

Predicting Covid-19 using CNN on lungs X-ray images:

Anubhav Nanda

Member of Robotics Society

Veer Surendra Sai University of Technology, Burla

Abstract :

In current scenario of the world under the Covid-19 pandemic we're all in an alarming situation. We've very limited resources to deal with the pandemic. With the number of cases crossing millions a plenty of resources are required for the testing and treatment of the disease. For testing millions of samples it requires millions of testing kits. Also the time consumed during the testing process is very crucial. Hence it is very crucial to develop a negligibly time consuming and accurate testing mehod which also requires very less resources. For this purpose we developed a Convolutional Neural Network with an accuracy of 98.15% to predict whether someone is infected by coronavirus from the X-ray of their lungs.

Introduction :-

In Neural networks CNN is the most commonly used category used for image classification, image recognition etc. Basically in CNN image classifications takes input and process it then further classify it under certain category as provided in the dataset. Computers sees an input image as array of pixels and it depends on the image resolution. Based on the image resolution, it will see $h \times w \times d$ (h = Height, w = Width, d = Dimension). Eg., An image of $6 \times 6 \times 3$ array of matrix of RGB (3 refers to RGB values) and an image of $4 \times 4 \times 1$ array of matrix of grayscale image.

Technically, deep learning CNN models to train and test, each input image will pass it through a series of convolution layers with filters (Kernels), Pooling, fully connected layers (FC) and apply Softmax function to classify an object with probabilistic values between 0 and 1. The below figure is a complete flow of CNN to process an input image and classifies the objects based on values

Objective:

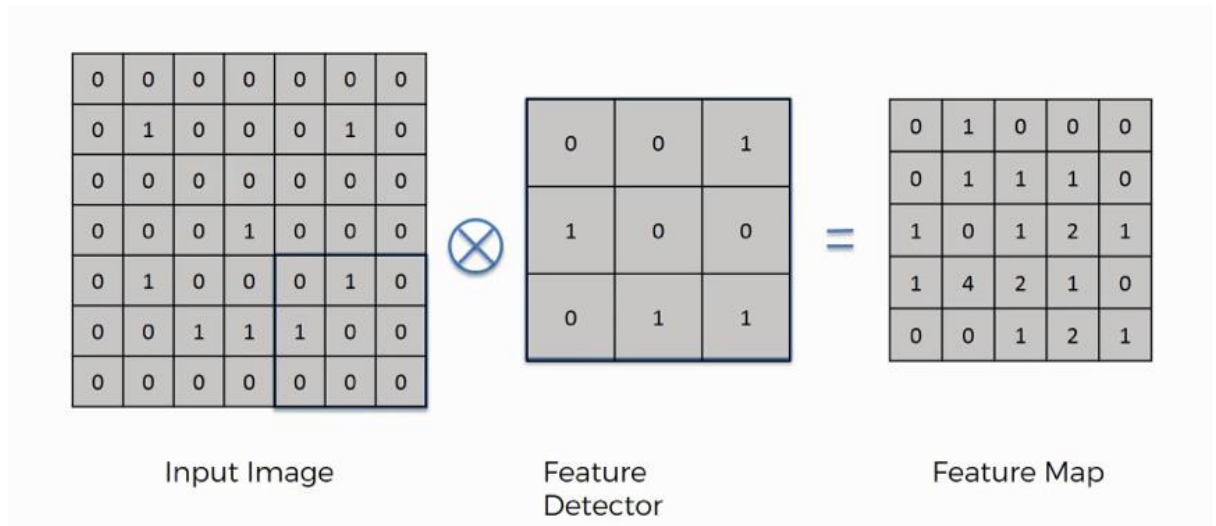
To develop a Neural Network with highest possible efficiency for Covid-19 detection in humans.

Software Required :

1. Spyder IDE.
2. Dataset of images of 250+ covid-19 and normal lungs X-ray each. Which will be further divided into 210 images for train and rest for test set.

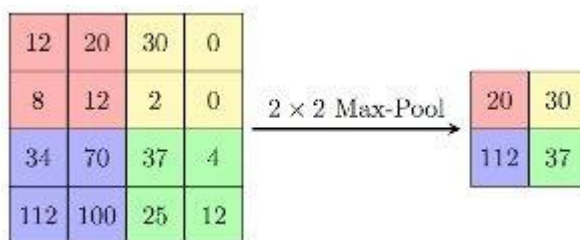
Steps to create a Neural Network :

- 1) Import all the necessary packages.I've used packages from keras.
- 2) Initialise the CNN.
- 3) Convolution : The convolution is the multiplication of a filter or kernel with the input data i.e. the image matrix extracted here.



