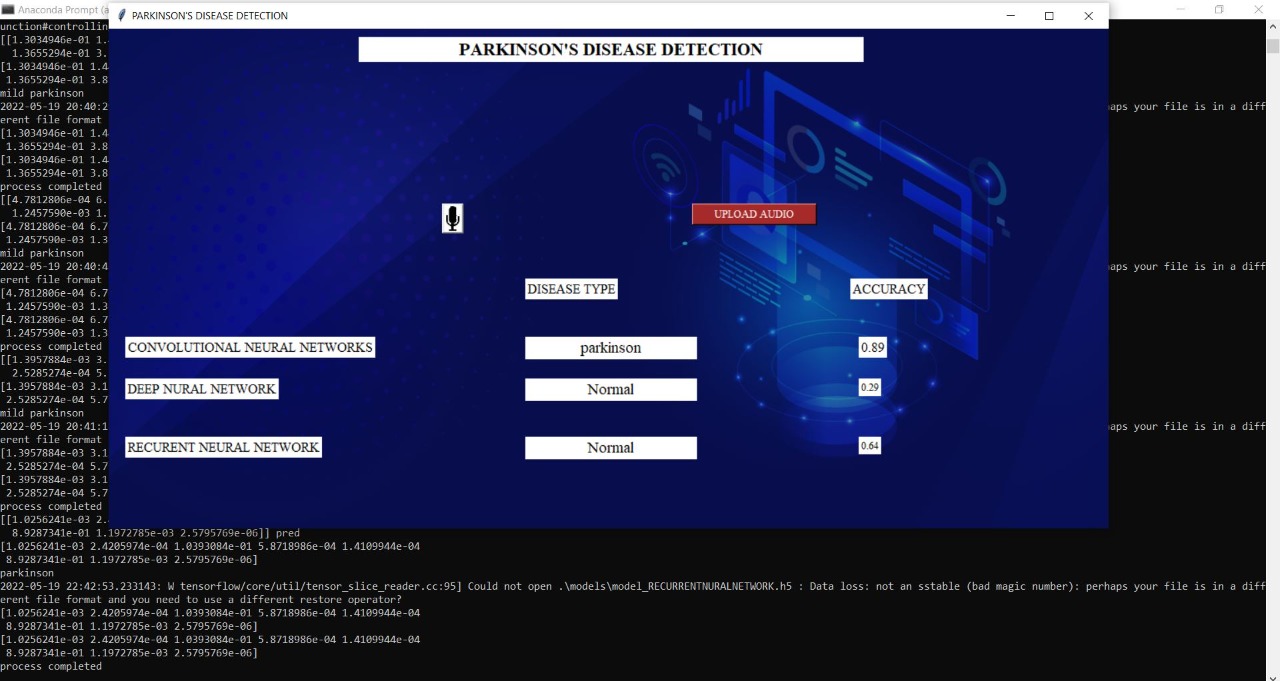
**A.1 SAMPLE SCREENS**



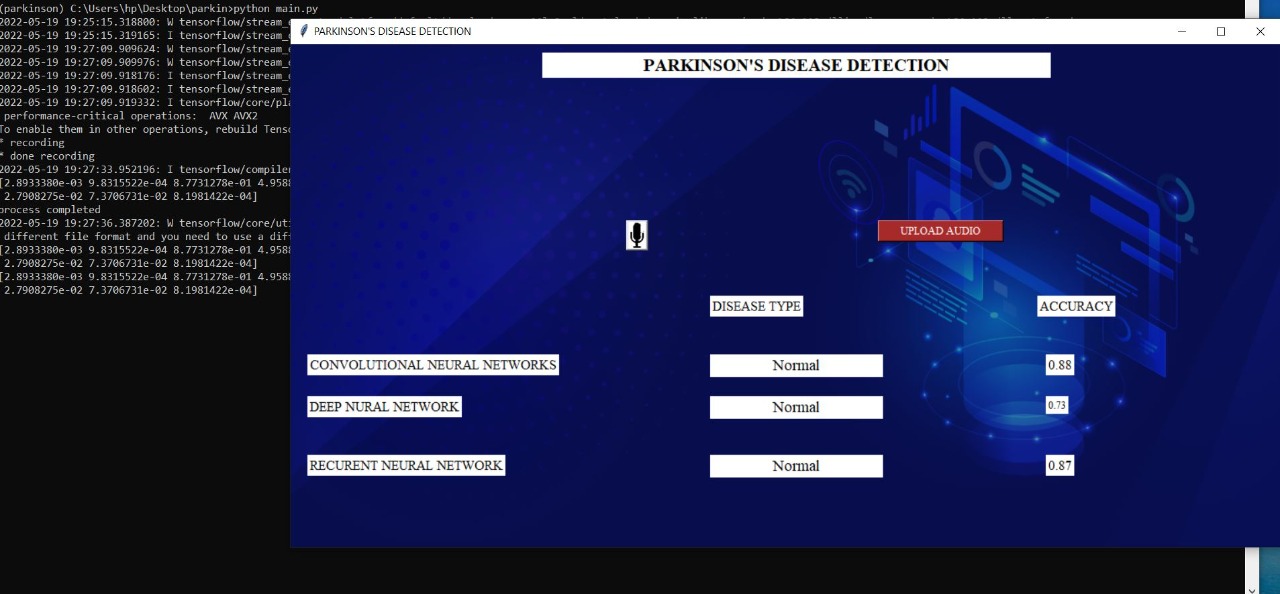
**FIG.A.1. HOMEPAGE**



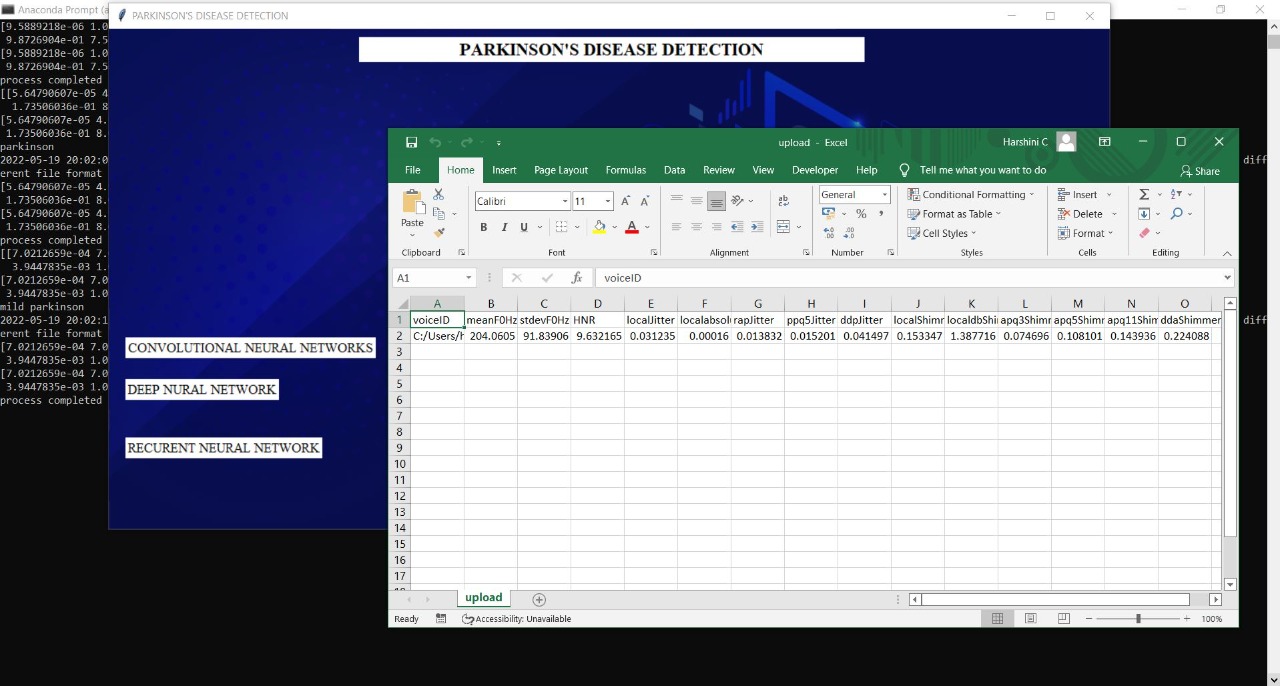
**FIG.A.2 UPLOAD AUDIO RESULT1**



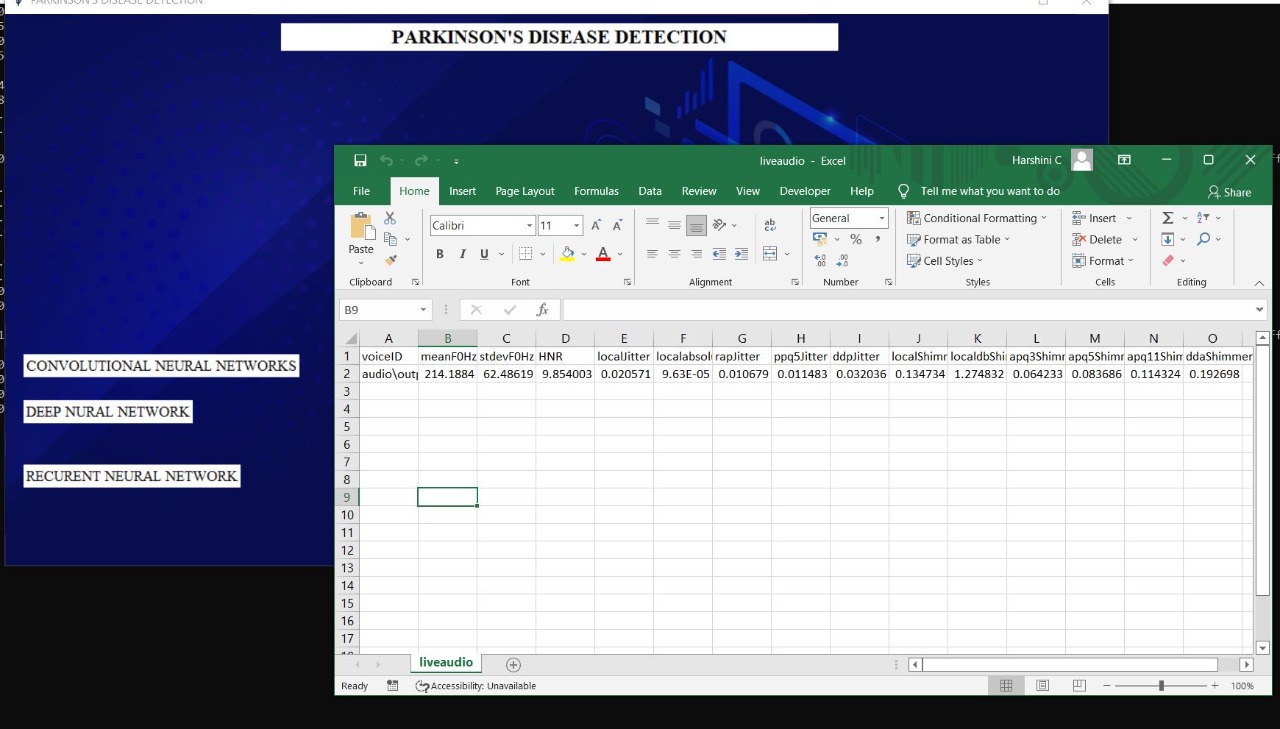
**FIG.A.3 UPLOAD AUDIO RESULT2**



**FIG.A.4 LIVE AUDIO RESULT**



**FIG.A.5. UPLOAD AUDIO VALUES**



**FIG.A.1.6. LIVE AUDIO VALUES**

**REFERENCES**

1. Amit Kumar Patra , Ratula Ray, Azian Azamimi Abdullah, Satya Ranjan Dash(2019),” *Prediction of Parkinson's disease using Ensemble Machine Learning classification from acoustic analysis “,* Journal of Physics: Conference Series, IOP publications , UK, Volume : 1372, NO:1742-6596/1372/1/012041 <https://iopscience.iop.org/article/10.1088/1742-6596/1372/1/012041/pdf>
2. Ahlem Kehili, dabbabi Karim, Cherif Adnen (2020) “*Early Parkinson Detection Using Fully-Connected Deep Neural Network based on Vocal Features “ ,*IJCSNS International Journal of Computer science and Network Security , University of Tunis , VOL.20 N0.6 June 2020 <https://www.researchgate.net/publication/343100129_Early_Parkinson_Detection_Using_Fully-Connected_Deep_Neural_Network_based_on_Vocal_Features>
