MINI PROJECT
(2020-21)

A Deep Learning approach for the early diagnosis of Parkinson's disease using Brain MRI Scans

MID-TERM REPORT

Institute of Engineering & Technology

Submitted by
Rishik Mishra
(181500569)

Supervised By: -

Dr. Anand Singh Jalal
H.O.D
Department of Computer Engineering & Applications

Contents

Abstract 3

Introduction 3

General introduction to the topic 3

Area of computer science 4

Hardware and software requirements 4

Problem Definition 5

Objectives 5

Implementation Details 5

Progress Till Date and the Remaining work 9

Screenshots 11

References 16

Abstract

Parkinson's disease is a neurological disorder with more than 6 million people worldwide suffering from it.It is commonly diagnosed using clinical assessments and progression scale which usually depends on the medical practitioner's expertise ,and accuracy varies greatly between various examiners which also takes a long time to accurately diagnose.This paper proposes to develop a computer aided diagnostic method to diagnose PD patients using MRI images of the brain ,thus reducing the time required to accurately differentiate between PD and Control subjects

Introduction

1.1 General Introduction to the topic

Parkinson's disease is a progressive neurodegenerative disorder that affects the motor as well as non motor functions. These present themselves in the form of tremors, bradykinesia, loss of balance, weakened sense of smell, hallucination and other motor / non motor symptoms [1].These characteristic motor symptoms provide the basis for clinical PD diagnosis. Parkinson's disease is widespread with 6.1 million individuals worldwide suffering from the disease in 2016, 2.9 million being women and 3.2 million men.This number is set to increase as people live longer[2].

The large population suffering from Parkinson's disease provides us with an urgency to provide a reliable system for the early diagnosis of Parkinson's disease, as no approved cure exists for the disease[3] .Early diagnosis of PD (before the apparent motor symptoms present themselves in a more violent manner) is essential to start the treatment of the disease before further incurable neurodegeneration has taken place[4].

Parkinson's progression can be classified into five following phases,according to the Hoehn and Yahr rating scale[1] as follows :

Stage 1 : at this stage the symptoms are not very noticeable ,but there can be changes in posture,walk or facial expressions and these symptoms are generally limited to one side of the body
Stage 2 : tremors,stiffness,change in facial expression on both side of the body ,without impairment of balance is characteristic of the second stage
Stage 3 : all the symptoms of stage 2 along with loss of balance
Stage 4 : violent tremors,bradykinesia ,loss of balance and requiring assistive devices
Stage 5 : stiffness in the legs,patient becomes wheelchair bound
It is necessary to diagnose Parkinson's disease before the disease progresses to Stage 3 ,as until stage 3 is reached ,the patient is still able to function independently .Advances in treatment techniques for parkinson's can enable the patient to maintain a sufficiently high quality of life if diagnosed earlier and further advancements in therapy techniques while the disease is diagnosed in the 1st or 2nd stage can even lead to subsequent neuronal loss in the substantia nigra region to be prevented entirely[13]
An early diagnosis (before the tumors or olfactory symptoms set in) gives the medical professionals a chance to control the further degeneration of the dopaminergic regions of the brain, thus preventing advanced symptoms to disturb the basic motor functions. Presently the diagnosis of Parkinson's disease is carried out by senior neurologists using behaviour monitoring and supporting tests to rule out other similar disorders. The best diagnostic test available for Parkinson's diagnosis is the DaTscan which is only available in specialised centers and very few technicians are available who can manage the process[7].

1.3 Hardware Requirements

- Memory [16GB RAM (or higher)]
- Intel core i7 64-bit Processor (or higher)
- Nvidia RTX 2080 ti (or higher)

1.3 Software requirements

- Tensorflow 2.0
- Keras
- Google colab
- Scikit-learn
- Seaborn
- Google drive
- Matplotlib
- Tensorboard
- openCV

Problem Definition

About 1% of the population over 60 years old are affected by Parkinson's Disease (PD)[11]. Hence, early diagnosis of the PD is essential for better treatment.The diagnosis of Parkinson's disease requires experienced neurologists and behavioral pattern tracking and takes a long time .A quicker way is DATScan which isn't available in the majority of the places.
The MRI scans can provide a visual biomarker for the classification of PD which can be utilized by new approaches like neural networks for faster and robust classification.In this study we are using MRI 3T scans and Convolutional Neural Networks to classify between PD and Control based on visual clues provided in the SN region of the brain.

Objective

We aim to develop a CNN based CAD system which classifies between PD and Healthy patients ,by utilising the differences between the substantia nigra region of the brain ,with one difference being the swallow signs in the nigrosome-1.this would improve upon the current systems by focussing more on the useful features.

Implementation Details

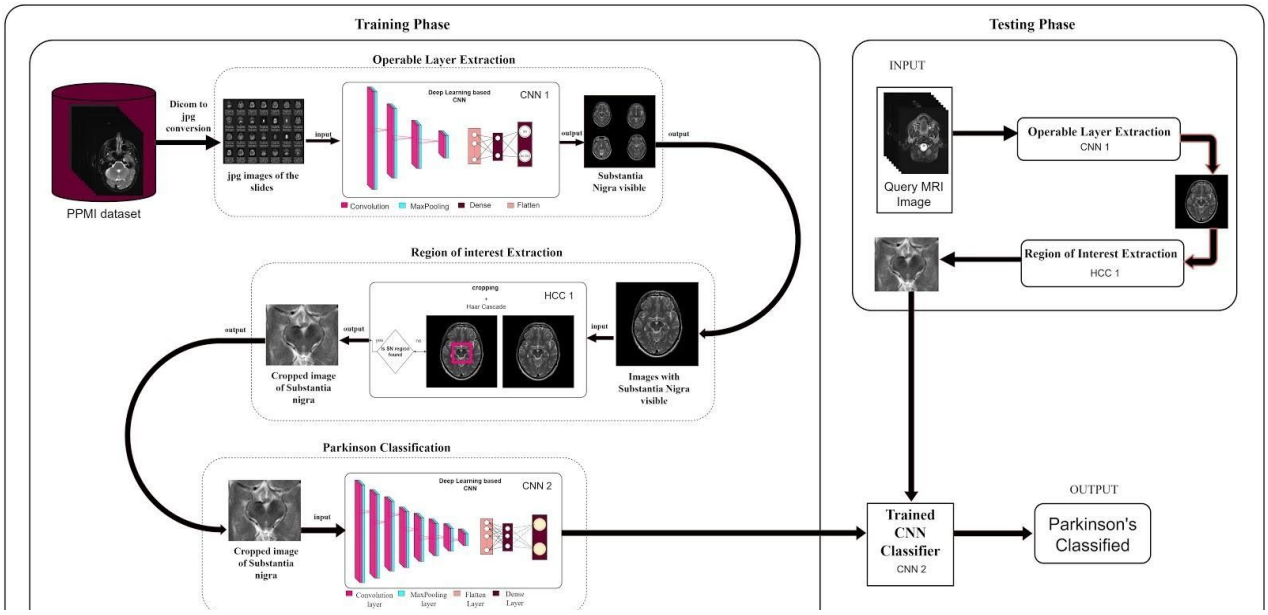


Fig: 1.0 Complete Process Diagram

DICOM to Lossless JPEG Conversion:

The image format used in medical imaging dataset is dicom which is not supported by the image processing frameworks and libraries which requires the conversion from .dicom to lossless jpeg format which can be used to store images in a lossless format and is flexible and compatible across multiple systems and operating systems. The dicom and pylab libraries are used to convert the dicom to jpeg format in python.
For training purposes each patient's images are converted from dicom to jpeg and stored in a combined dataset which is segregated in pd or control.