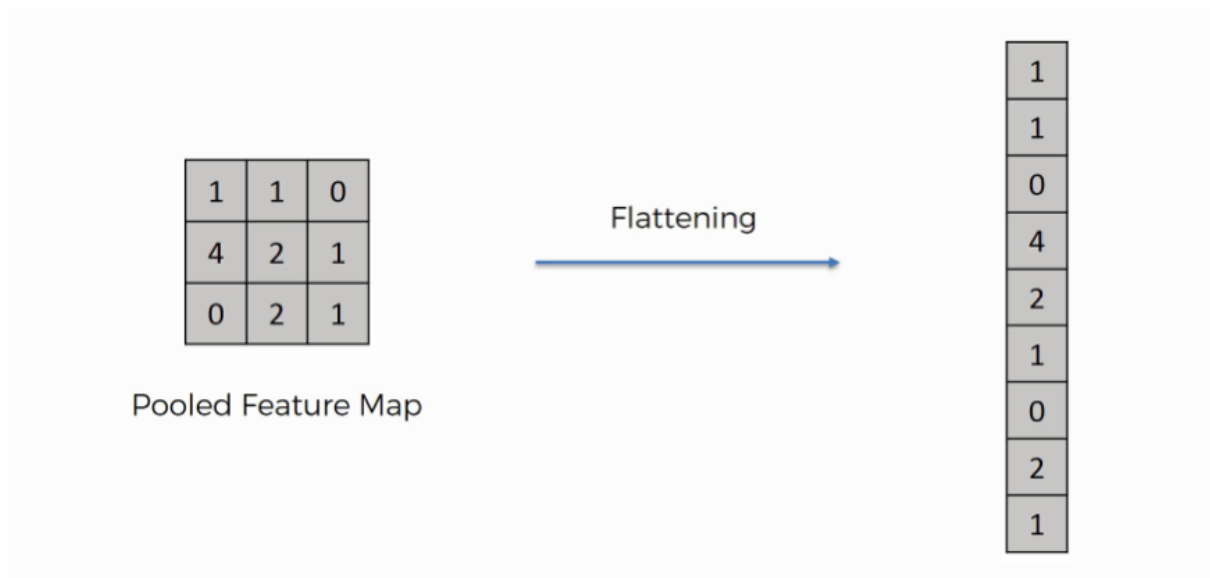
- 4) Pooling : Pooling is required to down sample the detection of features in feature maps.In our model we've applied MaxPooling.

*We've made two convolutional layers for better accuracy.

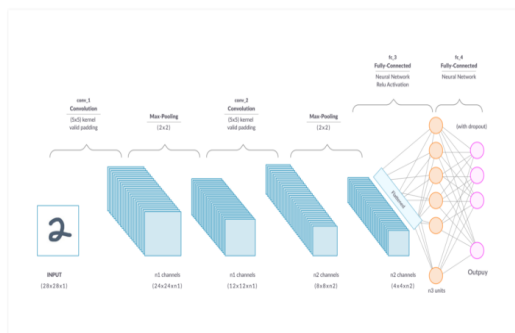


- 5) Flattening : Flattening transforms the two dimensional Pooled matrix into a single dimension array which our model can easily

understand.



- 6) Full connection : Full connection is the process of feeding the flattened matrix into our Neural Network. Here the input is number of neurons at input and the output is one neuron since the prediction will be a binary value (Yes/No question whether covid affected or not). In the full connection layer we use further data augmentation like adding a Dropout Layer and a Normalization layer.



- 7) Compile the CNN.
- 8) Fitting the CNN to the images: This code was taken from Keras image preprocessing section where the training set is fed to the CNN. Now the CNN make new predictions on the test set and we get the validation accuracy on the basis of those prediction. Here the sample images are flipped, zoomed and etc. image processing are performed to randomize the image and train our model better.

