COVID-19 Chest X-Ray Classification Using Deep Learning

1. Introduction

Ever since the onset of COVID-19, there has been a threat to humanity. The healthcare system is over burden with shortage of staff, medicines, equipment etc. Doctors use Reverse Transcription-Polymerase Chain Reaction (RT-PCR) testing, one of the most widely used official screening methods for COVID detection. However, the cost of kits is too high and its availability is scarce making it difficult for health workers to diagnose COVID.

The other alternate and more reliable way to diagnose presence of COVID is using CXR (Chest X-Ray) as it shows abnormal symptoms, such as ground-glass opacity, bilateral abnormalities, and interstitial abnormalities. However, with limited health workers, computer can reduce the manual labour and classify a patient as normal or COVID infect using Deep Learning techniques. The challenge was to screen COVID patients from healthy people.

2. Dataset

The dataset was downloaded from online repositories (Kaggle and GitHub). The dataset contains 2168 images, having 1585 CXR of normal/healthy patients and 583 CXR of COVID-19 infected people.

The data was pre-processed as follows:

- Gaussian blur was used to remove noise from the image with kernel size of 5*5.
- Thresholding was used for better segmentation and analysis of the images.
- Basic morphological features such as erosion and dilation were used for getting sharper boundary lines and overall a shaper enhanced picture.
- Next, in order to remove the extra black space around the CXR, contour selection was used and the picture was cropped to retain only the necessary information.


```
array([[32, 34, 33, ..., 31, 32, 33],
       [33, 34, 32, ..., 33, 34, 34],
       [34, 33, 30, ..., 35, 35, 34],
       ...,
       [28, 28, 28, ..., 34, 33, 34],
       [28, 29, 29, ..., 34, 33, 34],
       [29, 29, 30, ..., 35, 34, 34]], dtype=uint8)
```

Figure 1: Image preprocessing details

The images were resized to (224,224) to be fed into the model. The dataset was split into training and testing set (test_size=0.3). The details are as follows:

number of training examples = 1517

number of test examples = 651

X_train shape: (1517, 224, 224)

Y_train shape: (1517, 1)

X_test shape: (651, 224, 224)
Y_test shape: (651, 1)

3. Model

The model is inspired by VGGNet-16 and ResNet. VGGNet-16 forms the backbone of the architecture whereas the skip connections in between the layers help to solve the problem of 'Degradation' and 'Vanishing/Exploding gradients.'

As the network goes deeper and deeper and becomes more complex, it tends to overfit. Below is a figure showing training and test error of 20- and 56-layers neural network (NN) wherein the 20-layer NN performs better.

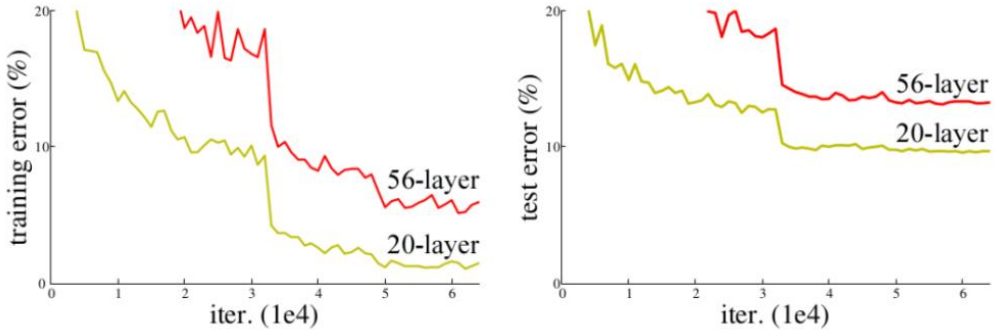

Figure 2: Denser networks are not always better

The other problem of exploding and vanishing gradients occur during the backpropagation wherein the gradients of the initial layers either don't change much or change drastically making it impossible for the model to converge to optimum.