3. U Anusri, G Dhatchayani, Y Princely Angelinal and S. Kamalraj(2021) *“An Early Prediction of Parkinson’s disease using facial emotional Recognition.”,* Journal of Physics 2021 J.Conf. Ser. 1937 012058 <https://iopscience.iop.org/article/10.1088/1742-6596/1937/1/012058>
4. Apoorva Safai, Nirvi Vakharia, Shweta Prasad, Jitender Saini, Apurva Shah, Abhishek Lenka, Pramod Kumar Pal, and Madhura*,” Multi-modal brain based prediction of Parkinson disease using Graph Attention Networks”,* Researchgate, Volume : 15 , article :748414 [https://www.researchgate.net/publication/358798991\_Multimodal\_Brain\_Connectomic Based\_Prediction\_of\_Parkinson's\_Disease\_Using\_Graph\_Attention\_Networks](https://www.researchgate.net/publication/358798991_Multimodal_Brain_Connectomic%20Based_Prediction_of_Parkinson's_Disease_Using_Graph_Attention_Networks)
5. Christos Laganas, Dimitrios Iakovakis, Stelios Hadjidimitriou, Vasileios Charisis, Sofia B. Dias, Sevasti Bostantzopoulou, Zoe Katsarou, Lisa Klingelhoefer, Heinz Reichmann, Dhaval Trivedi, K. Ray Chaudhuri, and Leontios J.Hadjileontiadis, Senior Member, IEEE(2021) “*Parkinson’s Disease Detection Based on Running Speech Data From Phone Calls”,* IEEE Transactions on Biomedical Engineering : DOI 10.1109/TBME.2021.3116935, <https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9556632>
6. M. Javed Mehedi Shamrat, Md. Asaduzzaman(2019),” *A Comparative Analysis Of Parkinson Disease Prediction Using Machine Learning Approaches”,* International Journal of Scientific and technology research, VOLUME 8, ISSUE 11, PN.NO – 2576-2580 <https://www.researchgate.net/publication/338924577_A_Comparative_Analysis_Of_Parkinson_Disease_Prediction_Using_Machine_Learning_Approaches>
7. Muntasir Hoq , Mohammed Nazim Uddin , Seung-Bo Park(2021)*,” Vocal Feature Extraction based Artificial Intelligence model for Parkinson disease detection”,*Pubmet, 2021 Jun 11 (6):1076. doi: 10.3390/diagnostics11061076. <https://pubmed.ncbi.nlm.nih.gov/34208330/>
