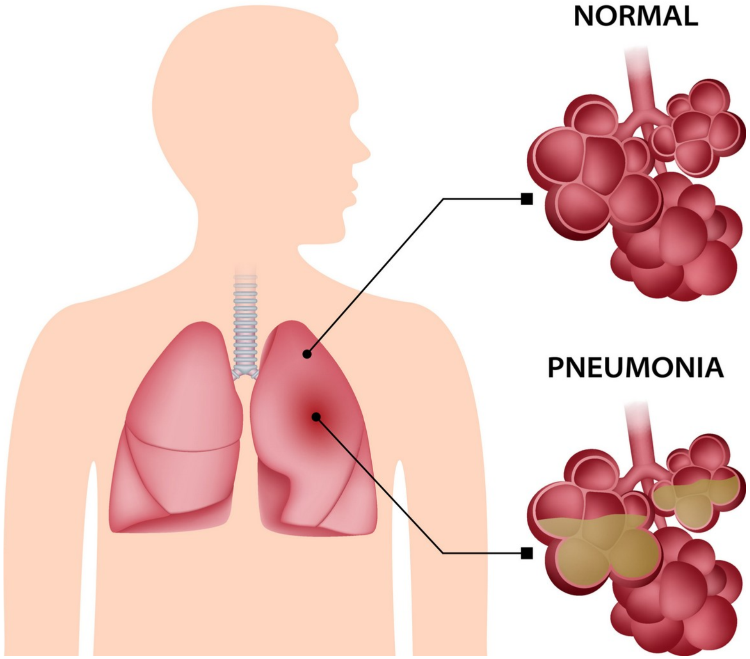
Pneumonia Prediction from Chest X-ray Images

(Under the supervision of Prof. Sujoy K. Biswas)

Srijan Mallick
M.Sc. 1st year
Big Data Analytics
Reg. No: B2030050

Aim and Motivation

- Pneumonia is an infection that inflames air sacs in one or both lungs, which may fill with fluid, leading to a difficulty in breathing.
- Life threatening to infants and elderly.
- A deep learning model which would accurately predict pneumonia would aid radiologists and save time and cost.



Data Overview

- Data source : https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia

- Data comprises of 5856 chest X-ray images, obtained from pediatric patients of 1-5 years of age, from the Guangzhou Women and Children's Medical Center, China.