Image Enhancement:

The images present in the dataset have varying brightness,colour and noise ,to remove these unwanted elements from our training and testing images ,we apply image filtering operations and histogram equalisation for contrast enhancement ,for better identifiable features.
In our study the image enhancement pipeline consists of converting the RGB image to YUV colour space, for accurate colour and features ,then the Luminance channel is filtered using gaussian blur to reduce the noise and pixelation,then contrast limited adaptive histogram equalisation is applied on the Y channel to improve the local image contrast while keeping the noise low.
finally the the 3 channels Y,U and V are merged and converted to RGB colorspace for further processing

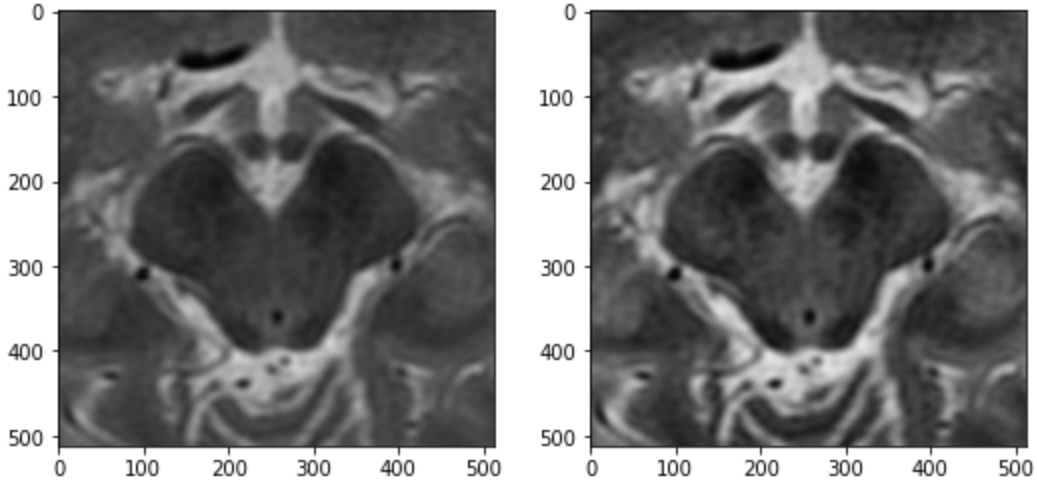


fig 1.2 . image of substantia nigra before (left) and after applying image enhancement (right)

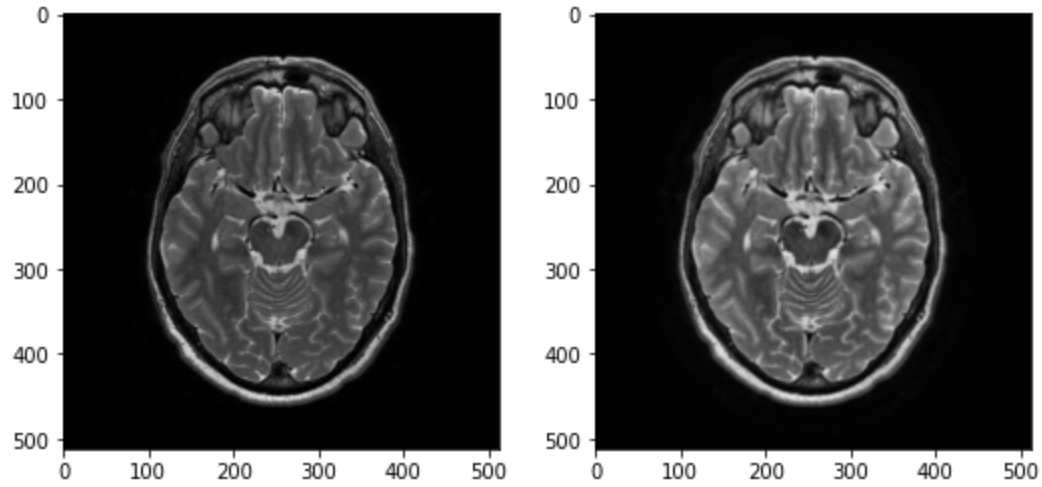


fig 1.3 . image of brain MRI before (top) and after applying the image enhancement techniques (bottom)

Region of Interest Extraction:

The images that the MRI dataset provides contain multiple slides of the brain and different studies might use different thickness of the slides . the most influencing region of the brain in the
detection of Parkinson's is the SN region [12], we localize the sn region to classify between pd and no pd ,to improve the accuracy of the localization we separate the images containing the SN region ,In this study we use a custom CNN for automatically differentiating SN in image and no sn in image .the image is processed through the convolutional layer with 8 filters and a kernel size of 3,3 which passes onto s max pooling layer of 2,2 and onto 4 similar constructs with filters growing as 16,32,32,64 with the kernel size remaining the same and a max pooling layer following each convolutional layer. The resulting features are passed onto 2 fully connected layers with 256 and 128 nodes respectively. The activation function 'relu' is used in all the conv2d layers.
Relu or Rectified linear function is a simple function that is non linear but behaves similarly to linear function,thus using stochastic gradient descent while diminishing backpropagation errors.
The rectified linear activation function is a simple calculation that returns the value provided as input directly, or the value 0.0 if the input is 0.0 or less.

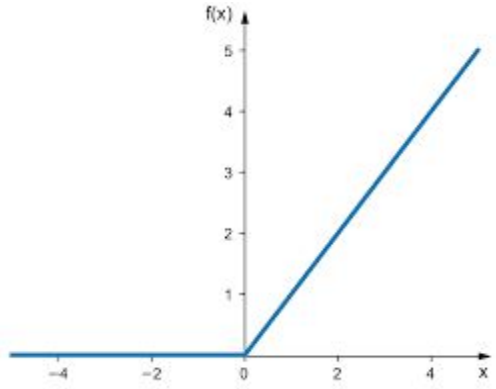


fig 1.4 .Visualising ReLu for negative and positive values

The Swallow signs which we are using to classify Parkinson's is present in the Substantia nigra pars compacta, we need to extract the substantia nigra region from the image to pass into the classifier, which is done by cropping the middle 170*170 regions from the positively classified images from the CNN1 and applying a haar cascade classifier to confirm the presence of the SN region in our cropped image.

