# Capstone Project

# On

## CT Scan Image Classification

**Data Overview:**

This dataset contains 1252 CT scans that are positive for SARS-CoV-2 infection (COVID-19) and 1230 CT scans for patients non-infected by SARS-CoV-2, 2482 CT scans in total. These data have been collected from real patients in hospitals from Sao Paulo, Brazil. The aim of this dataset is to encourage the research and development of artificial intelligent methods which are able to identify if a person is infected by SARS-CoV-2 through the analysis of his/her CT scans.

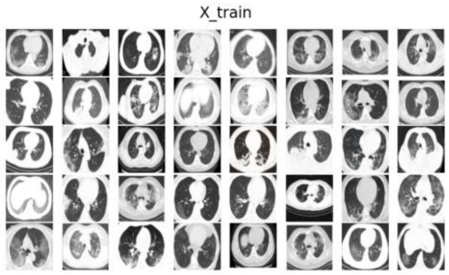
This Python 3 environment comes with many helpful analytics libraries installed. Google compute engine backend GPU.

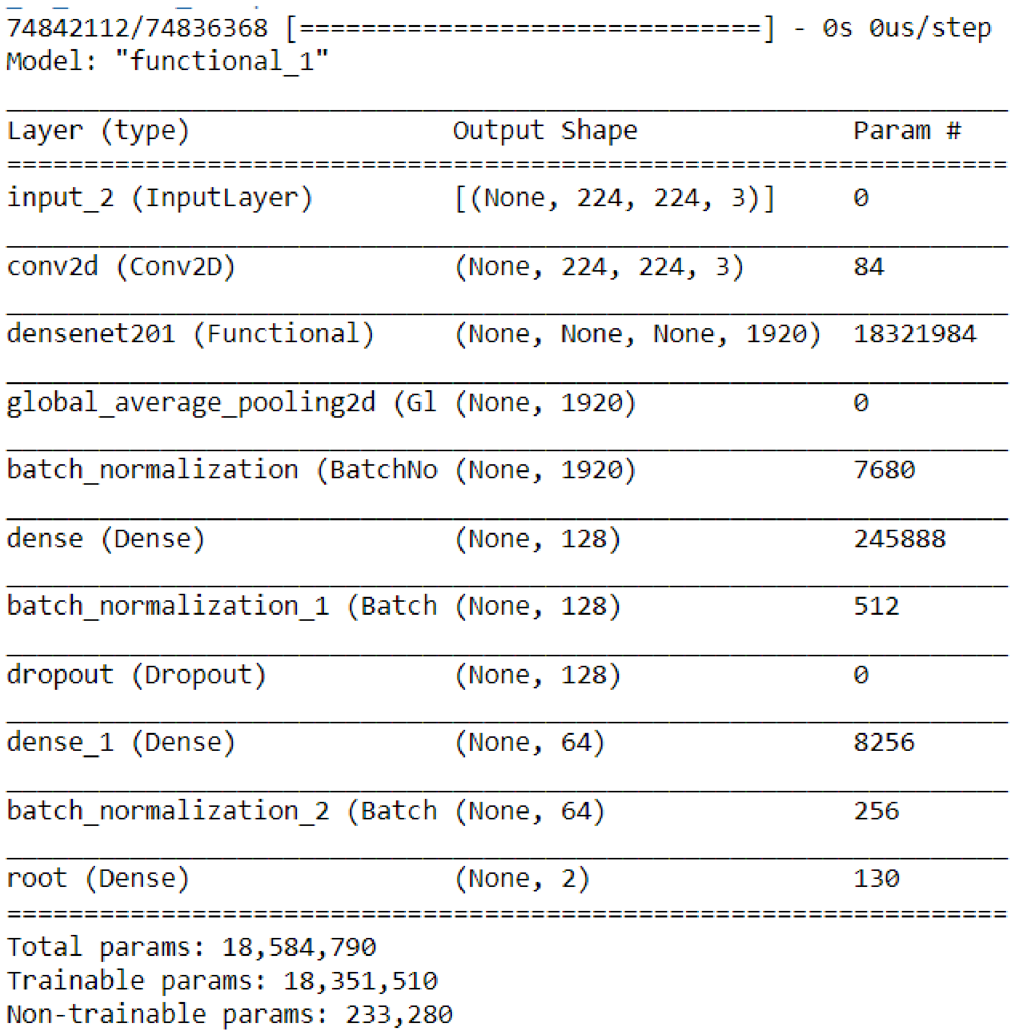
Here's several helpful packages to load :

```
import numpy as np # linear algebra

import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

import matplotlib.pyplot as plt

%matplotlib inline

import seaborn as sns

import cv2

import os

from tqdm import tqdm

from sklearn.metrics import confusion_matrix

from sklearn.model_selection import train_test_split

from keras.utils.np_utils import to_categorical

from keras.models import Model,Sequential, Input, load_model

from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPool2D, BatchNormalization, AveragePooling2D, GlobalAveragePooling2D

from keras.optimizers import Adam

from keras.preprocessing.image import ImageDataGenerator

from keras.callbacks import ModelCheckpoint, ReduceLROnPlateau
```

