Documentation on Pneumonia Prediction Using X-RAY Images

Name : Anand mohan
email: 18bcs6218@gmail.com

Internship Title : RSIP Career Basic AI 048
Project ID:SPS_PRO_182
Project Title : Pneumonia Prediction Using X-RAY Images
Duration : 19 Days

Category: Artificial Intelligence
Internship at smartinternz.com@2020
Skills Required: Python,Python Web Frame Works,CNN

Table of Contents

Documentation on Pneumonia Prediction Using X-RAY Images

1. Introduction

1.1 OverView

We will be able to develop a web application using Flask. By the end of the project we'll learn best practices of combining Machine Learning Code , Machine Learning Model , HTML and CSS to buld a web application .

1.2 Purpose

In general, a person suffering from Pneumonia goes to the hospital for a X-Ray image and waits for the doctor to check the image manually himself and then decides whether the person has Pneumonia or not . The aim of this project is to help the doctors to predict the pneumonia disease more accurately using a deep learning model.

2. Literature Survey

2.1 Existing Problem

In general, a patient suffering from Pneumonia goes to the hospital to take an X-ray image waits for the doctor and then the doctor will check the X-ray then he decides whether the person has pneumonia or not. The results are not only concluded based on just seeing the X-ray images but furthermore, tests were conducted on the patient to verify the results of the doctor. The process is time-consuming and if the patient has severe pneumonia or not he has to wait several days to get the test results. But in recent developments of the artificial intelligence and the computational powers of the computers have increased it helps in predicting pneumonia by just passing the X-ray image as an input to our model.

2.2 Proposed Solution

The main objective of this project is to help the doctors to predict the pneumonia disease more accurately using a deep learning model. The objective is not only to help the doctors but also to the patients to verify whether they have pneumonia or not. By using this model we can precisely predict pneumonia. A convolutional neural network model is built from scratch to extract features from a given chest X-ray image and classify it to determine if a person is infected with pneumonia. a web is built where the user can upload the x - ray image and the result is shown on the UI .

Documentation on Pneumonia Prediction Using X-RAY Images

3. Theoritical Analysis

3.1 Language/Software

This project was mostly developed in python version 3.6 for building the AI model and for deplying the model on the web and the UI for the web was developed using HTML5 and CSS.

3.1.1 IDE

- for AI Model development jupyter notebook was used and the model was saves with a .h5 extention.
- for Developing the web application i used Visual Studio Code in which the AI model, HTML and CSS code was integrated to form the web application. Any other IDE's like pycham , spyder can be used it you are not comfortable with Visual Studio Code.

3.2 FlowChart

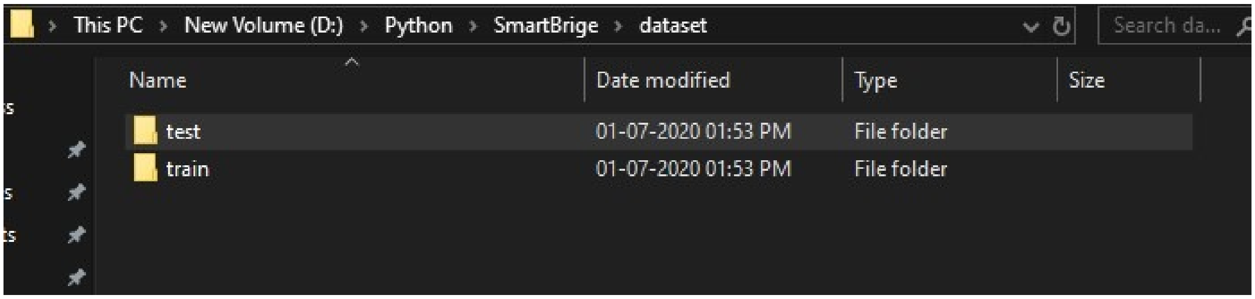
Steps which are taken while developing this project

1. Data Collection
2. Data Preprocessing
3. Model Building
4. Web Application Building

4.Experimental Investigation

1.Data Collection

The Dataset is collected from kaggle.com and divided into two folders test and train

