MRI.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 cv2

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

from sklearn.metrics import accuracy\_score

import imutils

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

from sklearn import metrics

import ftplib

from tkinter import ttk

main = tkinter.Tk()

main.title("An Overview of Deep Learning in Medical Imaging Focusing on MRI") #designing main screen

main.geometry("1300x1200")

global filename

global accuracy

X = []

Y = []

global classifier

disease = ['No Tumor Detected','Tumor Detected']

def upload(): #function to upload tweeter profile

global filename

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

text.delete('1.0', END)

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

def generateModel():

global X

global Y

X.clear()

Y.clear()

if os.path.exists('Model/myimg\_data.txt.npy'):

X = np.load('Model/myimg\_data.txt.npy')

Y = np.load('Model/myimg\_label.txt.npy')

else:

for root, dirs, directory in os.walk(filename+"/no"):

for i in range(len(directory)):

name = directory[i]

img = cv2.imread(filename+"/no/"+name,0)

ret2,th2 = cv.threshold(img,0,255,cv.THRESH\_BINARY+cv.THRESH\_OTSU)

img = cv2.resize(img, (128,128))

im2arr = np.array(img)

im2arr = im2arr.reshape(128,128,1)

X.append(im2arr)

Y.append(0)

print(filename+"/no/"+name)

for root, dirs, directory in os.walk(filename+"/yes"):

for i in range(len(directory)):

name = directory[i]

img = cv2.imread(filename+"/yes/"+name,0)

ret2,th2 = cv.threshold(img,0,255,cv.THRESH\_BINARY+cv.THRESH\_OTSU)

img = cv2.resize(img, (128,128))

im2arr = np.array(img)

im2arr = im2arr.reshape(128,128,1)

X.append(im2arr)

Y.append(1)

print(filename+"/yes/"+name)

X = np.asarray(X)

Y = np.asarray(Y)

np.save("Model/myimg\_data.txt",X)

np.save("Model/myimg\_label.txt",Y)

print(X.shape)

print(Y.shape)

print(Y)

cv2.imshow('ss',X[20])

cv2.waitKey(0)

text.insert(END,"Total number of images found in dataset : "+str(len(X))+"\n")

text.insert(END,"Total number of classes : "+str(len(set(Y)))+"\n\n")

def CNN():

global accuracy

global classifier

YY = to\_categorical(Y)

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

np.random.shuffle(indices)

x\_train = X[indices]

y\_train = YY[indices]

if os.path.exists('Model/model.json'):

with open('Model/model.json', "r") as json\_file:

loaded\_model\_json = json\_file.read()

classifier = model\_from\_json(loaded\_model\_json)

classifier.load\_weights("Model/model\_weights.h5")

classifier.\_make\_predict\_function()

print(classifier.summary())

f = open('Model/history.pckl', 'rb')

data = pickle.load(f)

f.close()

acc = data['accuracy']

accuracy = acc[9] \* 100

text.insert(END,'\n\nCNN Model Generated. See black console to view layers of CNN\n\n')

text.insert(END,"CNN Prediction Accuracy on Test Images : "+str(accuracy)+"\n")

else:

X\_trains, X\_tests, y\_trains, y\_tests = train\_test\_split(x\_train, y\_train, test\_size = 0.2, random\_state = 0)

classifier = Sequential() #alexnet transfer learning code here

classifier.add(Convolution2D(32, 3, 3, input\_shape = (128, 128, 1), 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 = 128, activation = 'relu'))

classifier.add(Dense(output\_dim = 2, activation = 'softmax'))

print(classifier.summary())