from keras.applications import DenseNet121

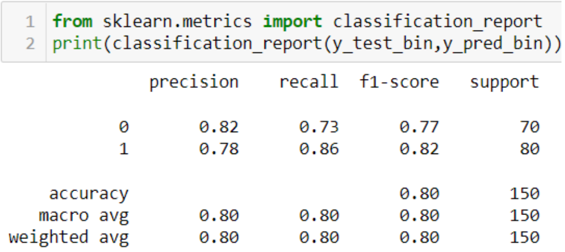
- Displaying images of Covid and Non-Covid
- Image resize and read function / Image augmentation
- Train the images
- Convert labels in to categorical data
- Splitting data in train and test
- 64*64 training data/images
- DenseNet201Model
- DenseNet121 Model
- Data Augmentation and fitting model
- Loss and accuracy : Loss – 0.20 and **accuracy – 0.92**
- Confusion matrix
- Activation function : Relu and softmax
- Plot ROC curve
- Resnet50
- Plotting accuracy and loss plot
- Classification report
- Finally prediction from Image





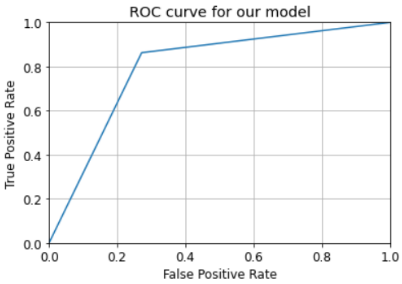
```
74842112/74836368 [==============================] - 0s 0us/step
Model: "functional_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
input_2 (InputLayer)         [(None, 224, 224, 3)]     0         
_________________________________________________________________
conv2d (Conv2D)              (None, 224, 224, 3)       84        
_________________________________________________________________
densenet201 (Functional)     (None, None, None, 1920)  18321984  
_________________________________________________________________
global_average_pooling2d (Gl (None, 1920)              0         
_________________________________________________________________
batch_normalization (BatchNo (None, 1920)              7680      
_________________________________________________________________
dense (Dense)                (None, 128)               245888    
_________________________________________________________________
batch_normalization_1 (Batch (None, 128)               512       
_________________________________________________________________
dropout (Dropout)            (None, 128)               0         
_________________________________________________________________
dense_1 (Dense)              (None, 64)                8256      
_________________________________________________________________
batch_normalization_2 (Batch (None, 64)                256       
_________________________________________________________________
root (Dense)                 (None, 2)                 130       
=================================================================
Total params: 18,584,790
Trainable params: 18,351,510
Non-trainable params: 233,280
```

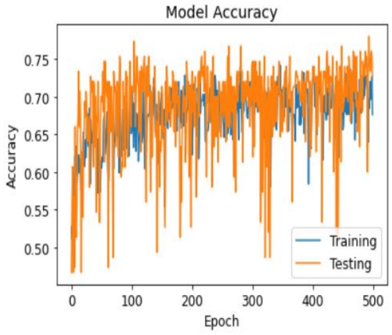
**Classification Report**

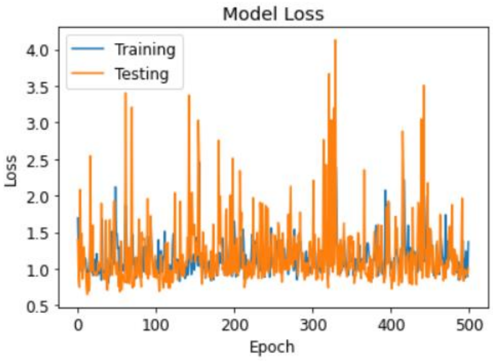


```
from sklearn.metrics import classification_report
print(classification_report(y_test_bin,y_pred_bin))
```

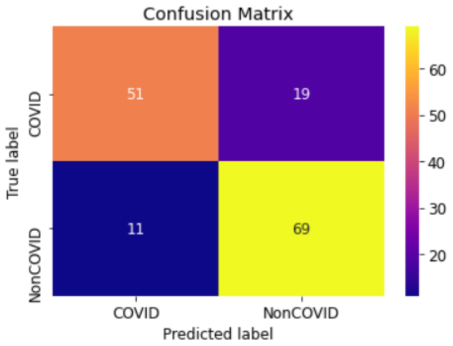
```
              precision    recall  f1-score   support

           0       0.82      0.73      0.77        70
           1       0.78      0.86      0.82        80

    accuracy                           0.80       150
   macro avg       0.80      0.80      0.80       150
weighted avg       0.80      0.80      0.80       150
```



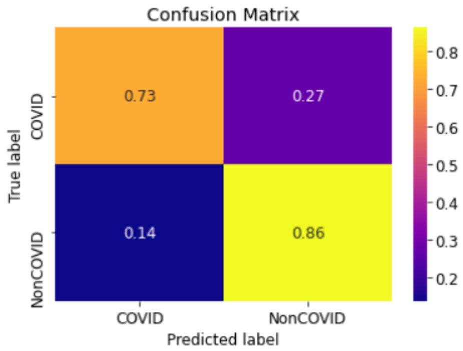




Confusion Matrix without Normalization



Confusion Matrix with Normalized Values



Conclusion: In this developed model, Resnet50, one of the CNN architectures, was used as the base. By removing 5 layers of the Resnet50 model, 10 new layers were added to the Resnet50. With this developed hybrid model, an accuracy rate of 92% was achieved. At the same time, results were acquired with Densenet201, Resnet50, Vgg16 and Densenet121 architectures. The highest accuracy rate was achieved with the hybrid model we improved.