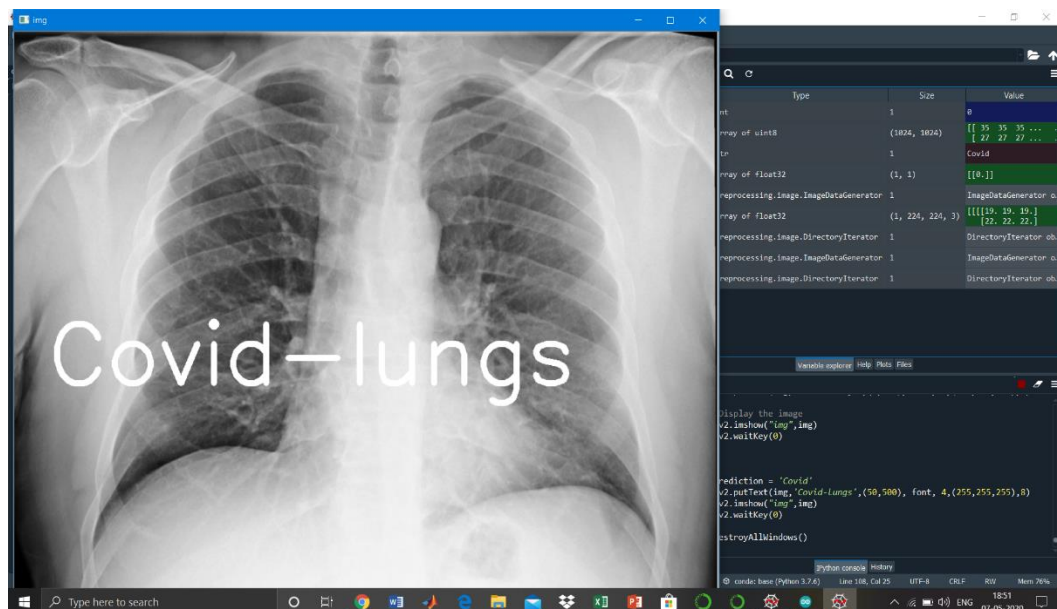
- 9) Making new predictions : Now we'll make new prediction on our own images that were neither in test or training set. This is done by converting the image read by keras image preprocessing into a numpy array and using the predict function.

Libraries to download :