- Images divided in 2 classes : Normal and Pneumonia

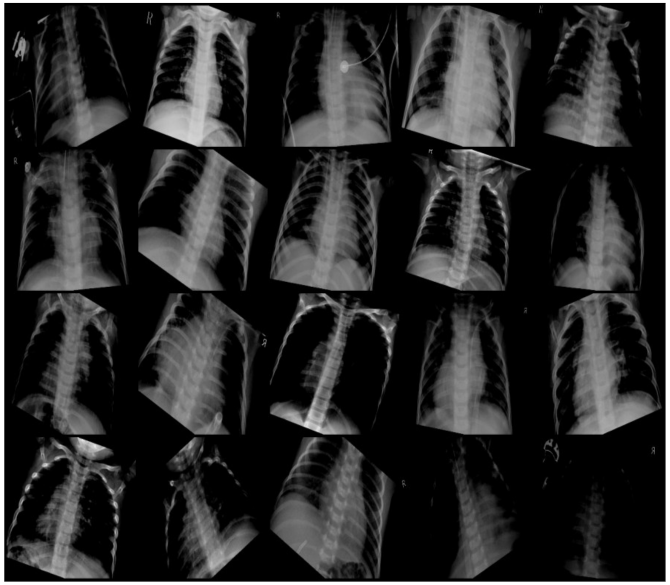
Normal

Pneumonia

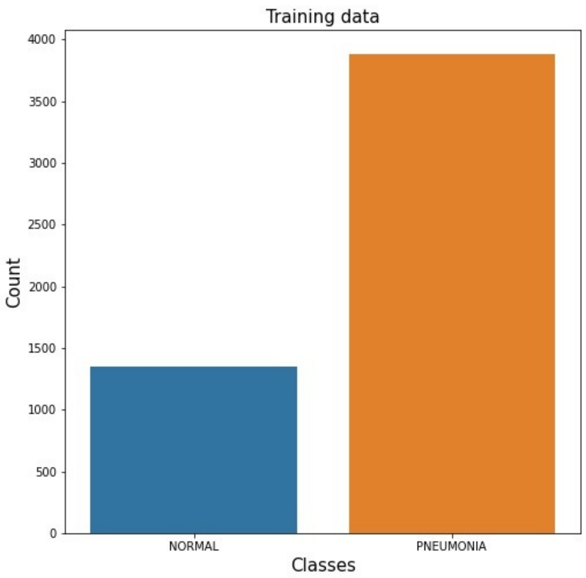
Data Visualisation

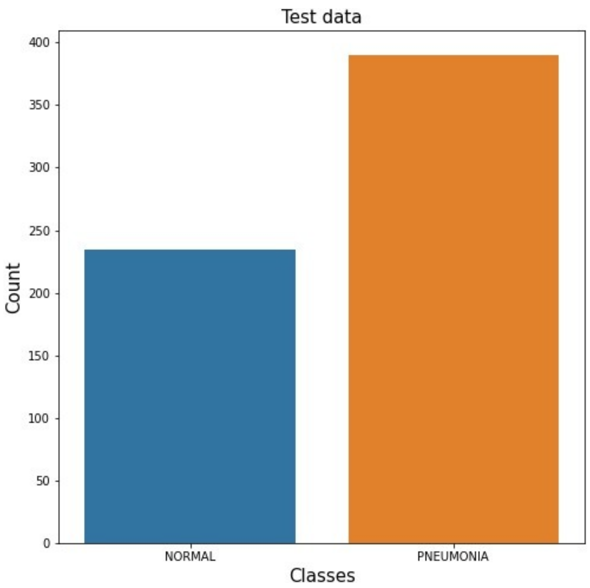
Final split:

- Train – 4186
- Validation –1046
- Test - 624



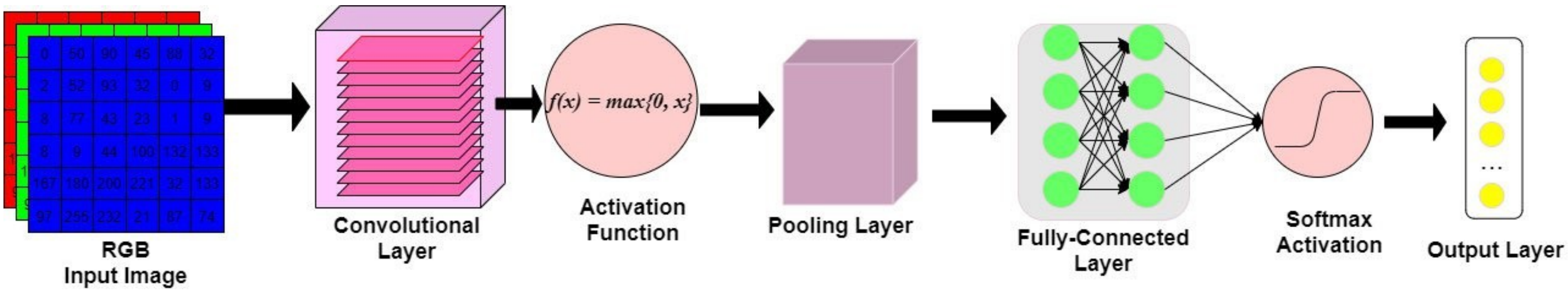
Batch of augumented Chest X-ray images of size 20





Data Statistics

CNN Architecture Overview




```
ConvNet(
  (conv1): Conv2d(3, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (conv2): Conv2d(16, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (conv3): Conv2d(32, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (conv4): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (conv5): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (bn1): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (bn2): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (bn3): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (bn4): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (bn5): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (pool): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (fc1): Linear(in_features=16384, out_features=1024, bias=True)
  (fc2): Linear(in_features=1024, out_features=512, bias=True)
  (fc3): Linear(in_features=512, out_features=2, bias=True)
  (dropout): Dropout(p=0.5, inplace=False)
)
```

