Pneumonia Detection On Chest X-rays using Convolutional Neural Networks

Devarasetti Kali Prasad
AI-Technology & Systems
http://www.ai-techsystems.com
kaliprasadeducation@gmail.com

***Abstract*: Developed a very basic Convolutional Neural Network that can detect whether a person has Pneumonia using X-Ray images, Instead of using pretrained networks with more weights, tried to use very few layers and get state of the art results, Proposed deep learning model produces a test accuracy of 93.5%, F1 score of 0.94 with a 0.97 recall rate on Pneumonia detection.**

Keywords: X-Ray image classification, Pneumonia detection, Deep Learning, Convolutional Neural Networks.

I. INTRODUCTION

Pneumonia Detection using X-Ray's is a very important task in medical industry, often medical Industry relies on trained radiologists to determine whether the person has pneumonia or not, in this paper a very few layered convolutional neural network model is proposed that can automatically detect Pneumonia on par with the radiologists. The model is trained on 5216 X-Ray's containing both Pneumonia and Normal, and tested on 624 chest X-Rays.

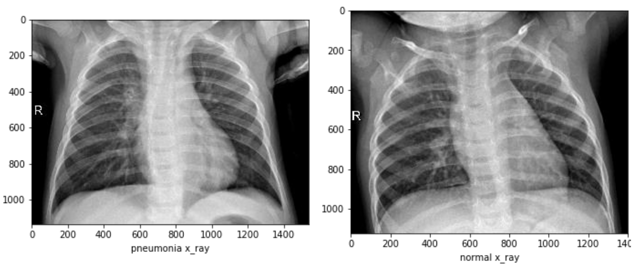

Figure 1. Pneumonia x-ray on left side and Normal x-ray on right side.

A. Constraints:

At any cost our model should not classify a Pneumonia patient X-Ray as normal, So we should target very high recall rate, Also we should not lose the precision too. So there will always be trade-off between both precession and recall, We can consider F1-score as a good metric for this problem, with special emphasize on recall rate.

II. DATA

A. Data Source and Basic Exploration:

Data we used to train our model is from Kaggle, Dataset contains three folders, train, test and val,

Further each folder contains two folders normal and Pneumonia. So problem we are trying to deal is a binary classification problem.

There are total 5216 train images, 16 validation images and 624 test images. Number of validation images given to us are low in number.

B. Data Augmentation:

As Data Augmentation will have serious impact on model performance following augmentation steps are considered.

Horizontally flipped every image, Rescaled every pixel of image between 0 and 1, Also as every image doesn't have a fixed shape, tried rescaling into different shapes like (64, 64), (128,128), (256, 256) etc, after trying different image shapes, sticked on with (128, 128) shape for the final model, Used Keras ImageDataGenerator for data augmentation, also find out the keras default shape (224, 224) doesn't perfom well and often stuck on same validation accuracy when used the default shape.

III. ARCHITECHTURE

A. Baseline Model:

Baseline model acts as the reference for the further models we build, and acts as the minimum performance comparison model to which all the other models will be compared.

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 254, 254, 32)	896
max_pooling2d_1 (MaxPooling2	(None, 127, 127, 32)	0
flatten_1 (Flatten)	(None, 516128)	0
dense_1 (Dense)	(None, 100)	51612900
dense_2 (Dense)	(None, 1)	101

Total params: 51,613,897
Trainable params: 51,613,897
Non-trainable params: 0

Figure 2. Base line model Summary

Baseline model contains a single block of conv2d with one max pooling layer and and one fully connected layer and a output layer.

Baseline model is giving training accuracy of 0.74 with a constant validation accuracy of 0.5.

Problem of constant validation accuracy is being taken care by adjusting the class weights.

B. *Final Model:*

Final Model consists of four blocks of convolution layers followed by maxpooling layer, and three fully connected layers followed by final output layer.