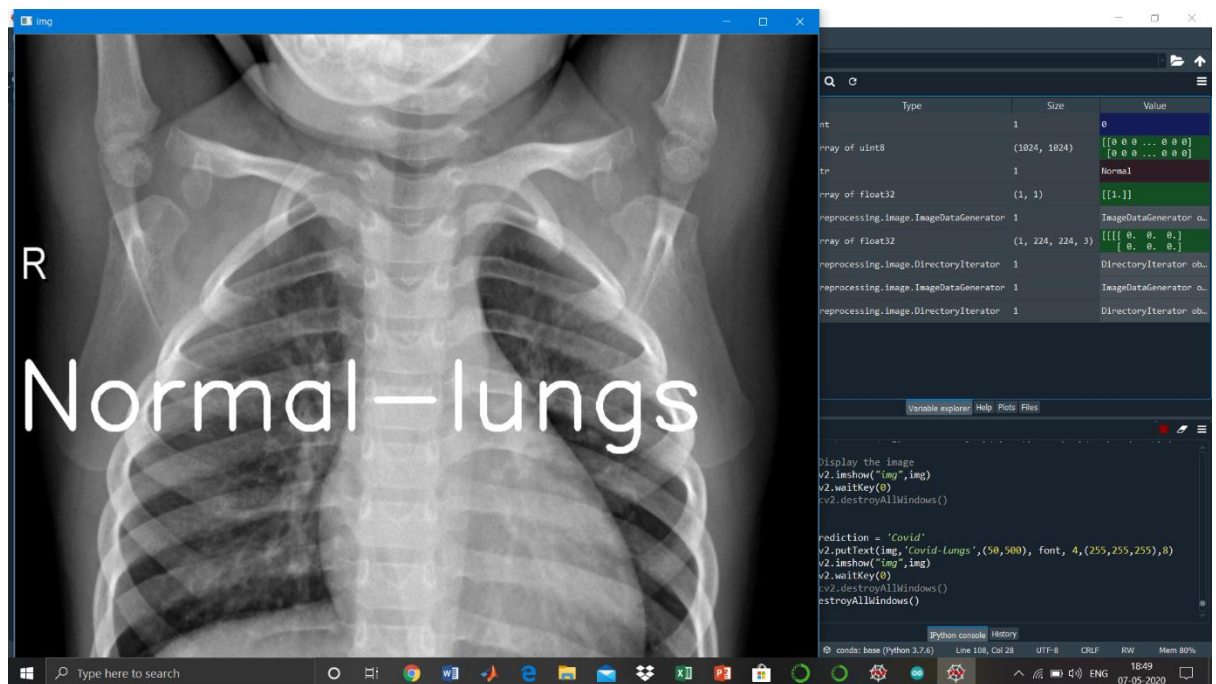
- 1.Tensorflow-GPU
- 2.OpenCV
- 3.Numpy

Output :

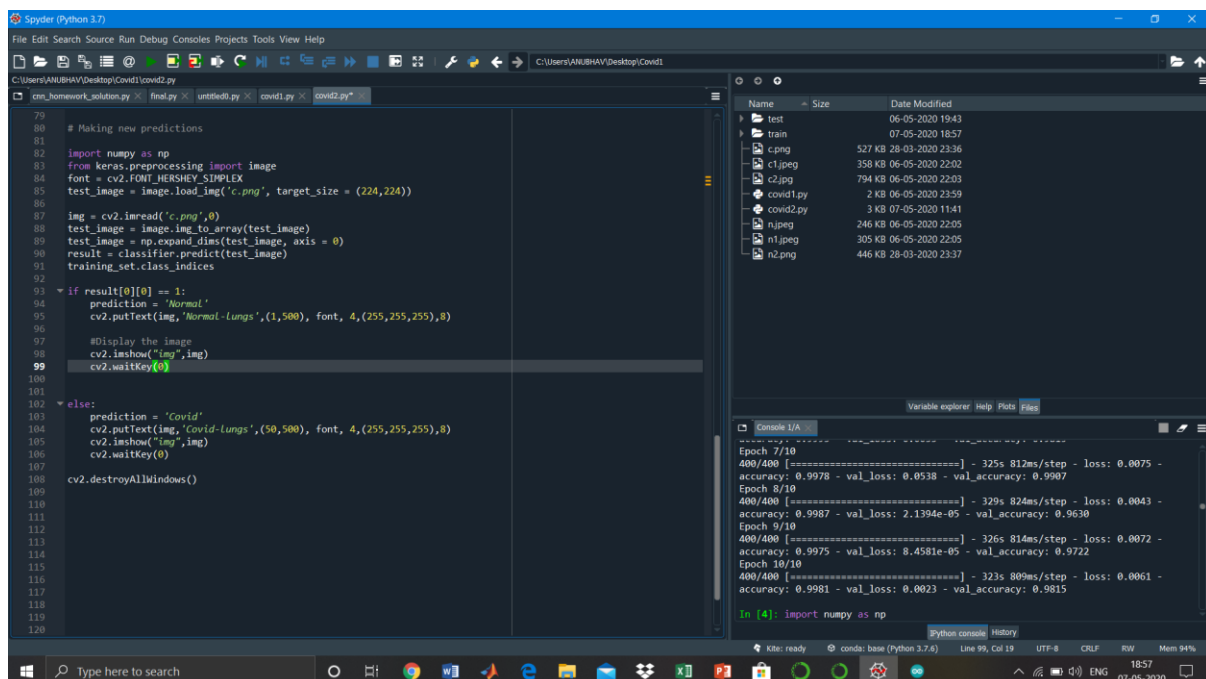
- 1.For a random Covid-19 affected patient lung X-ray