8-layer deep CNN model used

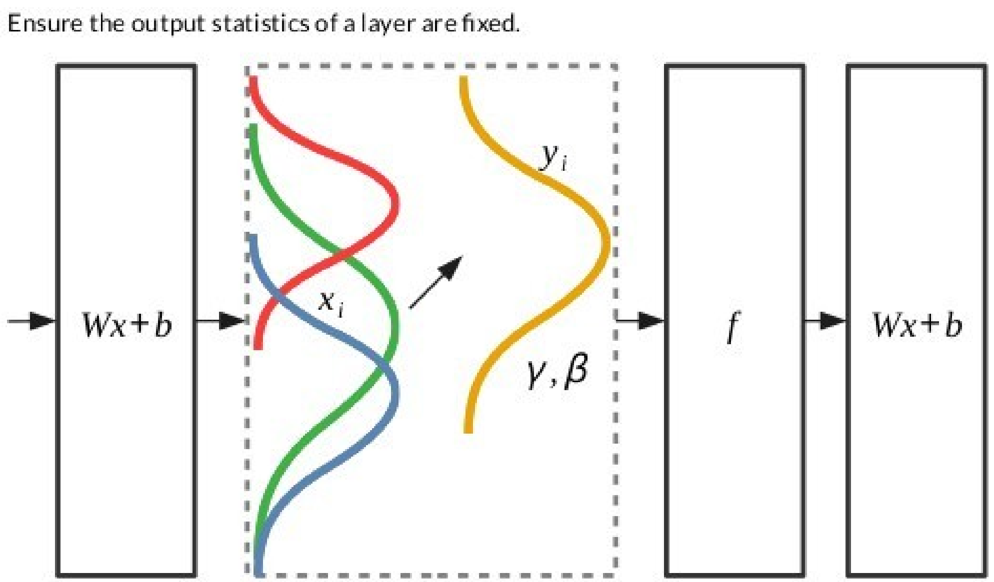
- Convolution
- Batch Normalisation
- ReLU
- Max Pool
- Dropout

Involved operations

CNN Architecture Overview

The need for Batch Normalisation

- **Covariate shift** : Different distribution of the features in different parts of the dataset.
- **Internal Covariate shift** : Covariate shift observed within the network layers due to the distribution of weights and activations.
- **Batch Normalisation** ensures common means and variances of layer inputs which helps in faster convergence and improved gradient flow through the network.