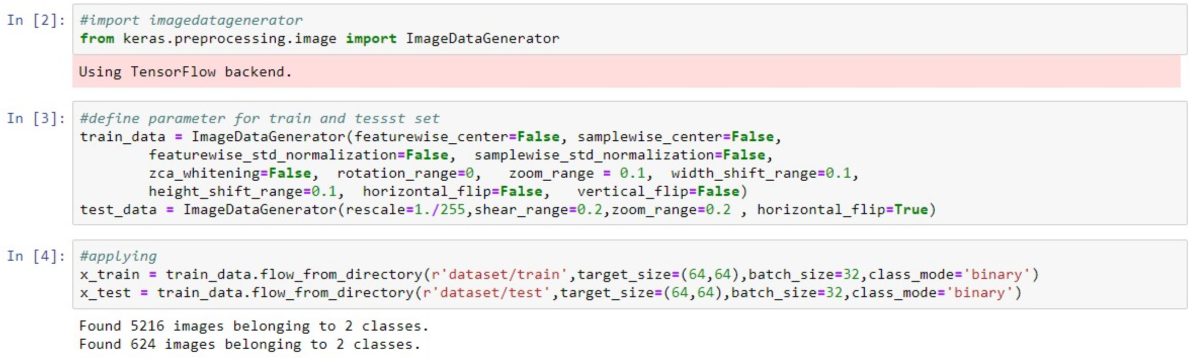


This PC › New Volume (D:) › Python › SmartBrige › dataset

Name	Date modified	Type	Size
test	01-07-2020 01:53 PM	File folder	
train	01-07-2020 01:53 PM	File folder	

2.Data Preprocessing

The Dataset that we got was already clean due to which we didn't have. Just the ImageDataGenerator was used to increase the number of images present in our test and train folder.

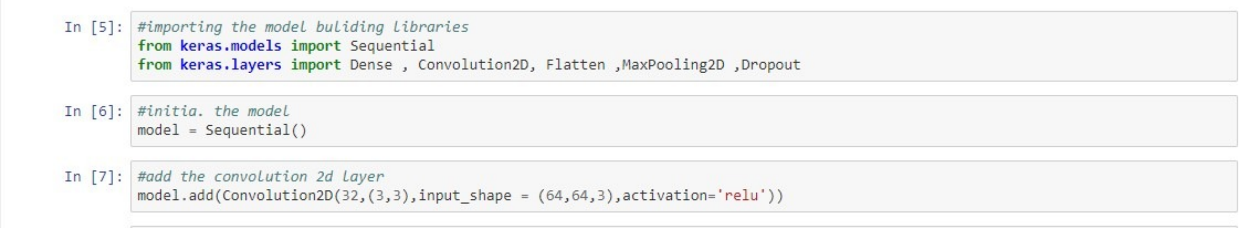


```
In [2]: #import imagedatagenerator
        from keras.preprocessing.image import ImageDataGenerator

        Using TensorFlow backend.

In [3]: #define parameter for train and tessst set
        train_data = ImageDataGenerator(featurewise_center=False, samplewise_center=False,
                featurewise_std_normalization=False,  samplewise_std_normalization=False,
                zca_whitening=False,  rotation_range=0,   zoom_range = 0.1,  width_shift_range=0.1,
                height_shift_range=0.1,  horizontal_flip=False,   vertical_flip=False)
        test_data = ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2 , horizontal_flip=True)

In [4]: #applying
        x_train = train_data.flow_from_directory(r'dataset/train',target_size=(64,64),batch_size=32,class_mode='binary')
        x_test = train_data.flow_from_directory(r'dataset/test',target_size=(64,64),batch_size=32,class_mode='binary')

        Found 5216 images belonging to 2 classes.
        Found 624 images belonging to 2 classes.
```