These two problems are overcome by using skip connections which create a connection between the l^{th} layer and $l+2^{th}$ or $l+3^{th}$ layer and so on. It ensures the flow of information from lower to much higher layers and vice versa during back propagation.

Each of the convolutions in VGGNet-16 use 3*3 filters, stride of 1 and retain padding. The max pooling layer uses 2*2 filters and stride of 2. The convolution in the skip connection uses 1*1 filter size and stride of 2.

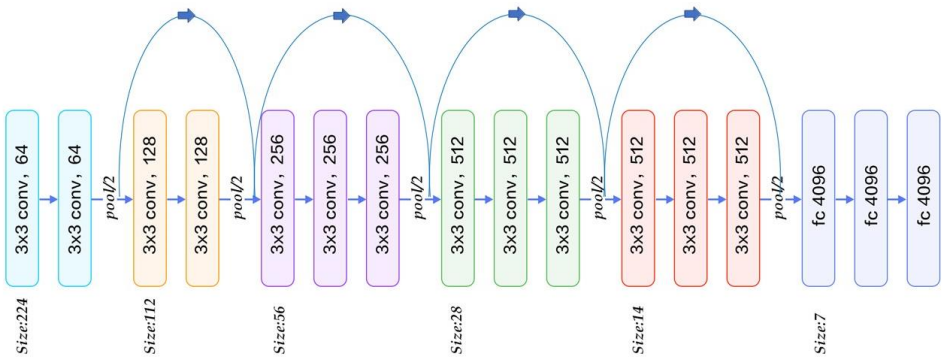

Figure 3: High level picture of the model architecture

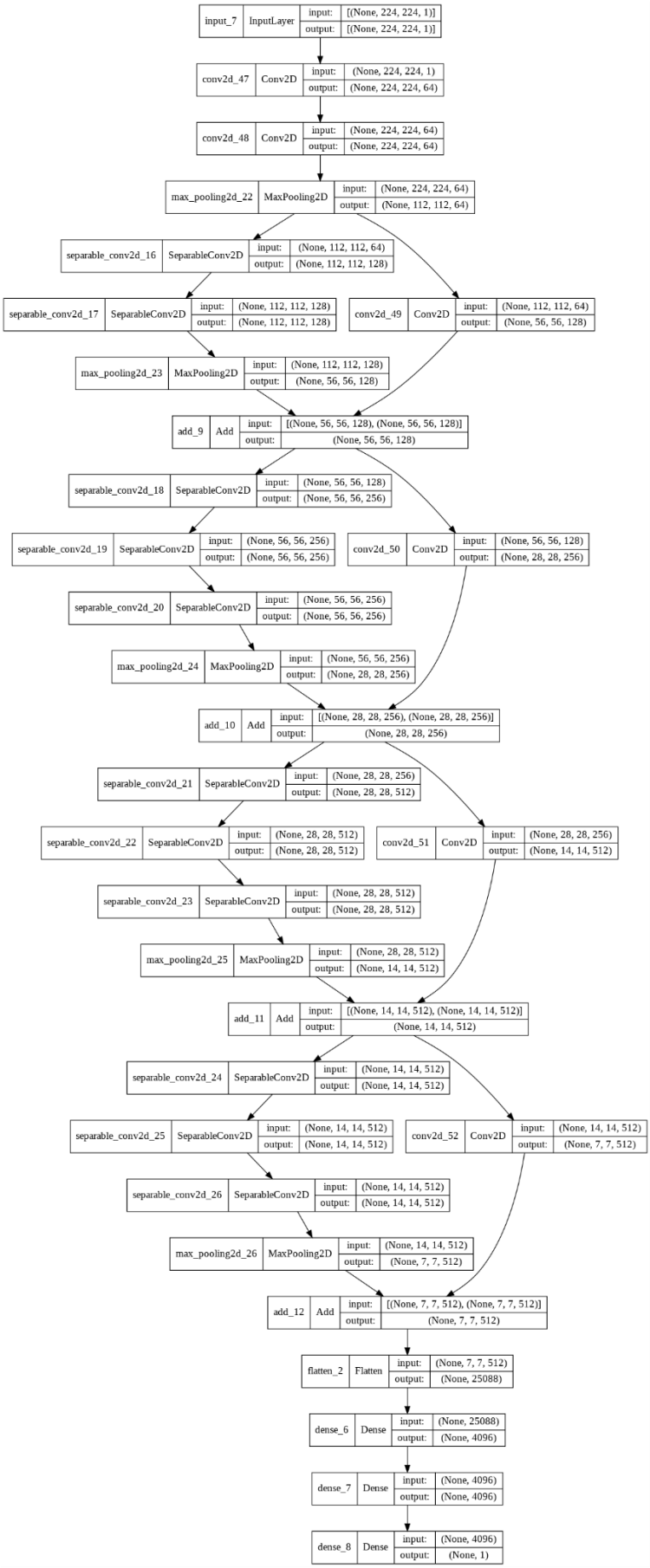

Figure 4: The complete overview of the model architecture

The input shape of (224,224) is finally converted to (4096,1) to give a classification.
A total of 121,690,241 parameters were trained.

4. Results