CNN Architecture Overview

Hyperparameters used

- **Adam optimiser** with

Learning rate : 0.008, with

Exponential Decay of 0.2

Weight decay : 0.02

Epochs : 30

- **SGD optimiser** with

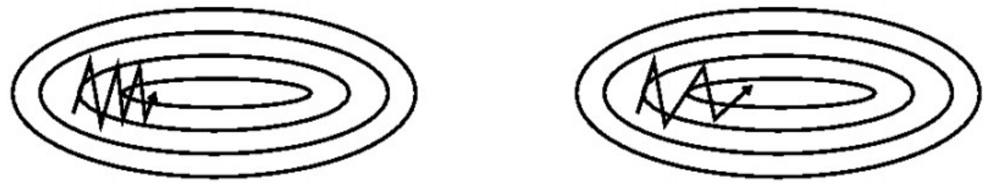
Learning rate : 0.008, with

Exponential Decay of 0.2

Momentum : 0.9

Nesterov accelarated gradient

Epochs : 20

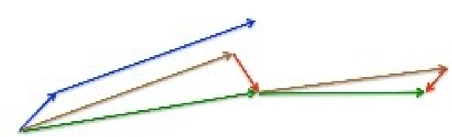


(a) SGD without momentum

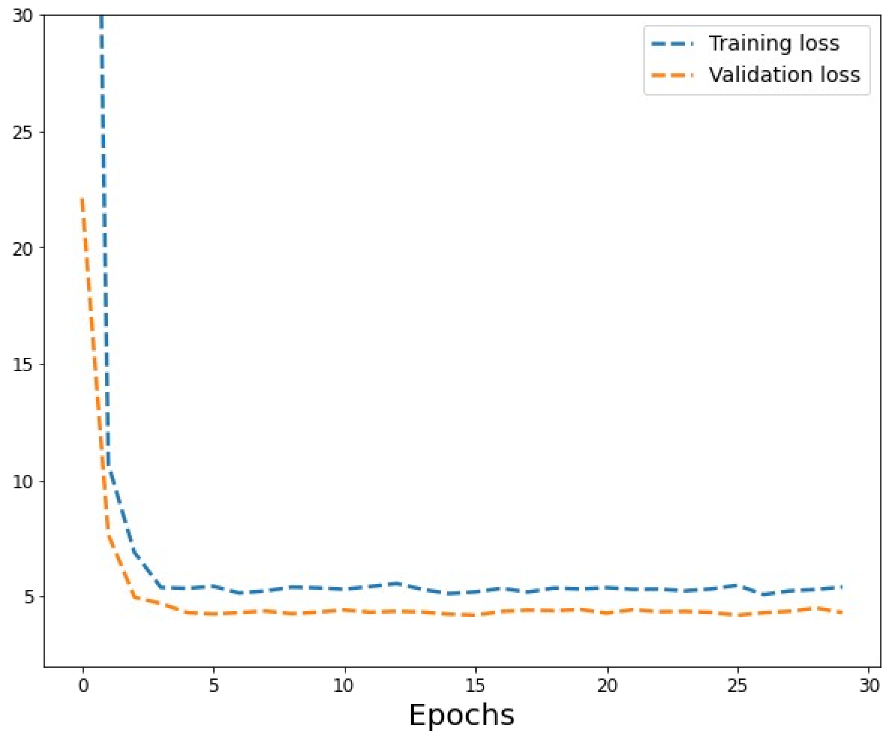
(b) SGD with momentum

- Momentum **blindly accelerates** down slopes: First computes gradient, then makes a big jump.
- Nesterov accelerated gradient (NAG) [Nesterov, 1983] first makes a big jump in the direction of the previous accumulated gradient $\theta - \gamma v_{t-1}$. Then measures where it ends up and makes a correction resulting in the complete update vector.

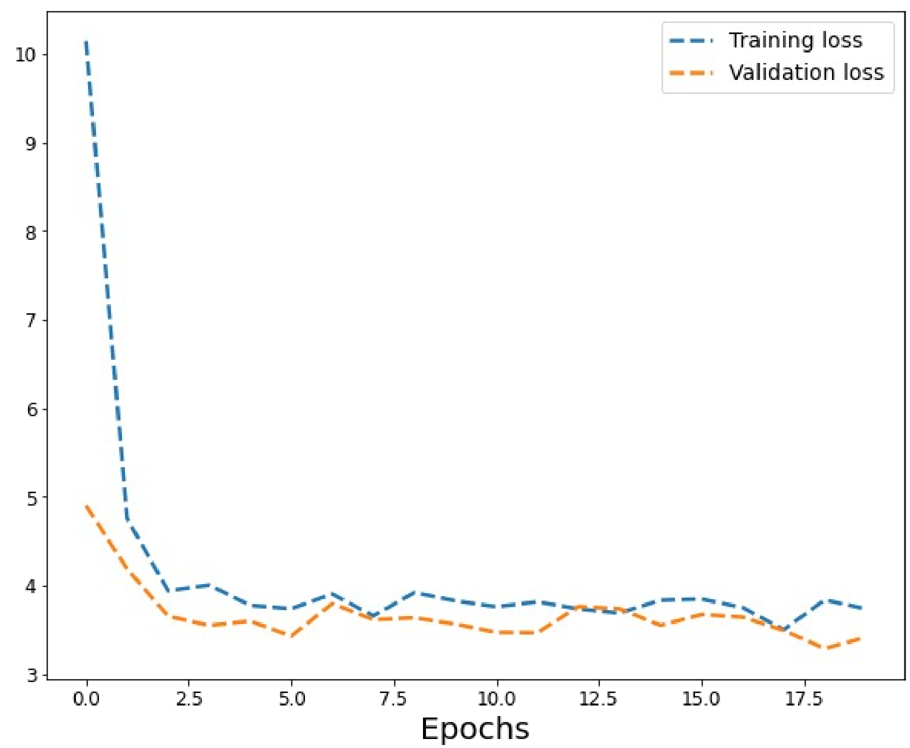
$$v_t = \gamma\ v_{t-1} + \eta \nabla_\theta J(\theta - \gamma v_{t-1})$$
$$\theta = \theta - v_t$$



