**CODE WITH EXECUTION PROCEDURE**

**CODE:**

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 tensorflow as tf

import pandas as pd

import os

import cv2

import numpy as np

from sklearn import svm

from sklearn.metrics import accuracy\_score

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

from sklearn.decomposition import PCA

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 PIL import ImageTk, Image

from tensorflow.keras.regularizers import l2

#from tensorflow.keras.models import Sequential

main = tkinter.Tk()

main.title("Lung cancer detection using Machine Learning(Hybrid CNN-SVM model)")

main.geometry("1300x1200")

global filename

global classifier

global svm\_sr, cnn\_sr

global X, Y

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

global pca

def uploadDataset():

global filename

filename = filedialog.askdirectory(initialdir=".")

text.delete('1.0', END)

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

def splitDataset():

global X, Y

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

global pca

text.delete('1.0', END)

X = np.load('features/X.txt.npy')

Y = np.load('features/Y.txt.npy')

X = np.reshape(X, (X.shape[0],(X.shape[1]\*X.shape[2]\*X.shape[3])))

pca = PCA(n\_components = 100)

X = pca.fit\_transform(X)

print(X.shape)

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

text.insert(END,"Total CT Scan Images Found in dataset : "+str(len(X))+"\n")

text.insert(END,"Train split dataset to 80% : "+str(len(X\_train))+"\n")

text.insert(END,"Test split dataset to 20% : "+str(len(X\_test))+"\n")

def executeSVM():

global classifier

global svm\_sr

text.delete('1.0', END)

cls = svm.SVC()

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

predict = cls.predict(X\_test)

svm\_sr = accuracy\_score(y\_test,predict) \* 100

classifier = cls

text.insert(END,"SVM ACCURACY : "+str(svm\_sr)+"\n")

def executeHybridmodel():

global cnn\_sr

X = np.load('features/X.txt.npy')

Y = np.load('features/Y.txt.npy')

Y = to\_categorical(Y)

classifier = Sequential()

classifier.add(Convolution2D(32, 3, 3, input\_shape = (64, 64, 3), activation = 'relu'))

classifier.add(MaxPooling2D(pool\_size = (2, 2)))

classifier.add(Convolution2D(32, 3, 3, activation = 'relu'))

classifier.add(MaxPooling2D(pool\_size = (2, 2)))

classifier.add(Flatten())

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

classifier.add(Dense(2, kernel\_regularizer=tf.keras.regularizers.l2(0.01),activation ='linear'))

print(classifier.summary())

classifier.compile(optimizer = 'adam', loss = 'hinge', metrics = ['acc'])

hist = classifier.fit(X, Y, batch\_size=16, epochs=13, shuffle=True, verbose=2)

hist = hist.history

acc = hist['acc']

cnn\_sr = acc[12] \* 100

text.insert(END,"CNN-SVM ACCURACY : "+str(cnn\_sr)+"\n")

def predictCancer():

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

img = cv2.imread(filename)

img = cv2.resize(img, (64,64))

im2arr = np.array(img)

im2arr = im2arr.reshape(64,64,3)

im2arr = im2arr.astype('float32')

im2arr = im2arr/255

test = []

test.append(im2arr)

test = np.asarray(test)

test = np.reshape(test, (test.shape[0],(test.shape[1]\*test.shape[2]\*test.shape[3])))

test = pca.transform(test)

predict = classifier.predict(test)[0]

msg = ''

if predict == 0:

msg = "image is Normal(benign)"

if predict == 1:

msg = "image is Abnormal(malignant)"

img = cv2.imread(filename)

img = cv2.resize(img, (400,400))

cv2.putText(img, msg, (10, 25), cv2.FONT\_HERSHEY\_SIMPLEX,0.7, (0, 255, 255), 2)

cv2.imshow(msg, img)

cv2.waitKey(0)

def graph():

height = [svm\_sr, cnn\_sr]

bars = ('SVM ACCURACY','CNN-SVM ACCURACY')

y\_pos = np.arange(len(bars))

plt.bar(y\_pos, height)

plt.xticks(y\_pos, bars)

plt.show()

width= main.winfo\_screenwidth()

height= main.winfo\_screenheight()

main.geometry("%dx%d" % (width, height))

main.resizable(False, False)

imgTemp = Image.open("C:/Users/ADMIN/Desktop/majorproject/lung2.webp")

img2 = imgTemp.resize((1400,900))

img = ImageTk.PhotoImage(img2)

label = Label(main,image=img)

label.pack(side='top',fill=Y,expand=True)

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

title = Label(main, text=’LUNG CANCER DETECTION USING MACHINE LEARNING(HYBRID CNN-SVM MODEL)’)

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

title.config(font=font)

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

title.place(x=85,y=5)

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

text=Text(main,height=20,width=50)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=75,y=100)

text.config(font=font1)

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

uploadButton = Button(main, text="Upload Lung Cancer Dataset", command=uploadDataset)

uploadButton.place(x=70,y=550)

uploadButton.config(font=font1)

readButton = Button(main, text="Read & Split Dataset to Train & Test", command=splitDataset)

readButton.place(x=1100,y=550)

readButton.config(font=font1)

svmButton = Button(main, text="Execute SVM Algorithm", command=executeSVM)

svmButton.place(x=70,y=600)

svmButton.config(font=font1)

kmeansButton = Button(main, text="Execute Hybrid CNN-SVM Algorithm", command=executeHybridmodel)

kmeansButton.place(x=1100,y=600)

kmeansButton.config(font=font1)

predictButton = Button(main, text="Predict Lung Cancer", command=predictCancer)

predictButton.place(x=70,y=650)

predictButton.config(font=font1)

graphButton = Button(main, text="Accuracy Graph", command=graph)

graphButton.place(x=1100,y=650)

graphButton.config(font=font1)

main.config(bg='Black')  
main.mainloop()

**EXECUTION PROCEDURE:**

1. First, we need to import all the necessary packages and libraries which are used to create GUI and also which are needed to execute the code.
2. We need to create required buttons and their individual functionalities or activities for step-by-step execution of the code.
3. The first step of execution is uploading our datasets, after collecting the suitable datasets, we need to upload the datasets by clicking the button called upload datasets.
4. After our datasets are uploaded, we can see the statement called “Datasets are Uploaded” on or interface.
5. Next step is to split the datasets into training and testing set by using the button called “Split the datasets”
6. After splitting step, we can see that the Datasets are divided into 80% of training set and 20% of testing set.
7. Now we need to execute our Normal SVM algorithm and see the accuracy, by clicking the button called “Execute SVM”.
8. After executing the ideal SVM algorithm we can see its accuracy on our interface, after this we need to execute our proposed Hybrid CNN-SVM algorithm by using the button “Execute Hybrid CNN-SVM” and check for its accuracy.
9. After seeing their accuracies, we can see that Hybrid CNN-SVM is showing greater accuracy than SVM alone.
10. The last step is testing our image datasets for cancer or non-cancer results.
11. By uploading the test samples of lung X-ray images, we can see the result as Normal or Abnormal i.e benign or malignant.
12. The last part of the execution is comparison graph of Accuracies of SVM and Hybrid CNN-SVM.