Detection of COVID-19 Based on Chest X-rays Using Deep Learning

A model by Tarun gupta(20174027) and Harinder Kumar(20174007)

Here first we are importing important libraries

In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import cv2
import os
import keras
from keras.preprocessing.image import ImageDataGenerator
from keras.optimizers import Adam
from tensorflow.keras.layers import Dense, BatchNormalization
import tensorflow as tf
print(keras.__version__)
print(tf.__version__)
```

```
2.5.0
2.5.0
```

Now the preprocessing part in which we are using image.ImageDataGenerator from keras.preprocessing rescaling the image and setting diffrent parameters like rotation, width shift,heigth shift and also fixing our validation split of 20 % of dataset

In [2]:

```
idg=tf.keras.preprocessing.image.ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    zoom_range=0.2,
    shear_range=0.2,
    horizontal_flip=True,
    vertical_flip=False,
    validation_split=0.2
)
```

Training data is represented as train_gen and validation data is represented as val_gen and setting dimensions of each image as (128,128)

class_mode is set as binary because the output is either 0(infected) or 1(normal)

shuffling the data to remove any pattern in data

In [3]:

```
train_gen = idg.flow_from_directory("data1",
                                                   target_size=(128,128),

                                                   subset='training',
                                                   class_mode='binary',
                                                   batch_size=32,
                                                   shuffle=True,
                                                   seed=1
                                                )
```

```
Found 18126 images belonging to 2 classes.
```

In [4]:

```
val_gen = idg.flow_from_directory("data1",
                                                   target_size=(128,128),

                                                   subset='validation',
                                                   class_mode='binary',
                                                   batch_size=32,
                                                   shuffle=True,
                                                   seed=1
                                                )
```

```
Found 4531 images belonging to 2 classes.
```

Output is binary and its a problem of classification

0 represents covid infected X-ray whereas 1 represent Normal X-ray

In [5]:

```
train_gen.class_indices
```

Out[5]:

```
{'covid': 0, 'normal': 1}
```

Here we are importing some import layers from tensorflow keras for model preparation and also importing Resnet50 pretrained model

In [10]:

```
from tensorflow.keras.applications.resnet50 import ResNet50
from tensorflow.keras.layers import GlobalAveragePooling2D
from tensorflow.keras.models import Sequential
from tensorflow.keras.models import Model
from tensorflow.keras import optimizers
from tensorflow.keras.callbacks import ReduceLROnPlateau, EarlyStopping
input_shape = (128,128, 3)
googleNet_model =ResNet50(include_top=False, weights='imagenet', input_shape=input_shape)
googleNet_model.trainable = True

model = Sequential()
model.add(googleNet_model)
model.add(GlobalAveragePooling2D())

model.add(Dense(units=2, activation='softmax'))
model.compile(loss='sparse_categorical_crossentropy',
              optimizer=optimizers.Adam(lr=1e-5, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.0, amsgrad=False),
              metrics=['accuracy'])
model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
resnet50 (Functional)        (None, 4, 4, 2048)        23587712
_________________________________________________________________
global_average_pooling2d_2 ( (None, 2048)              0
_________________________________________________________________
module_wrapper_1 (ModuleWrap (None, 2)                 4098
=================================================================
Total params: 23,591,810
Trainable params: 23,538,690
Non-trainable params: 53,120
_________________________________________________________________
```

Here we are declaring checkpoint which save the best accuracy model as "2resnet50.h5"

we are training for over 20 epochs and a batch size of 32

In [8]:

```
checkpoint=tf.keras.callbacks.ModelCheckpoint(
   '2resnet50.h5',
    save_best_only=True)





EPOCHS = 20
BATCH_SIZE =32
history = model.fit(train_gen, batch_size = BATCH_SIZE, epochs = EPOCHS, validation_data
= val_gen, verbose = 1,callbacks=checkpoint)
```

```
Epoch 1/20
567/567 [==============================] - 213s 353ms/step - loss: 0.3754 - accuracy: 0.8
263 - val_loss: 1.2766 - val_accuracy: 0.4423
```

```
C:\Users\user\anaconda3\lib\site-packages\tensorflow\python\keras\utils\generic_utils.py:
494: CustomMaskWarning: Custom mask layers require a config and must override get_config.
When loading, the custom mask layer must be passed to the custom_objects argument.
  warnings.warn('Custom mask layers require a config and must override '
```