Comparing Training/Validation loss

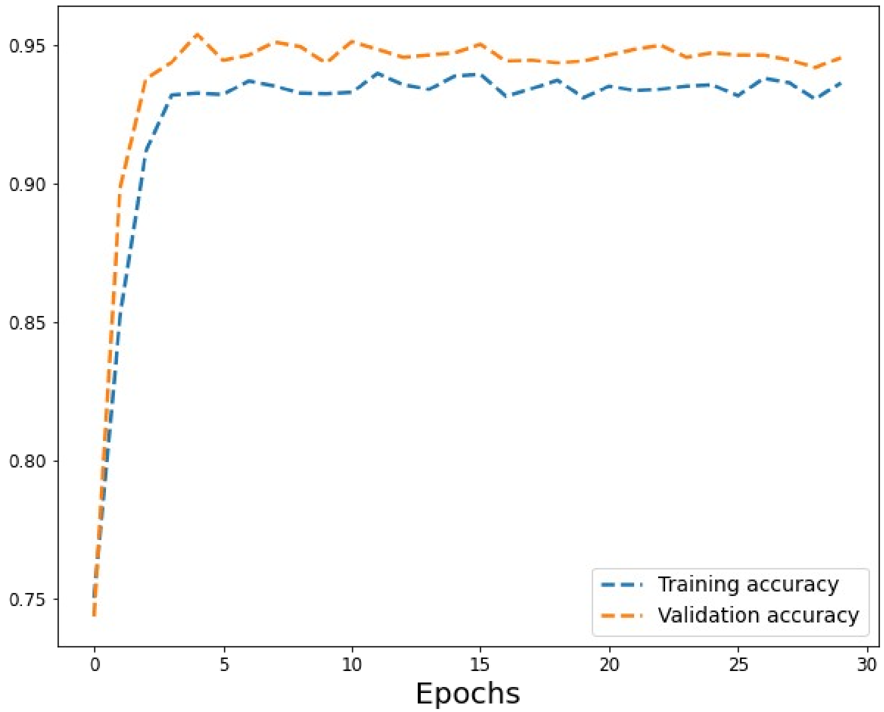


Using Adam optimizer

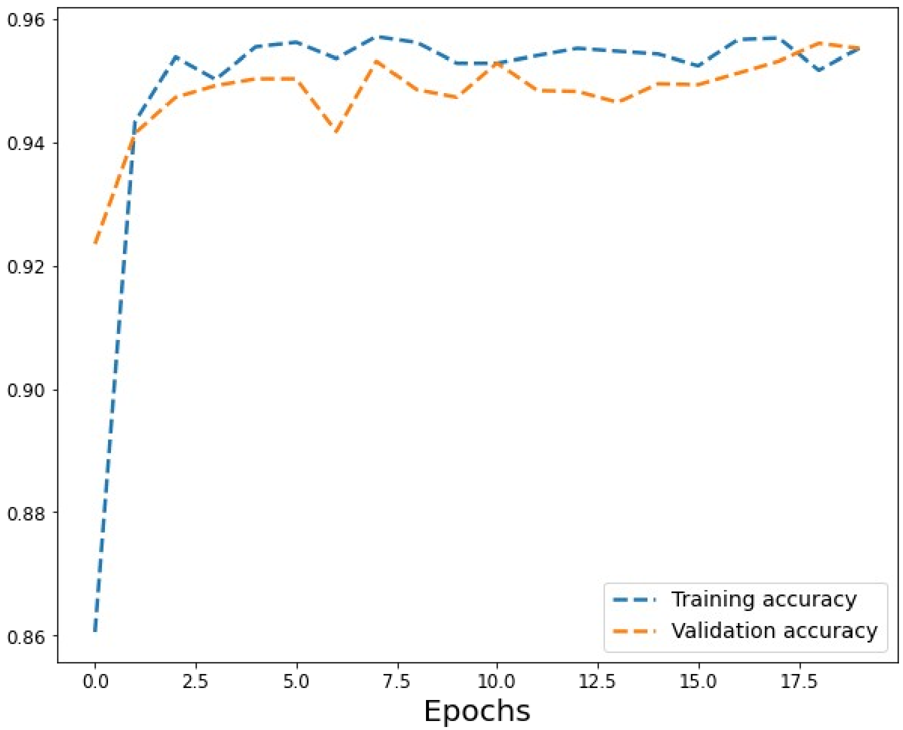


Using SGD with
Nesterov accelaration

Comparing Training/Validation accuracy

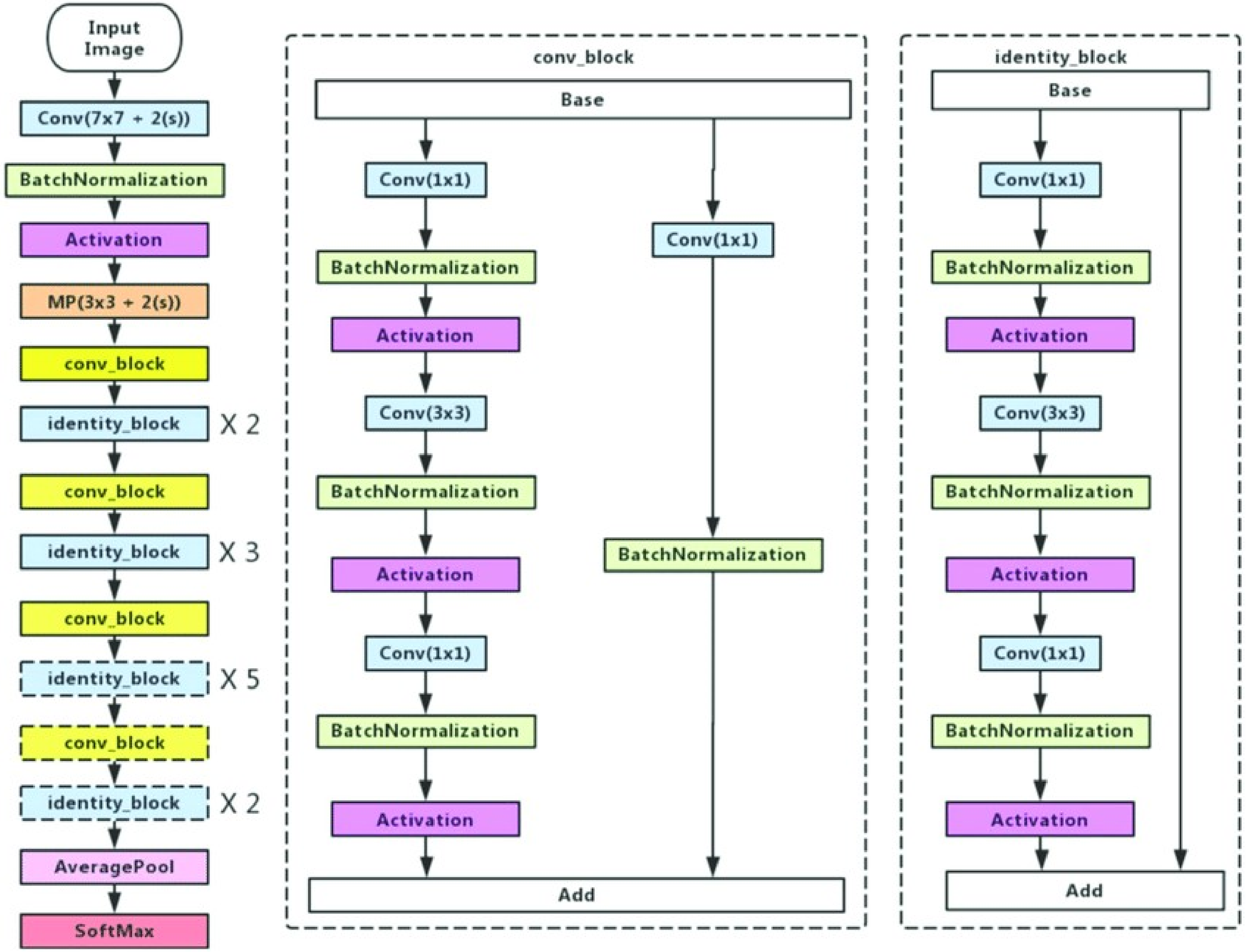


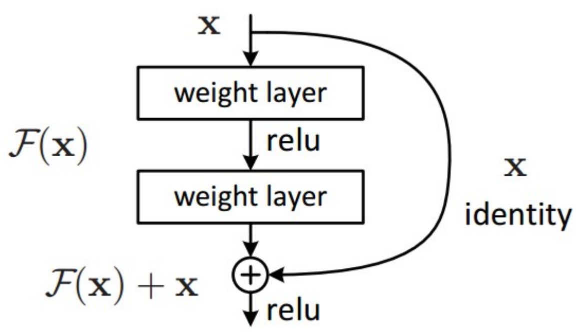
Using Adam optimizer



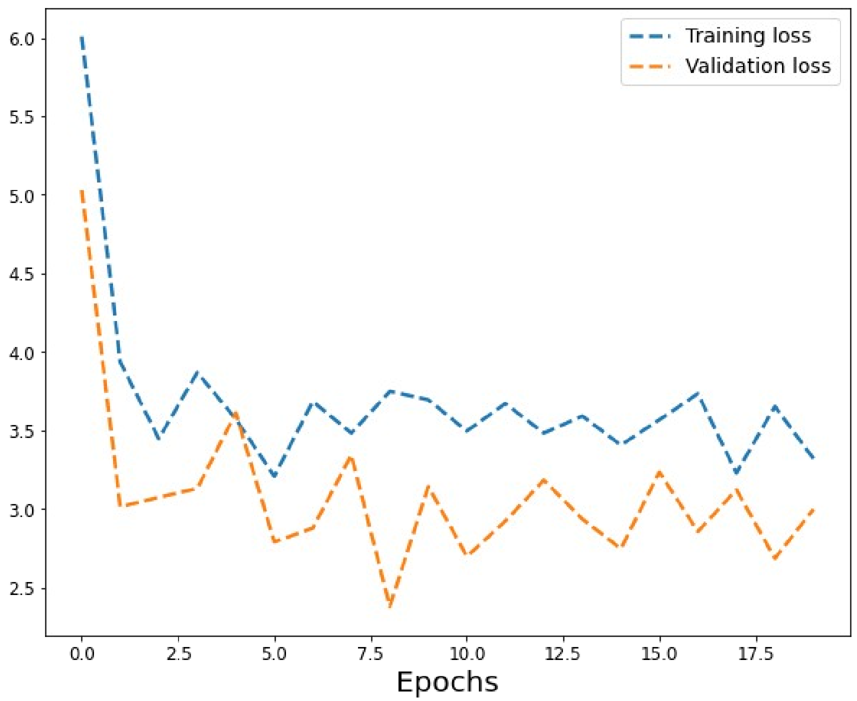
Using SGD with