3. Model Building

After Preprocessinng we start with the model building

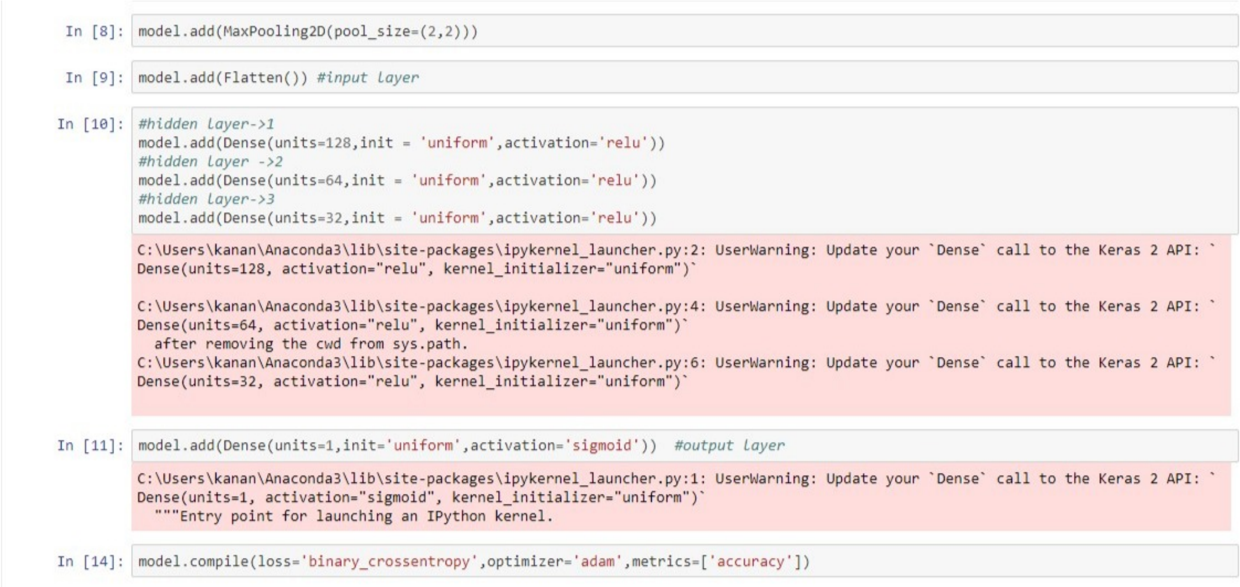
1. adding all required libraries
2. Sequentialise the model
3. dd the convoution2D layer with all parameters the input shape was taken as 64,64,3 and the activation function as relu but other functions could also been used



```
In [5]: #importing the model buliding Libraries
        from keras.models import Sequential
        from keras.layers import Dense , Convolution2D, Flatten ,MaxPooling2D ,Dropout

In [6]: #initia. the model
        model = Sequential()

In [7]: #add the convolution 2d Layer
        model.add(Convolution2D(32,(3,3),input_shape = (64,64,3),activation='relu'))
```

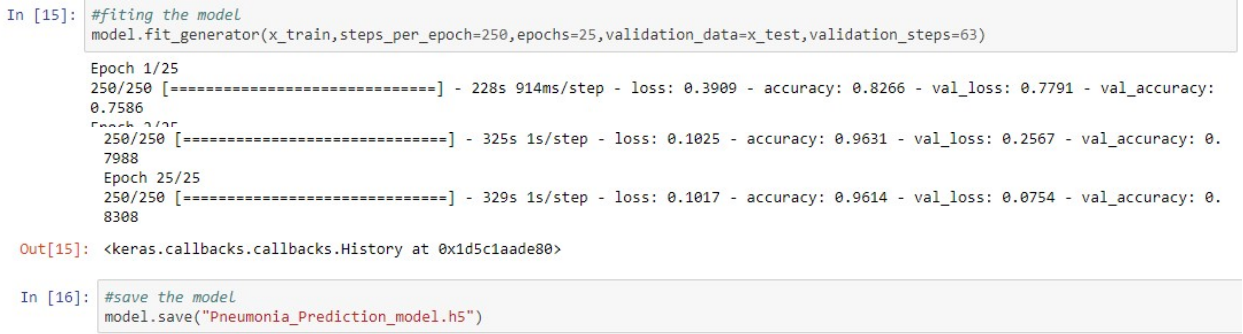
4. adding theMaxpooling layer then flatten layer
5. after this we will add the hidden layes of the neural network. There are 3 Dense layers which have half output node to that of there input

6. then we add the output later which has 1 output node and after output layer we complie the model for loss , accuracy and also optimise



```
In [8]:  model.add(MaxPooling2D(pool_size=(2,2)))

In [9]:  model.add(Flatten()) #input layer

In [10]: #hidden layer->1
         model.add(Dense(units=128,init = 'uniform',activation='relu'))
         #hidden layer ->2
         model.add(Dense(units=64,init = 'uniform',activation='relu'))
         #hidden layer->3
         model.add(Dense(units=32,init = 'uniform',activation='relu'))

         C:\Users\kanan\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: UserWarning: Update your `Dense` call to the Keras 2 API: `
         Dense(units=128, activation="relu", kernel_initializer="uniform")`

         C:\Users\kanan\Anaconda3\lib\site-packages\ipykernel_launcher.py:4: UserWarning: Update your `Dense` call to the Keras 2 API: `
         Dense(units=64, activation="relu", kernel_initializer="uniform")`
           after removing the cwd from sys.path.
         C:\Users\kanan\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: UserWarning: Update your `Dense` call to the Keras 2 API: `
         Dense(units=32, activation="relu", kernel_initializer="uniform")`

In [11]: model.add(Dense(units=1,init='uniform',activation='sigmoid'))  #output layer

         C:\Users\kanan\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: UserWarning: Update your `Dense` call to the Keras 2 API: `
         Dense(units=1, activation="sigmoid", kernel_initializer="uniform")`
           """Entry point for launching an IPython kernel.

In [14]: model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
```

7. then we train the model and after training we save the model.