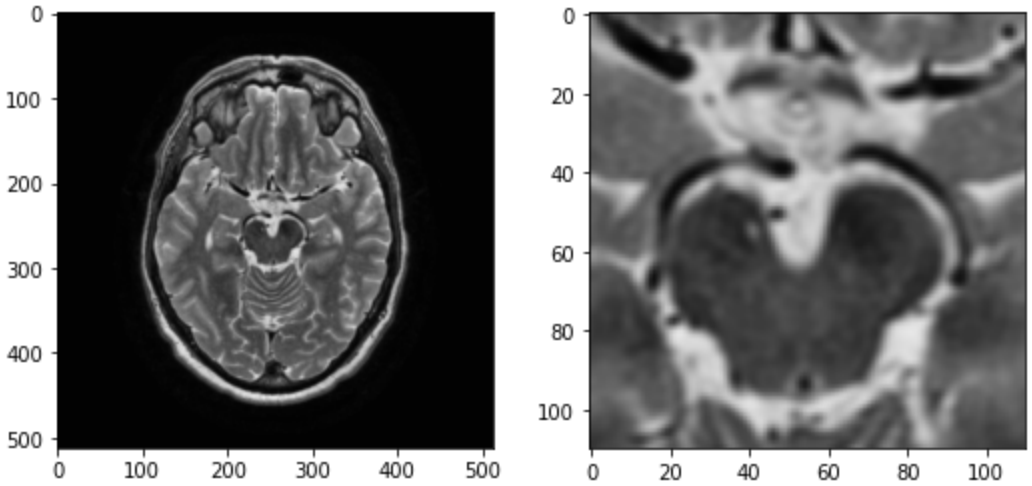


fig 1.5 . a. example of extracted brain slide(left) and b. example of cropping the region of interest(right)

Parkinson's Classification:

The final stage of the process is the classification of the Substantia nigra region as belonging to a Parkinson's patient or a control. A modified version of the Alex Net with activation at the last fully connected layer having a sigmoid activation for the 2 classes and 6 convolutional layers ,to improve the feature vectors as compared to manually selected vectors, which are prone to changes in orientation and intensity.

An image of the Substantia Nigra region in the MRI of size 512x512 is taken as an input by the convolutional neural network which can extract the features from the image autonomously for the classification of the image into the classes as PD or control.

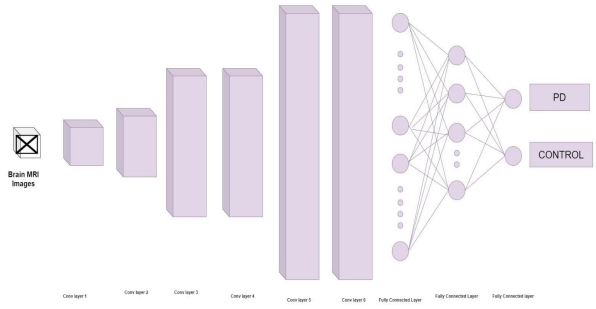


fig 1.6 . convolutional neural network architecture for phase II

The convolutional network uses 6 convolutional layers of filters as 16,32,32,64,64,128 with each layer followed by a max pooling layer;the result of these convolution layers is passed onto 4 fully connected layers and finally onto a sigmoid non linear function.
Sigmoid function lies between (0 and 1),thus giving us the probability of the result.we use sigmoid instead of softmax as we have to make binary classification and not multiclass classification.
the logistic sigmoid function can be represented as :

$$f(x) = \frac{1}{1 + e^{-x}}$$

This final phase returns the class of the selected image and finds the mean of the multiple layers that contain substantia nigra and returns the probability of the person having Parkinson's disease. Using a CNN instead of just comparing the swallow signs is done to not just limit the classification to the swallow signs, which can be blurred or sometimes not clearly recorded, thus the CNN is used to include multiple features of the substantia nigra region, which could be, area, distortion, shape and other changes across the region. This step is performed based on the highest impact of SN region on Parkinson's detection in an MRI image[12].

Progress

Phase I is complete

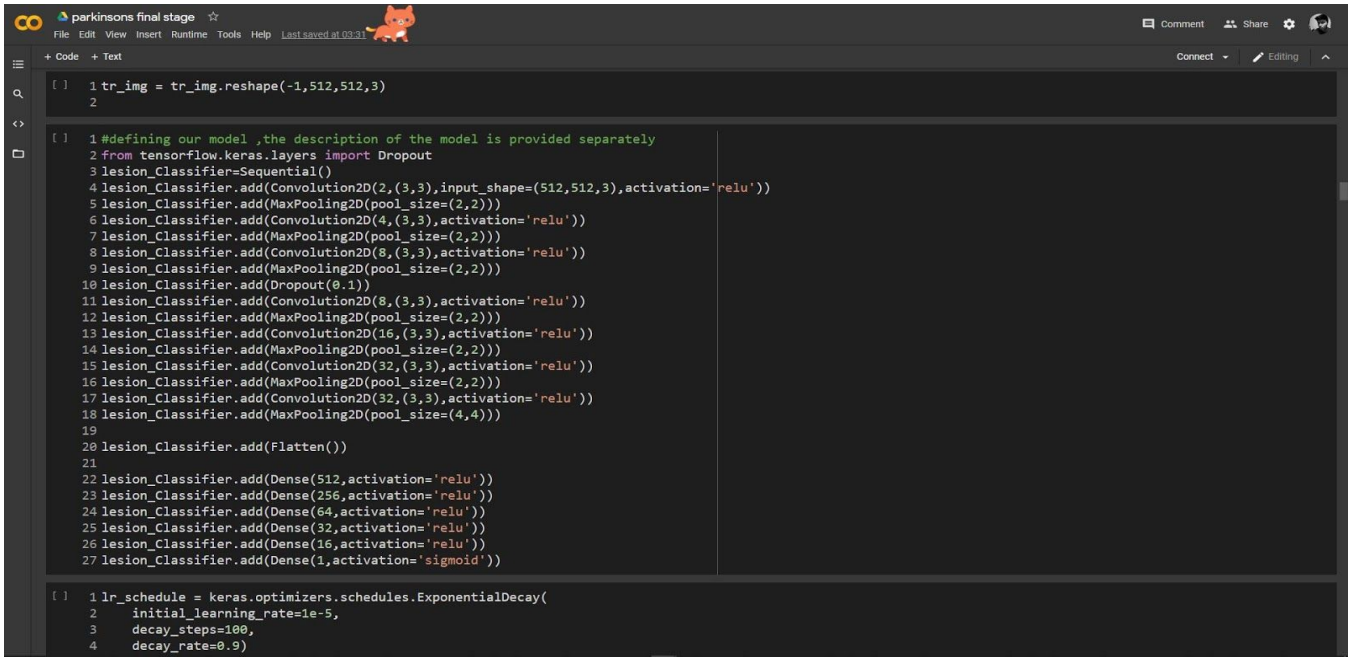
1. Dataset has been obtained
2. Dataset has been cleaned
3. DICOM images converted to Lossless JPG
4. Dataset images preprocessed
5. Substantia nigra regions localised and extracted by training the CNN1