8. Shubham Bind , Arvind Kumar Tiwari , Anil Kumar Sahani(2015),” *A Survey of Machine Learning Based Approaches for Parkinson Disease Prediction”* (IJCSIT) International Journal of Computer Science and Information Technologies, Department of CSE, KIT, Department of CSE, IIT (BHU) Varanasi, India , Vol. 6 (2) , 2015, 1648-1655 <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.735.5660&rep=rep1&type=pdf>
9. Srishti Grover, Saloni Bhartia, Akshama, Abhilasha Yadav, Seeja K.R(2018), “*Predicting Severity Of Parkinson’s Disease Using Deep Learning”,*International conference on Computational Intelligence and Data ,Department of Computer Science and Engineering Indira Gandhi Delhi Technical University for Women, Kashmere Gate, Delhi, 110006, India, Procedia Computer Science Volume 132 , Pages1788-1794,2018 <https://www.sciencedirect.com/science/article/pii/S1877050918308883>
10. J. Sujatha, Dr . S.P. Rajagopalan(2017), “*Performance Evaluation of Machine Learning Algorithms in the Classification of Parkinson Disease Using Voice Attributes “,* International Journal of Applied Engineering Research , India : ISSN 0973-4562 Volume 12, Number 21 <http://www.ripublication.com/ijaer17/ijaerv12n21_24.pdf>