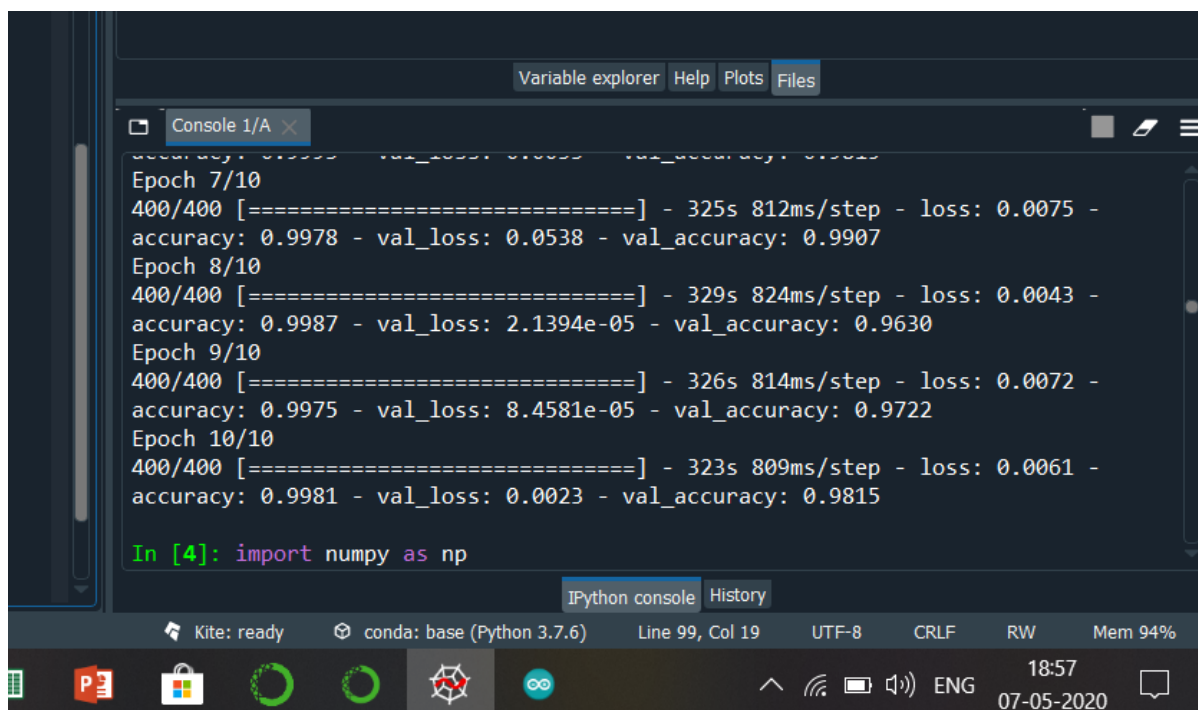


2. For a normal person's lungs X-ray :



3. Computational Accuracy :





```

Epoch 7/10
400/400 [=====] - 325s 812ms/step - loss: 0.0075 -
accuracy: 0.9978 - val_loss: 0.0538 - val_accuracy: 0.9907
Epoch 8/10
400/400 [=====] - 329s 824ms/step - loss: 0.0043 -
accuracy: 0.9987 - val_loss: 2.1394e-05 - val_accuracy: 0.9630
Epoch 9/10
400/400 [=====] - 326s 814ms/step - loss: 0.0072 -
accuracy: 0.9975 - val_loss: 8.4581e-05 - val_accuracy: 0.9722
Epoch 10/10
400/400 [=====] - 323s 809ms/step - loss: 0.0061 -
accuracy: 0.9981 - val_loss: 0.0023 - val_accuracy: 0.9815

In [4]: import numpy as np

```

Here we find that the validation accuracy is 98.15%. Also the train loss and validation loss are very close to each other proving that the model is perfectly fitted.

Conclusion :

Hence the Convolutional Neural Network was tested and the resulting accuracy was 98.15% without the model over-fitting or under-fitting.

References:

1. <https://missinglink.ai/guides/convolutional-neural-networks/fully-connected-layers-convolutional-neural-networks-complete-guide/>
2. <https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-step-3-flattening>
3. <https://machinelearningmastery.com/pooling-layers-for-convolutional-neural-networks/>
4. https://www.google.com/search?q=convolution+in+cnn&rlz=1C1CHBD_enIN886IN887&sxsrf=AleKk02H3MtNtCWb8y5RUF7rcEmaTSApZg:1588870541210&tbm=isch&source=iu&ictx=1&fir=oxdU4Yt5XuljbM%253A%252CjHhFke9jPucZbM%252C_&vet=1&usg=AI4_kTGTfY79di-myXwZ9TQxirX85zZg&sa=X&ved=2ahUKEwixsc-BnKLpAhWk6nMBHdUUDBsQ_h0wAHOECACQBA#imgsrc=oxdU4Yt5XuljbM

5. <https://www.freecodecamp.org/news/an-intuitive-guide-to-convolutional-neural-networks-260c2de0a050/>
6. <https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148>