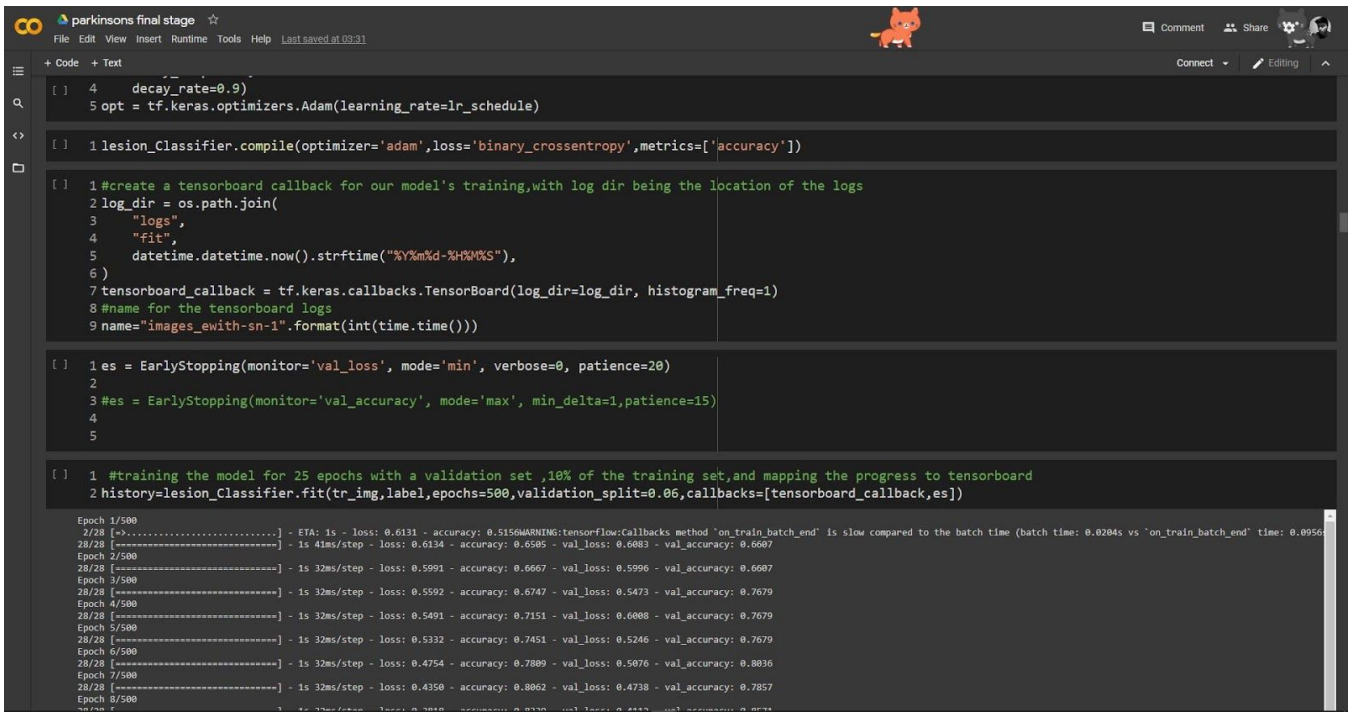
6. Substantia nigra images have been used to train CNN2
7. Trained CNN2 has been used to classify images

Work still left :

1. Test the effectiveness of transfer learning in our problem
2. Improve the model architecture to better suit the data
3. Improve the performance of the code

SCREENSHOTS





parkinsons final stage

File Edit View Insert Runtime Tools Help Last saved at 03:31

Comment Share

+ Code + Text

Connect Editing

```
Model: "sequential_21"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_180 (Conv2D)          (None, 510, 510, 2)       56
_________________________________________________________________
max_pooling2d_81 (MaxPooling (None, 255, 255, 2)       0
_________________________________________________________________
conv2d_181 (Conv2D)          (None, 253, 253, 4)       76
_________________________________________________________________
max_pooling2d_82 (MaxPooling (None, 126, 126, 4)       0
_________________________________________________________________
conv2d_182 (Conv2D)          (None, 124, 124, 8)       296
_________________________________________________________________
max_pooling2d_83 (MaxPooling (None, 62, 62, 8)         0
_________________________________________________________________
dropout_14 (Dropout)         (None, 62, 62, 8)         0
_________________________________________________________________
conv2d_183 (Conv2D)          (None, 60, 60, 8)         584
_________________________________________________________________
max_pooling2d_84 (MaxPooling (None, 30, 30, 8)         0
_________________________________________________________________
conv2d_184 (Conv2D)          (None, 28, 28, 16)        1168
_________________________________________________________________
max_pooling2d_85 (MaxPooling (None, 14, 14, 16)        0
_________________________________________________________________
conv2d_185 (Conv2D)          (None, 12, 12, 32)        4640
_________________________________________________________________
max_pooling2d_86 (MaxPooling (None, 6, 6, 32)          0
_________________________________________________________________
conv2d_186 (Conv2D)          (None, 4, 4, 32)          9248
_________________________________________________________________
max_pooling2d_87 (MaxPooling (None, 1, 1, 32)          0
_________________________________________________________________
flatten_20 (Flatten)         (None, 32)                0
_________________________________________________________________
dense_81 (Dense)             (None, 512)               16896
_________________________________________________________________
dense_82 (Dense)             (None, 256)               131328
_________________________________________________________________
dense_83 (Dense)             (None, 64)                16448
_________________________________________________________________
dense_84 (Dense)             (None, 32)                2080
_________________________________________________________________
dense_85 (Dense)             (None, 16)                528
_________________________________________________________________
dense_86 (Dense)             (None, 1)                 17
=================================================================
Total params: 183,365
Trainable params: 183,365
Non-trainable params: 0
```

parkinsons final stage

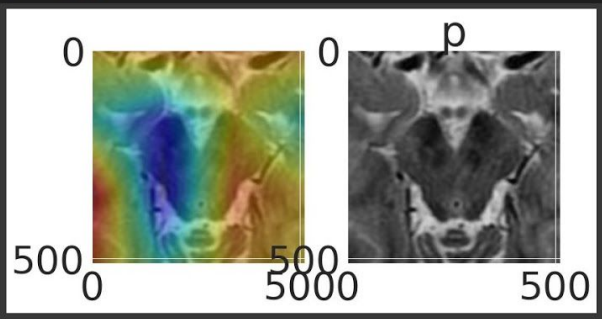
File Edit View Insert Runtime Tools Help Last saved at 03:31