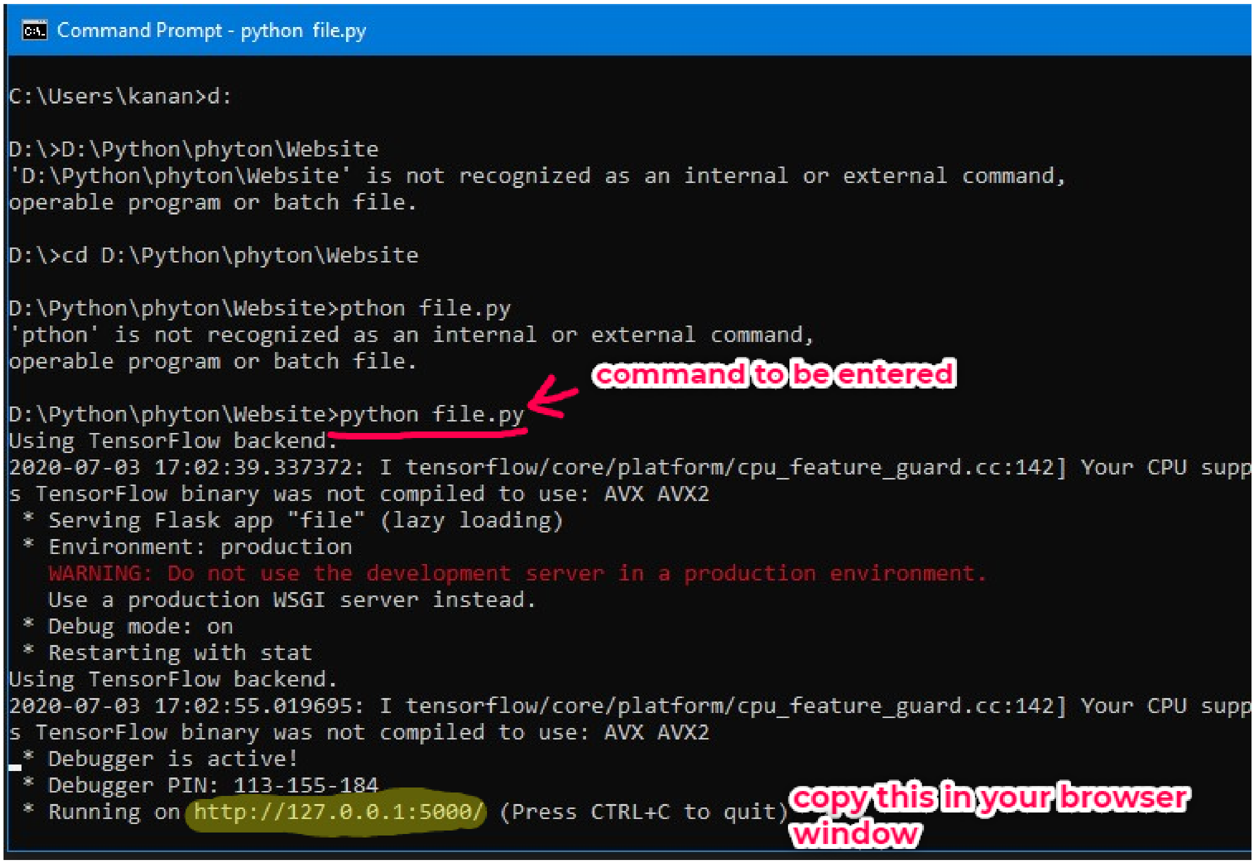
```
In [15]: #fiting the model
         model.fit_generator(x_train,steps_per_epoch=250,epochs=25,validation_data=x_test,validation_steps=63)

         Epoch 1/25
         250/250 [==============================] - 228s 914ms/step - loss: 0.3909 - accuracy: 0.8266 - val_loss: 0.7791 - val_accuracy:
         0.7586
         Epoch 2/25
           250/250 [==============================] - 325s 1s/step - loss: 0.1025 - accuracy: 0.9631 - val_loss: 0.2567 - val_accuracy: 0.
           7988
           Epoch 25/25
           250/250 [==============================] - 329s 1s/step - loss: 0.1017 - accuracy: 0.9614 - val_loss: 0.0754 - val_accuracy: 0.
           8308

Out[15]:   <keras.callbacks.callbacks.History at 0x1d5c1aade80>

In [16]:   #save the model
           model.save("Pneumonia_Prediction_model.h5")
```

the model achived an accuracy of 96%

8. then we start to build our HTML page the design of the page depends on how you want it it does not follow any kind of rules. So i developed a very basic page which just has a few text , an image chooser and a button which on clicked used the AI model which was saved in last step to give an output and that is displayed . The HTML codee is been uploaded on my github under a folder named as template.
9. After the UI is made we start the pyhton code using flask framework in this we import out model and provide it with the picture uploaded by the user then the model works on ths picture and returns a render template with the html and text as areguments. the images which are given by the user are saved in a folder named uploads and the python

file is named as file.py

10. After developing the model and web app now we need to run the application go to anaconda prompt or command prompt and to the location where the .py file is located and type the command "python file.py" and it will start the server.