The model was trained for 50 epochs, with *adam* optimizer, *binary cross entropy* to calculate loss function and to give *accuracy* as the metric to evaluate the model. Upon validating on test set, the results were Test loss: 0.1427641361951828, Test accuracy: 0.981566846370697. The below graph show model accuracy and loss.

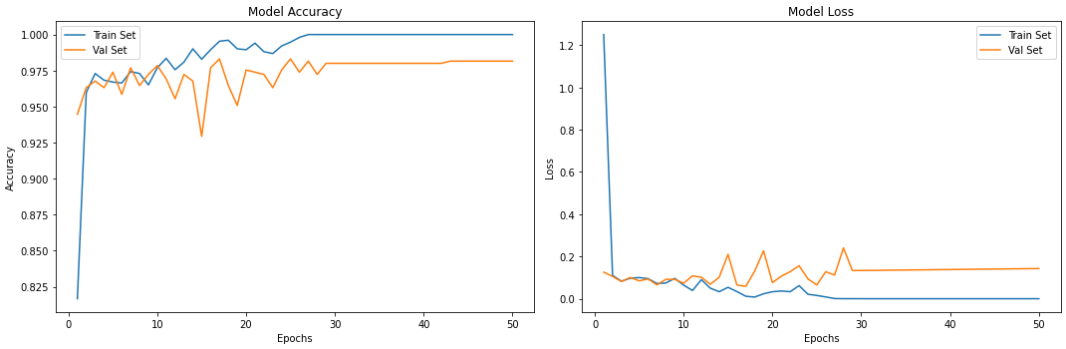

Figure 5: Model Accuracy and Loss

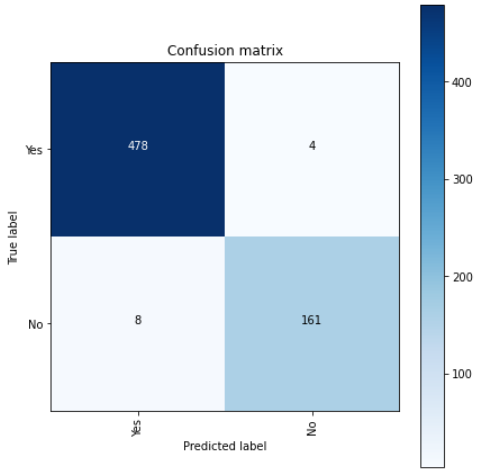

Figure 6: Confusion matrix to show positive predicted and negative predicted values

5. Limitations

This model architecture took 54 mins 03 seconds to complete 50 epochs giving test accuracy of approx. 98.15%. However, a simple VGGNet-16 architecture takes 34 mins 15 secs to complete the processing and gives an accuracy of 95.23%. Thus, a trade off exists between the computation cost and accuracy.