Comment Share

+ Code + Text

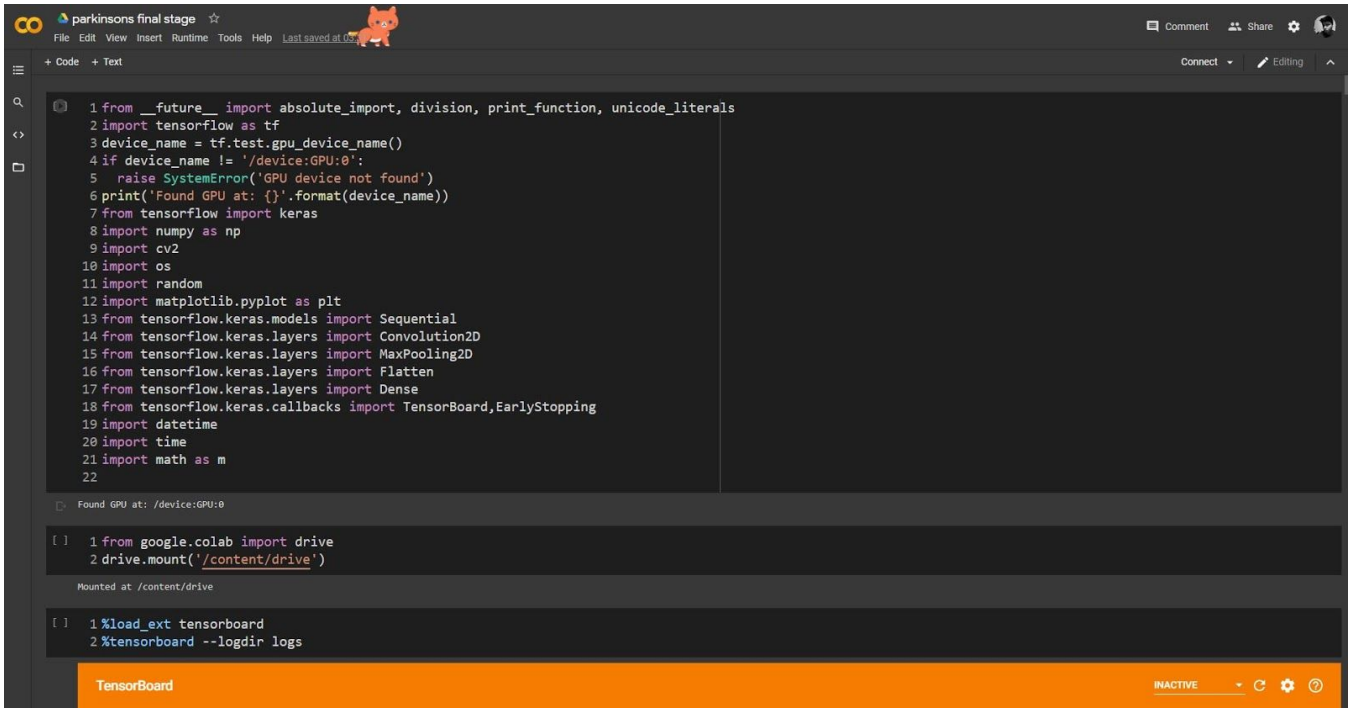
Connect Editing

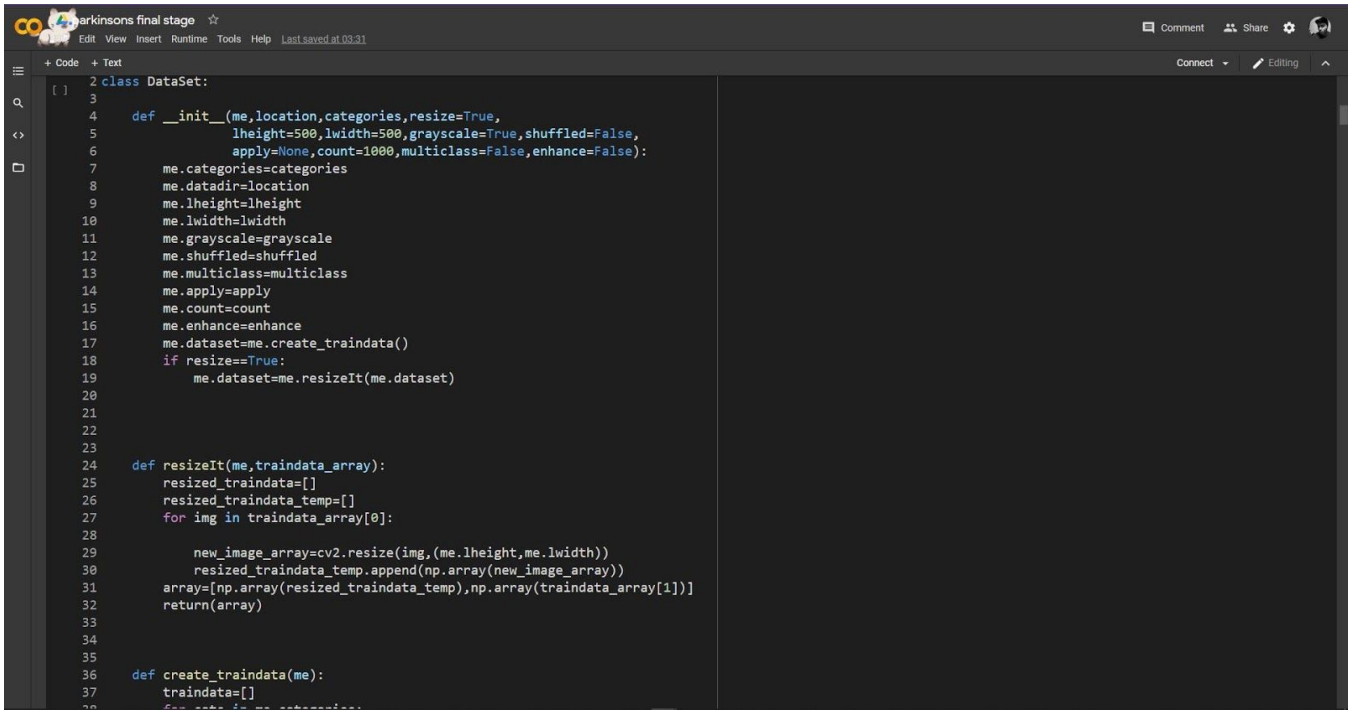
```
20    plt.imshow(cam,cmap='jet',alpha=0.4)
21    plt.subplot(1,2,2)
22    plt.imshow(data[0][ni])
23    plt.title(classnames)
24    plt.show()
25    break
```