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

hist = classifier.fit(x\_train, y\_train, batch\_size=16, epochs=10,validation\_split=0.2, shuffle=True, verbose=2)

classifier.save\_weights('Model/model\_weights.h5')

model\_json = classifier.to\_json()

with open("Model/model.json", "w") as json\_file:

json\_file.write(model\_json)

f = open('Model/history.pckl', 'wb')

pickle.dump(hist.history, f)

f.close()

f = open('Model/history.pckl', 'rb')

data = pickle.load(f)

f.close()

acc = data['accuracy']

accuracy = acc[9] \* 100

text.insert(END,'\n\nCNN Model Generated. See black console to view layers of CNN\n\n')

text.insert(END,"CNN Prediction Accuracy on Test Images : "+str(accuracy)+"\n")

def predict():

ftp = ftplib.FTP\_TLS("ftp.drivehq.com")

ftp.login("cloudfilestorageacademic", "Offenburg965#")

ftp.prot\_p()

name = imagelist.get()

with open('drivehq.jpg', 'wb' ) as file :

ftp.retrbinary('RETR %s' % name, file.write)

file.close()

img = cv2.imread('drivehq.jpg',0)

img = cv2.resize(img, (128,128))

im2arr = np.array(img)

im2arr = im2arr.reshape(1,128,128,1)

XX = np.asarray(im2arr)

predicts = classifier.predict(XX)

print(predicts)

cls = np.argmax(predicts)

print(cls)

img = cv2.imread('drivehq.jpg')

img = cv2.resize(img, (800,500))

cv2.putText(img, 'Disease Identified as : '+disease[cls], (10, 25), cv2.FONT\_HERSHEY\_SIMPLEX,0.7, (0, 255, 255), 2)

cv2.imshow('Disease Identified as : '+disease[cls], img)

cv2.waitKey(0)

def getImages():

ftp = ftplib.FTP\_TLS("ftp.drivehq.com")

ftp.login("cloudfilestorageacademic", "Offenburg965#")

ftp.prot\_p()

filenames = ftp.nlst()

value.clear()

for filename in filenames:

value.append(filename)

font = ('times', 16, 'bold')

title = Label(main, text='An Overview of Deep Learning in Medical Imaging Focusing on MRI')

title.config(bg='darkviolet', fg='gold')

title.config(font=font)

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

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

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

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

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=50,y=120)

text.config(font=font1)

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

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

uploadButton.place(x=50,y=550)

uploadButton.config(font=font1)

modelButton = Button(main, text="Generate Images Train & Test Model (OSTU Features)", command=generateModel)

modelButton.place(x=290,y=550)

modelButton.config(font=font1)

cnnButton = Button(main, text="Generate Deep Learning CNN Model", command=CNN)

cnnButton.place(x=710,y=550)

cnnButton.config(font=font1)

imageButton = Button(main, text="Get DriveHQ Images", command=getImages)

imageButton.place(x=50,y=600)

imageButton.config(font=font1)

value = ["DriveHQ Images"]

imagelist = ttk.Combobox(main,values=value,postcommand=lambda: imagelist.configure(values=value))

imagelist.place(x=240,y=600)

imagelist.current(0)

imagelist.config(font=font1)

predictButton = Button(main, text="Predict Tumor", command=predict)

predictButton.place(x=440,y=600)

predictButton.config(font=font1)

main.config(bg='turquoise')

main.mainloop()

test.py

import ftplib

import urllib

'''

session = ftplib.FTP\_TLS('ftp.drivehq.com','cloudfilestorageacademic','Offenburg965#')

file = open('test.txt','rb') # file to send

session.storbinary('test.txt', file) # send the file

file.close() # close file and FTP

session.quit()

'''

ftp = ftplib.FTP\_TLS("ftp.drivehq.com")

ftp.login("cloudfilestorageacademic", "Offenburg965#")

ftp.prot\_p()

'''

file = open("test.txt", "rb")

ftp.storbinary("STOR test.txt", file)

file.close()

ftp.close()

'''

filenames = ftp.nlst()

for filename in filenames:

with open( filename, 'wb' ) as file :

ftp.retrbinary('RETR %s' % filename, file.write)

file.close()