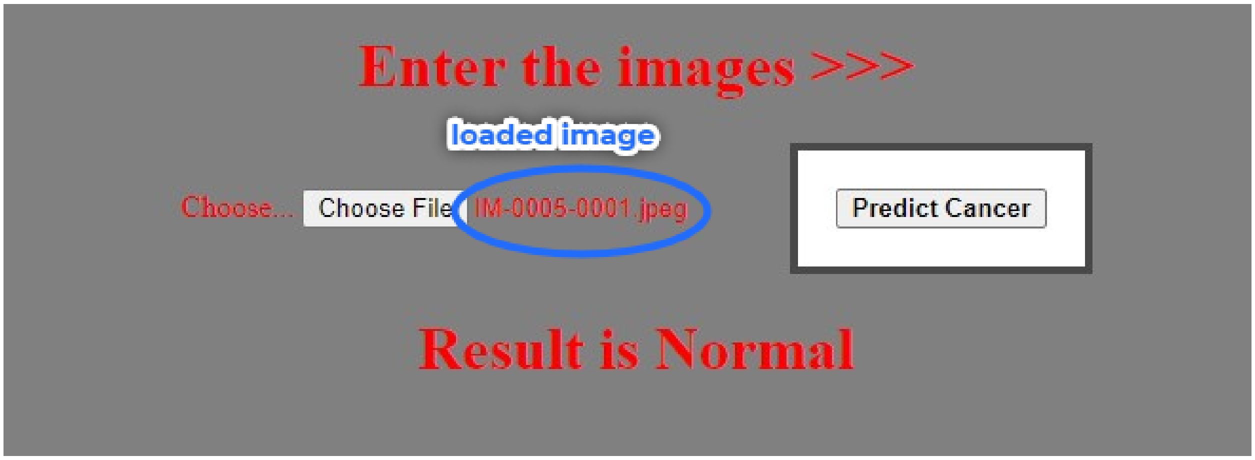
Command Prompt - python file.py

```
C:\Users\kanan>d:

D:\>D:\Python\phyton\Website
'D:\Python\phyton\Website' is not recognized as an internal or external command,
operable program or batch file.

D:\>cd D:\Python\phyton\Website

D:\Python\phyton\Website>pthon file.py
'pthon' is not recognized as an internal or external command,
operable program or batch file.

D:\Python\phyton\Website>python file.py
Using TensorFlow backend.
2020-07-03 17:02:39.337372: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supp
s TensorFlow binary was not compiled to use: AVX AVX2
 * Serving Flask app "file" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
Using TensorFlow backend.
2020-07-03 17:02:55.019695: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supp
s TensorFlow binary was not compiled to use: AVX AVX2
 * Debugger is active!
 * Debugger PIN: 113-155-184
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

command to be entered

copy this in your browser window

11. now the web app is working load an image and see the output



Enter the images >>>

IM-0005-0001.jpeg

Result is Normal

5. Advantages

The advantage of using Deep learning over learning techniquise like machine learning are :

- in this project we have lack of knowledge about all the features present in the images which can be used to identify pneumonia. in this case Deep learning is more useful as we have to worry less about the feature enginnering that is taken care .
- we are using images as inputs as Deep Learning really shines when it comes to complex problems such as image classification, natural language processing, and speech recognition.
- no need for label data deep learning works well with unlabeled data.
- the results or the model which is developed is of high quality

6. Disadvantages

The disadvantage of using Deep learning over learning techniquise like machine learning are :

- it requires a lot of data.
- a deep learning model is developed for a specific task it is not flexible to be used for some other task.
- Neural Networks at the Core of Deep Learning are Black Boxes. It's impossible to look inside of it to see how it works. Just like in a human brain, the reasoning of a neural network is embedded in the behavior of thousands of simulated neurons, arranged into dozens or even hundreds of intricately interconnected layers.

7. Conclusion

By doing the above procedure and all we have sucessfully created a Deep learning CNN model and deployed it using Flask framework.

8. Future Scope

A new model with higher accuracy , detecting more than one diseases or developing a new UI for the web are some of the future scope of this project.