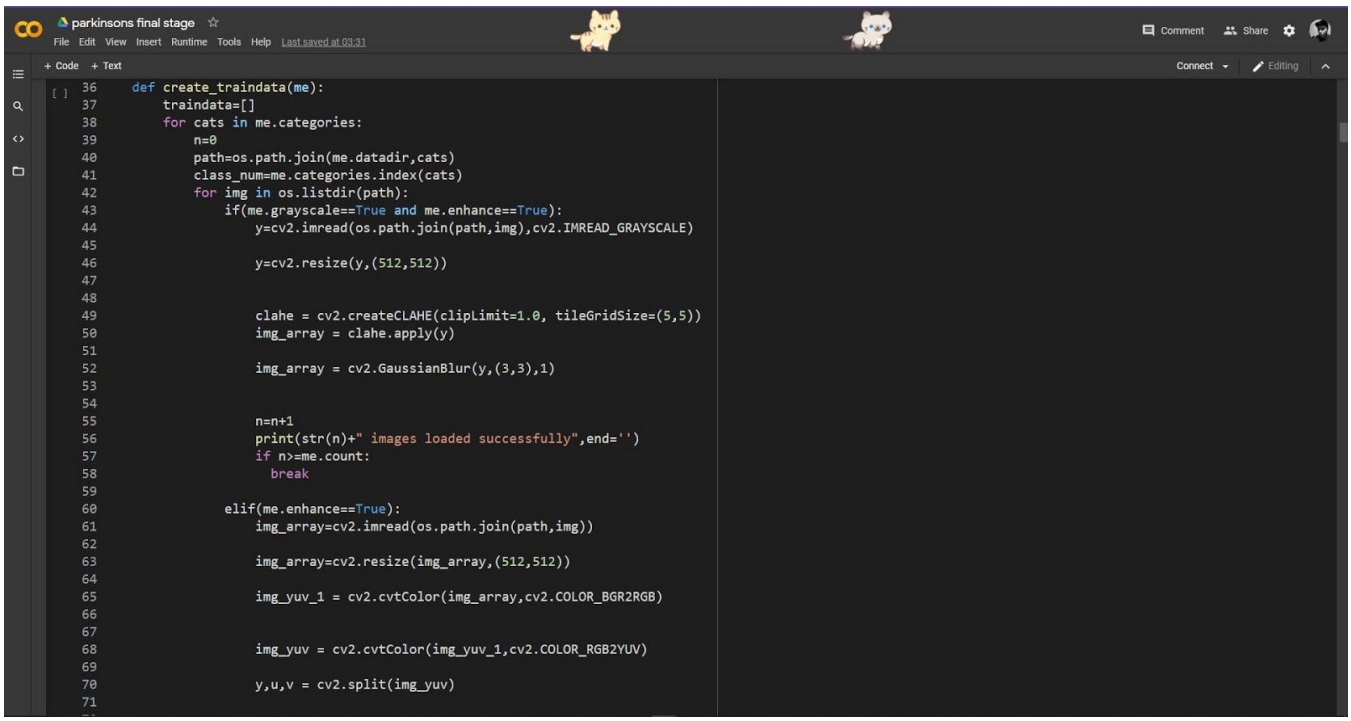



```
1 import numpy as np
2 import tensorflow as tf
3 from tensorflow import keras
4
5 # Display
6 from IPython.display import Image
7 import matplotlib.pyplot as plt
8 import matplotlib.cm as cm
```

```
1 model_builder = keras.applications.xception.Xception
```


parkinsons final stage
File Edit View Insert Runtime Tools Help Last saved at 03:31
Comment Share
+ Code + Text
Connect Editing
1 from __future__ import absolute_import, division, print_function, unicode_literals
2 import tensorflow as tf
3 device_name = tf.test.gpu_device_name()
4 if device_name != '/device:GPU:0':
5 raise SystemError('GPU device not found')
6 print('Found GPU at: {}'.format(device_name))
7 from tensorflow import keras
8 import numpy as np
9 import cv2
10 import os
11 import random
12 import matplotlib.pyplot as plt
13 from tensorflow.keras.models import Sequential
14 from tensorflow.keras.layers import Convolution2D
15 from tensorflow.keras.layers import MaxPooling2D
16 from tensorflow.keras.layers import Flatten
17 from tensorflow.keras.layers import Dense
18 from tensorflow.keras.callbacks import TensorBoard,EarlyStopping
19 import datetime
20 import time
21 import math as m
22
Found GPU at: /device:GPU:0
1 from google.colab import drive
2 drive.mount('/content/drive')
Mounted at /content/drive
1 %load_ext tensorboard
2 %tensorboard --logdir logs
TensorBoard
INACTIVE


parkinsons final stage
File Edit View Insert Runtime Tools Help Last saved at 03:31
Comment Share
+ Code + Text
Connect Editing
2 class DataSet:
3
4 def __init__(me,location,categories,resize=True,
5 lheight=500,lwidth=500,grayscale=True,shuffled=False,
6 apply=None,count=1000,multiclass=False,enhance=False):
7 me.categories=categories
8 me.datadir=location
9 me.lheight=lheight
10 me.lwidth=lwidth
11 me.grayscale=grayscale
12 me.shuffled=shuffled
13 me.multiclass=multiclass
14 me.apply=apply
15 me.count=count
16 me.enhance=enhance
17 me.dataset=me.create_traindata()
18 if resize==True:
19 me.dataset=me.resizeIt(me.dataset)
20
21
22
23
24 def resizeIt(me,traindata_array):
25 resized_traindata=[]
26 resized_traindata_temp=[]
27 for img in traindata_array[0]:
28
29 new_image_array=cv2.resize(img,(me.lheight,me.lwidth))
30 resized_traindata_temp.append(np.array(new_image_array))
31 array=[np.array(resized_traindata_temp),np.array(traindata_array[1])]
32 return(array)
33
34
35
36 def create_traindata(me):
37 traindata=[]



parkinsons final stage

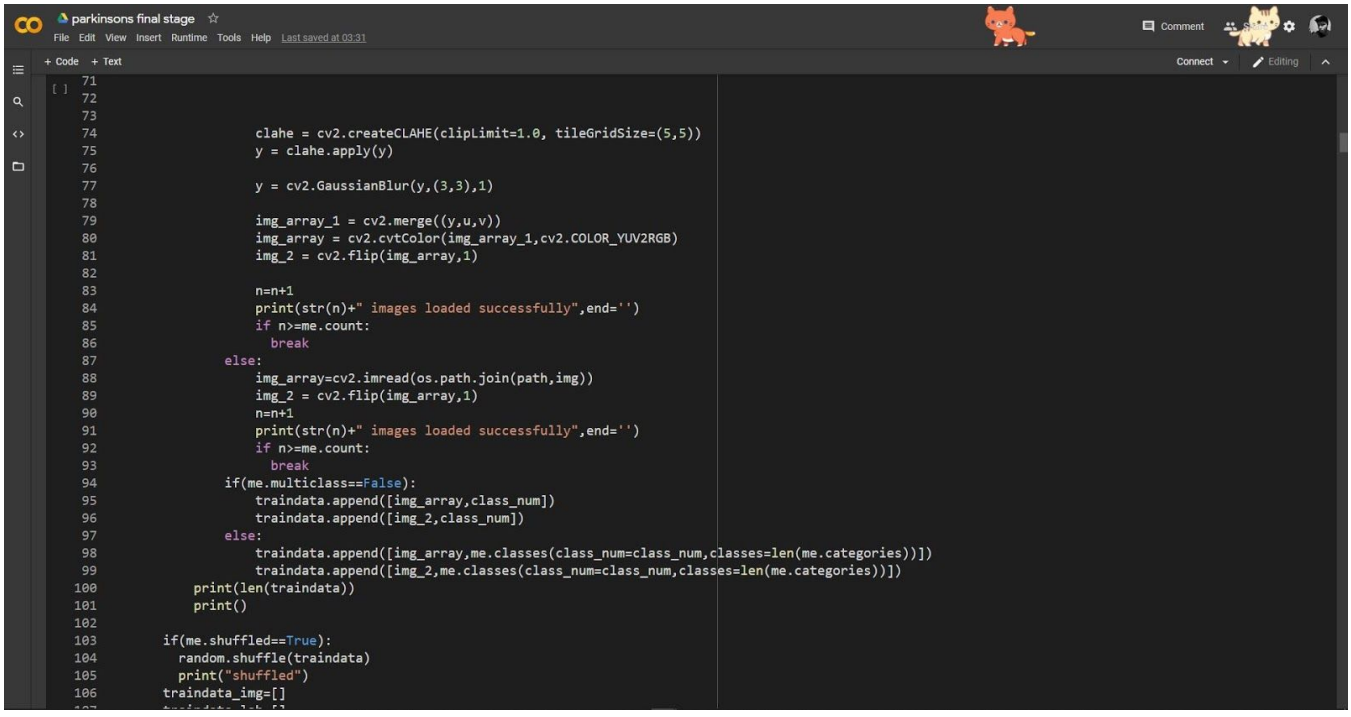
File Edit View Insert Runtime Tools Help Last saved at 03:31