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_51 (Conv2D)           (None, 128, 128, 128)     3584
_________________________________________________________________
max_pooling2d_50 (MaxPooling (None, 64, 64, 128)       0
_________________________________________________________________
conv2d_52 (Conv2D)           (None, 64, 64, 32)        36896
_________________________________________________________________
max_pooling2d_51 (MaxPooling (None, 32, 32, 32)        0
_________________________________________________________________
conv2d_53 (Conv2D)           (None, 30, 30, 128)       36992
_________________________________________________________________
max_pooling2d_52 (MaxPooling (None, 15, 15, 128)       0
_________________________________________________________________
conv2d_54 (Conv2D)           (None, 15, 15, 128)       147584
_________________________________________________________________
max_pooling2d_53 (MaxPooling (None, 7, 7, 128)         0
_________________________________________________________________
flatten_23 (Flatten)         (None, 6272)              0
_________________________________________________________________
dense_47 (Dense)             (None, 128)               802944
_________________________________________________________________
dense_48 (Dense)             (None, 256)               33024
_________________________________________________________________
dense_49 (Dense)             (None, 64)                16448
_________________________________________________________________
dense_50 (Dense)             (None, 1)                 65
=================================================================
Total params: 1,077,537
Trainable params: 1,077,537
Non-trainable params: 0
_________________________________________________________________
```

Figure 3: Final Model layers summary

$$L(X, y) = -\sum I(y, i) \log p(Y = i|X)$$

Where y is the output label generate by the network,

p(Y= i|X) is the probability that the network assigns the label I given the input data, and $I(y, i)$ is the indicator function define by

$$I(y, i) = \begin{cases} o \; if y \neq i \\ 1 \; if y = i \end{cases}$$

We are training using Adam optimizer with a learning rate of 0.01.

For each input we optimize the binary cross entropy loss,

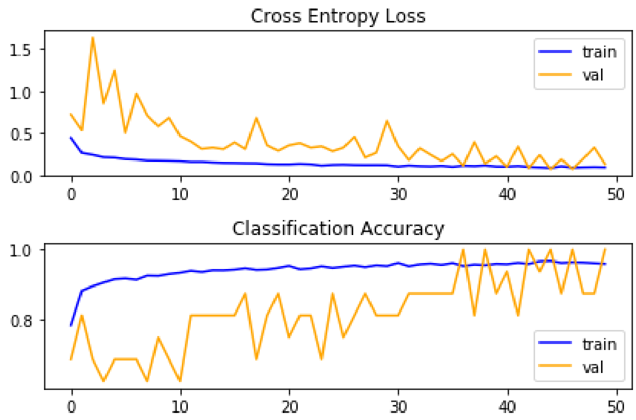

Figure 3(upper): Loss vs Epochs graph

Figure 3(lower): Accuracy vs Epochs graph

IV. Results

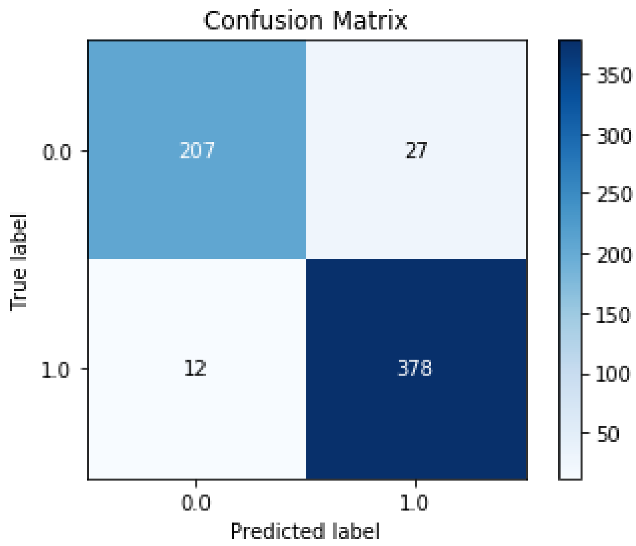

Figure5: Confusion Matrix

Our simple model only predicts 12 Pneumonia cases as Normal cases out of total 390 Pneumonia test cases. Also 27 Normal Cases are predicted as Pneumoina cases out of total 234 Cases. We can still further tune our model based on the specific requirement, Also experimented increasing recall rate to 0.98 but it came with a cost of 2% decrease in the Pneumonia precission.

	precision	recall	f1-score	support
Normal	0.95	0.88	0.91	234
PNEUMONIA	0.93	0.97	0.95	390
accuracy			0.94	624
macro avg	0.94	0.93	0.93	624
weighted avg	0.94	0.94	0.94	624

Figure6: Precision Recall F1score

Our model achieved 97% highest recall rate on Pneumonia detection with a Precision of 93% and F1 score of 0.94.

REFERENCES

[1] Yann LeCun, Patrick Haffner Leon Bottou, Yoshua Bengio Object Recognition with Gradient Based Learning.

[2] https://en.wikipedia.org/wiki/Cross_entropy

[3] https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimization-6be9a291375c