```
Epoch 2/20
567/567 [==============================] - 140s 247ms/step - loss: 0.2190 - accuracy: 0.9
094 - val_loss: 0.4745 - val_accuracy: 0.8029
Epoch 3/20
567/567 [==============================] - 148s 261ms/step - loss: 0.1716 - accuracy: 0.9
313 - val_loss: 0.2662 - val_accuracy: 0.8947
Epoch 4/20
567/567 [==============================] - 143s 252ms/step - loss: 0.1418 - accuracy: 0.9
421 - val_loss: 0.2712 - val_accuracy: 0.8949
Epoch 5/20
567/567 [==============================] - 150s 265ms/step - loss: 0.1226 - accuracy: 0.9
496 - val_loss: 0.1198 - val_accuracy: 0.9574
Epoch 6/20
567/567 [==============================] - 140s 247ms/step - loss: 0.1106 - accuracy: 0.9
577 - val_loss: 0.1789 - val_accuracy: 0.9369
Epoch 7/20
567/567 [==============================] - 138s 243ms/step - loss: 0.0998 - accuracy: 0.9
617 - val_loss: 0.1121 - val_accuracy: 0.9623
Epoch 8/20
567/567 [==============================] - 134s 236ms/step - loss: 0.0902 - accuracy: 0.9
659 - val_loss: 0.1128 - val_accuracy: 0.9585
Epoch 9/20
567/567 [==============================] - 133s 234ms/step - loss: 0.0783 - accuracy: 0.9
702 - val_loss: 0.0871 - val_accuracy: 0.9669
Epoch 10/20
567/567 [==============================] - 132s 232ms/step - loss: 0.0807 - accuracy: 0.9
684 - val_loss: 0.1468 - val_accuracy: 0.9477
Epoch 11/20
567/567 [==============================] - 132s 233ms/step - loss: 0.0701 - accuracy: 0.9
737 - val_loss: 0.0864 - val_accuracy: 0.9715
Epoch 12/20
Epoch 12/20
567/567 [==============================] - 132s 233ms/step - loss: 0.0668 - accuracy: 0.9
761 - val_loss: 0.0993 - val_accuracy: 0.9653
Epoch 13/20
567/567 [==============================] - 136s 239ms/step - loss: 0.0627 - accuracy: 0.9
759 - val_loss: 0.0907 - val_accuracy: 0.9704
Epoch 14/20
567/567 [==============================] - 132s 232ms/step - loss: 0.0595 - accuracy: 0.9
767 - val_loss: 0.0932 - val_accuracy: 0.9709
Epoch 15/20
567/567 [==============================] - 132s 233ms/step - loss: 0.0564 - accuracy: 0.9
790 - val_loss: 0.0961 - val_accuracy: 0.9676
Epoch 16/20
567/567 [==============================] - 134s 236ms/step - loss: 0.0565 - accuracy: 0.9
784 - val_loss: 0.0897 - val_accuracy: 0.9784
Epoch 17/20
567/567 [==============================] - 133s 234ms/step - loss: 0.0471 - accuracy: 0.9
813 - val_loss: 0.0960 - val_accuracy: 0.9715
Epoch 18/20
567/567 [==============================] - 133s 234ms/step - loss: 0.0524 - accuracy: 0.9
799 - val_loss: 0.1089 - val_accuracy: 0.9673
Epoch 19/20
567/567 [==============================] - 132s 233ms/step - loss: 0.0433 - accuracy: 0.9
844 - val_loss: 0.0707 - val_accuracy: 0.9759
Epoch 20/20
567/567 [==============================] - 132s 233ms/step - loss: 0.0397 - accuracy: 0.9
846 - val_loss: 0.0966 - val_accuracy: 0.9724
```

we are now loading our saved model

In [11]:

```
model = keras.models.load_model('2resnet50.h5')
```

so we have done all our work and made a model now we are testing it on 14035 chest Xrays in which 3834 are infected one and the rest 10192 are noraml chest Xrays

In [14]:

```
test_datagen = image.ImageDataGenerator(rescale=1./255)
test_gen = test_datagen.flow_from_directory(
        'covid',
        target_size=(128,128),
        batch_size=1,
        class_mode='binary',
        shuffle=False)
pred= model.predict(test_gen, verbose=1)
predicted_class_indices=np.argmax(pred,axis=1)
count0=0
count1=0
for val in range(len(pred)):
    if predicted_class_indices[val]==0:
        count0=count0+1
    else:
        count1=count1+1
print("all_positive")
print("No.of true positive is {} ".format(count0))
print("No.of false positive is {}".format(count1))
print(" ")

############################################


test_gen = test_datagen.flow_from_directory(
        'norm',
        target_size=(128,128),
        batch_size=1,
        class_mode='binary',
        shuffle=False)
pred= model.predict(test_gen, verbose=1)
predicted_class_indices=np.argmax(pred,axis=1)
count0=0
count1=0
for val in range(len(pred)):
    if predicted_class_indices[val]==0:
        count0=count0+1
    else:
        count1=count1+1
print("all_negative")
print("No. of false negative are {}".format(count0))
print("No of true negative are {}".format(count1))
print(" ")
```

```
Found 3834 images belonging to 1 classes.
3834/3834 [==============================] - 70s 13ms/step
all_positive
No.of true positive is 3687
No.of false positive is 147

Found 10192 images belonging to 1 classes.
10192/10192 [==============================] - 129s 13ms/step
all_negative
No. of false negative are 333
No of true negative are 9859
```

conclusion :-we got an accuracy of 96.51 percent which is a pretty good accuracy so if we check an xray there is 96.5 percent chance that the prediction is correct