Comment Share

+ Code + Text

Connect Editing

```
[ ]
def create_traindata(me):
    traindata=[]
    for cats in me.categories:
        n=0
        path=os.path.join(me.datadir,cats)
        class_num=me.categories.index(cats)
        for img in os.listdir(path):
            if(me.grayscale==True and me.enhance==True):
                y=cv2.imread(os.path.join(path,img),cv2.IMREAD_GRAYSCALE)

                y=cv2.resize(y,(512,512))


                clahe = cv2.createCLAHE(clipLimit=1.0, tileGridSize=(5,5))
                img_array = clahe.apply(y)

                img_array = cv2.GaussianBlur(y,(3,3),1)


                n=n+1
                print(str(n)+" images loaded successfully",end='')
                if n>=me.count:
                  break

            elif(me.enhance==True):
                img_array=cv2.imread(os.path.join(path,img))

                img_array=cv2.resize(img_array,(512,512))

                img_yuv_1 = cv2.cvtColor(img_array,cv2.COLOR_BGR2RGB)


                img_yuv = cv2.cvtColor(img_yuv_1,cv2.COLOR_RGB2YUV)

                y,u,v = cv2.split(img_yuv)
```



parkinsons final stage

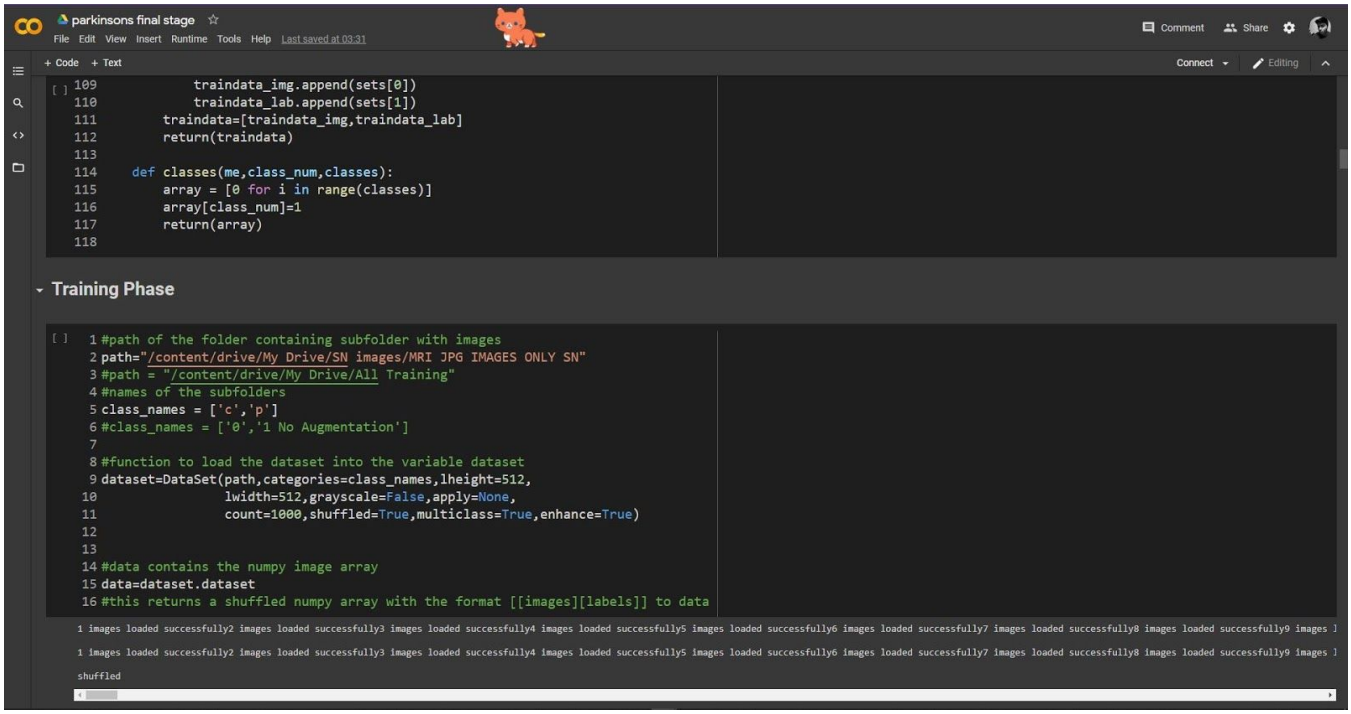
File Edit View Insert Runtime Tools Help Last saved at 03:31

Comment Share

+ Code + Text

Connect Editing

```
[ ]


                clahe = cv2.createCLAHE(clipLimit=1.0, tileGridSize=(5,5))
                y = clahe.apply(y)

                y = cv2.GaussianBlur(y,(3,3),1)

                img_array_1 = cv2.merge((y,u,v))
                img_array = cv2.cvtColor(img_array_1,cv2.COLOR_YUV2RGB)
                img_2 = cv2.flip(img_array,1)

                n=n+1
                print(str(n)+" images loaded successfully",end='')
                if n>=me.count:
                  break
            else:
                img_array=cv2.imread(os.path.join(path,img))
                img_2 = cv2.flip(img_array,1)
                n=n+1
                print(str(n)+" images loaded successfully",end='')
                if n>=me.count:
                  break
            if(me.multiclass==False):
                traindata.append([img_array,class_num])
                traindata.append([img_2,class_num])
            else:
                traindata.append([img_array,me.classes(class_num=class_num,classes=len(me.categories))])
                traindata.append([img_2,me.classes(class_num=class_num,classes=len(me.categories))])
        print(len(traindata))
        print()

    if(me.shuffled==True):
      random.shuffle(traindata)
      print("shuffled")
    traindata_img=[]
```



