**Pneumonia Detection using Deep Learning**

**CODE:**

from keras.models import Model

from keras.layers import Flatten,Dense

from keras.applications.vgg16 import VGG16 #Import all the necessary modules

import matplotlib.pyplot as plot

from glob import glob

IMAGESHAPE = [224, 224, 3] #Provide image size as 224 x 224 this is a fixed-size for VGG16 architecture

vgg\_model = VGG16(input\_shape=IMAGESHAPE, weights='imagenet', include\_top=False)

#3 signifies that we are working with RGB type of images.

training\_data = 'chest\_xray/train'

testing\_data = 'chest\_xray/test' #Give our training and testing path

for each\_layer in vgg\_model.layers:

each\_layer.trainable = False #Set the trainable as False, So that all the layers would not be trained.

classes = glob('chest\_xray/train/\*') #Finding how many classes present in our train dataset.

flatten\_layer = Flatten()(vgg\_model.output)

prediction = Dense(len(classes), activation='softmax')(flatten\_layer)

final\_model = Model(inputs=vgg\_model.input, outputs=prediction) #Combine the VGG output and prediction , this all together will create a model.

final\_model.summary() #Displaying the summary

final\_model.compile( #Compiling our model using adam optimizer and optimization metric as accuracy.

loss='categorical\_crossentropy',

optimizer='adam',

metrics=['accuracy']

)

from keras.preprocessing.image import ImageDataGenerator

train\_datagen = ImageDataGenerator(rescale = 1./255, #importing our dataset to keras using ImageDataGenerator in keras.

shear\_range = 0.2,

zoom\_range = 0.2,

horizontal\_flip = True)

testing\_datagen = ImageDataGenerator(rescale =1. / 255)

training\_set = train\_datagen.flow\_from\_directory('chest\_xray/train', #inserting the images.

target\_size = (224, 224),

batch\_size = 4,

class\_mode = 'categorical')

test\_set = testing\_datagen.flow\_from\_directory('chest\_xray/test',

target\_size = (224, 224),

batch\_size = 4,

class\_mode = 'categorical')

fitted\_model = final\_model.fit\_generator( #Fitting the model.

training\_set,

validation\_data=test\_set,

epochs=5,

steps\_per\_epoch=len(training\_set),

validation\_steps=len(test\_set)

)

plot.plot(fitted\_model.history['loss'], label='training loss') #Plotting the accuracies

plot.plot(fitted\_model.history['val\_loss'], label='validation loss')

plot.legend()

plot.show()

plot.savefig('LossVal\_loss')

plot.plot(fitted\_model.history['acc'], label='training accuracy')

plot.plot(fitted\_model.history['val\_acc'], label='validation accuracy')

plot.legend()

plot.show()

plot.savefig('AccVal\_acc')

final\_model.save('our\_model.h5') #Saving the model file.

**TESTING:**

from keras\_preprocessing import image

from keras.models import load\_model

from keras.applications.vgg16 import preprocess\_input

import numpy as np

model=load\_model('our\_model.h5') #Loading our model

img=image.load\_img('D:/Semester - 6/PneumoniaGFG/chest\_xray/val/PNEUMONIA/person1954\_bacteria\_4886.jpeg',target\_size=(224,224))

imagee=image.img\_to\_array(img) #Converting the X-Ray into pixels

imagee=np.expand\_dims(imagee, axis=0)

img\_data=preprocess\_input(imagee)

prediction=model.predict(img\_data)

if prediction[0][0]>prediction[0][1]: #Printing the prediction of model.

print('Person is safe.')

else:

print('Person is affected with Pneumonia.')

print(f'Predictions: {prediction}')