parkinsons final stage

File Edit View Insert Runtime Tools Help Last saved at 03:31

Comment Share

+ Code + Text

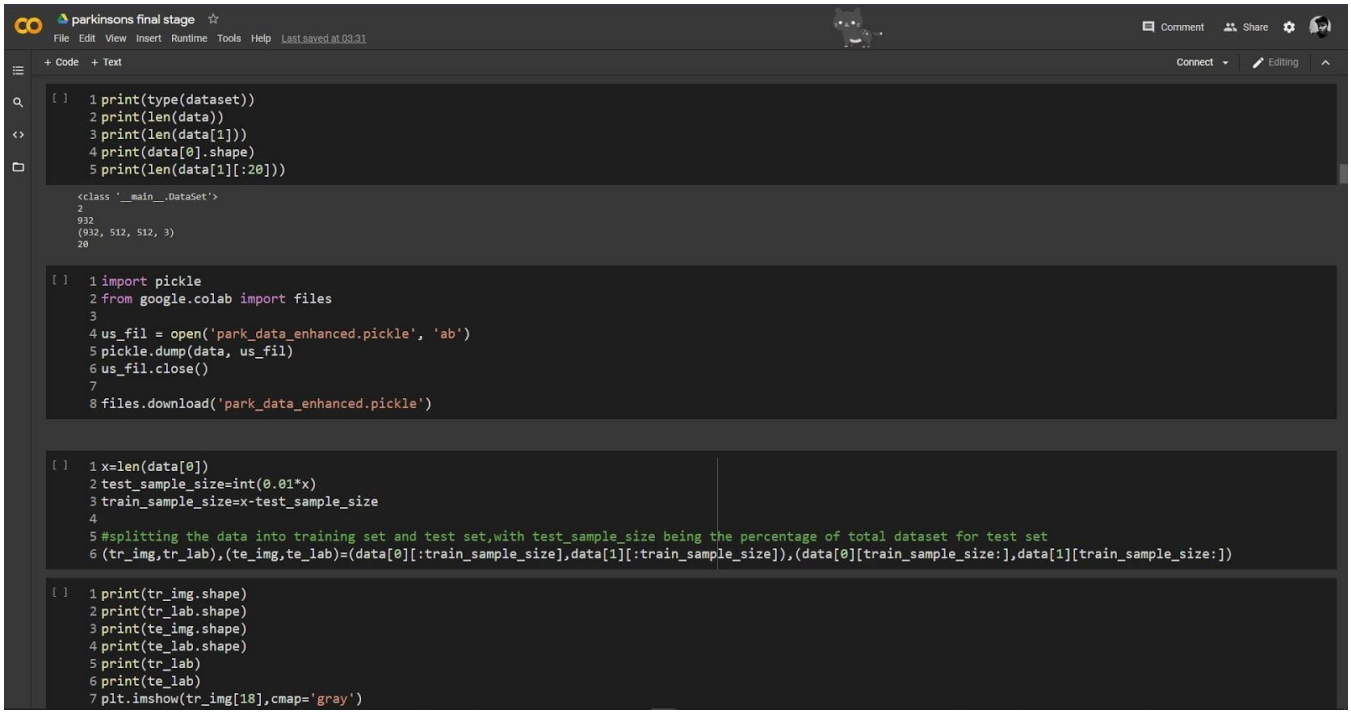
Connect Editing

```
            traindata_img.append(sets[0])
            traindata_lab.append(sets[1])
        traindata=[traindata_img,traindata_lab]
        return(traindata)

    def classes(me,class_num,classes):
        array = [0 for i in range(classes)]
        array[class_num]=1
        return(array)
```

Training Phase

```
#path of the folder containing subfolder with images
path="/content/drive/My Drive/SN images/MRI JPG IMAGES ONLY SN"
#path = "/content/drive/My Drive/All Training"
#names of the subfolders
class_names = ['c','p']
#class_names = ['0','1 No Augmentation']

#function to load the dataset into the variable dataset
dataset=DataSet(path,categories=class_names,lheight=512,
                lwidth=512,grayscale=False,apply=None,
                count=1000,shuffled=True,multiclass=True,enhance=True)


#data contains the numpy image array
data=dataset.dataset
#this returns a shuffled numpy array with the format [[images][labels]] to data
```

```
1 images loaded successfully2 images loaded successfully3 images loaded successfully4 images loaded successfully5 images loaded successfully6 images loaded successfully7 images loaded successfully8 images loaded successfully9 images l

1 images loaded successfully2 images loaded successfully3 images loaded successfully4 images loaded successfully5 images loaded successfully6 images loaded successfully7 images loaded successfully8 images loaded successfully9 images l

shuffled
```



parkinsons final stage

File Edit View Insert Runtime Tools Help Last saved at 03:31

Comment Share

+ Code + Text

Connect Editing

```
print(type(dataset))
print(len(data))
print(len(data[1]))
print(data[0].shape)
print(len(data[1][:20]))
```

```
<class '__main__.DataSet'>
2
932
(932, 512, 512, 3)
20
```

```
import pickle
from google.colab import files

us_fil = open('park_data_enhanced.pickle', 'ab')
pickle.dump(data, us_fil)
us_fil.close()

files.download('park_data_enhanced.pickle')
```

```
x=len(data[0])
test_sample_size=int(0.01*x)
train_sample_size=x-test_sample_size

#splitting the data into training set and test set,with test_sample_size being the percentage of total dataset for test set
(tr_img,tr_lab),(te_img,te_lab)=(data[0][:train_sample_size],data[1][:train_sample_size]),(data[0][train_sample_size:],data[1][train_sample_size:])
```

```
print(tr_img.shape)
print(tr_lab.shape)
print(te_img.shape)
print(te_lab.shape)
print(tr_lab)
print(te_lab)
plt.imshow(tr_img[18],cmap='gray')
```

References

1. *Hoehn M, Yahr M. Parkinsonism: onset, progression, and mortality. Neurology. 1967;57(2):318 and 16 pages following.*
2. *Global, regional, and national burden of Parkinson's disease, 1990–2016: a systematic analysis for the Global Burden of Disease Study 2016 P939-953*
3. *Oertel, Wolfgang H. "Recent advances in treating Parkinson's disease." F1000Research vol. 6 260. 13 Mar. 2017, doi:10.12688/f1000research.10100.1*
4. *Pagán FL. Improving Outcomes Through Early Diagnosis of Parkinson's Disease. Am J Manag Care. 2012;18(September):176–82*
5. *Bharucha NE, Bharucha EP, Bharucha AE, Bhise AV, Schoenberg BS. Prevalence of Parkinson's disease in the Parsi community of Bombay, India. Arch Neurol. 1988;45:1321–3.*
6. *Surathi, Pratibha et al. "Research in Parkinson's disease in India: A review." Annals of Indian Academy of Neurology vol. 19,1 (2016): 9-20. doi:10.4103/0972-2327.167713*
7. *Seifert KD, Wiener JI. The impact of DaTscan on the diagnosis and management of movement disorders: A retrospective study. Am J Neurodegener Dis. 2013;2(1):29–34.*
8. *Blazejewska, Anna I et al. "Visualization of nigrosome 1 and its loss in PD: pathoanatomical correlation and in vivo 7 T MRI." Neurology vol. 81,6 (2013): 534-40. doi:10.1212/WNL.0b013e31829e6fd2*