Nesterov accelaration

Transfer Learning using ResNet50 backbone

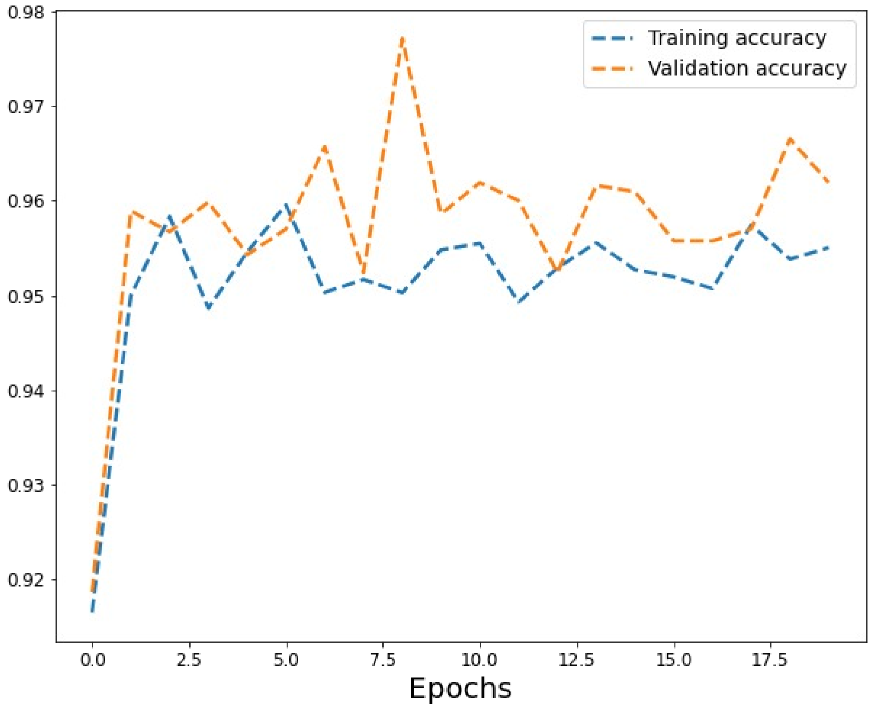




Loss and Accuracy obtained using ResNet50 backbone



Training/Validation Losses



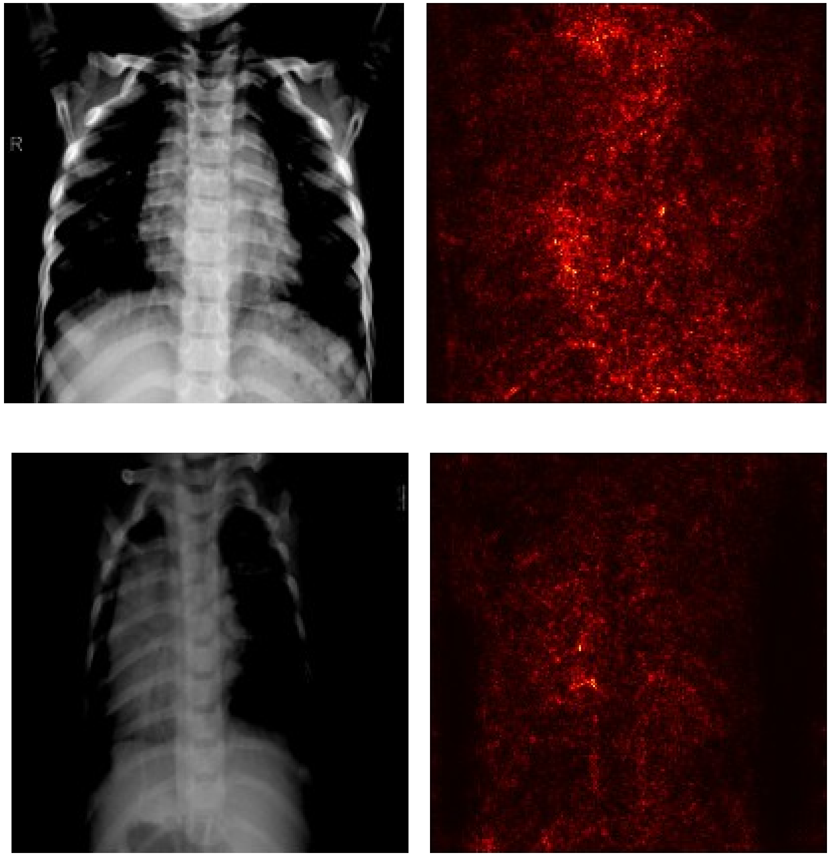
Training/Validation Accuracy

Comparing the different approaches

		8 layer-deep CNN model		ResNet50 model
		Adam	SGD with Nesterov Accelaration	
Min Training loss		4.8844	3.4412	3.2076
Min Validation loss		3.8515	3.0245	2.3778
Test Loss		0.5391	0.7645	0.3771
Highest Training accuracy		94.19%	95.71%	95.95%
Highest Validation accuracy		95.39%	96.36%	97.71%
Test accuracy	NORMAL	49%	49%	73%
	PNEUMONIA	97%	98%	95%
Overall test accuracy		79%	80%	87%

Visual Saliency

The Image and Its Saliency Map



- How does the neural network arrive at its decision ?
- Which portions of an image are ‘salient’, ie play a more important role in this process ?

Saliency maps specifically plot the gradient of the predicted class from the model with respect to the input, or pixel values.

$$Y_c = \text{score of class c}$$
$$\text{saliency} = \max_{r,g,b}\left(\left|\frac{\partial Y_c}{\partial I}\right|\right)$$

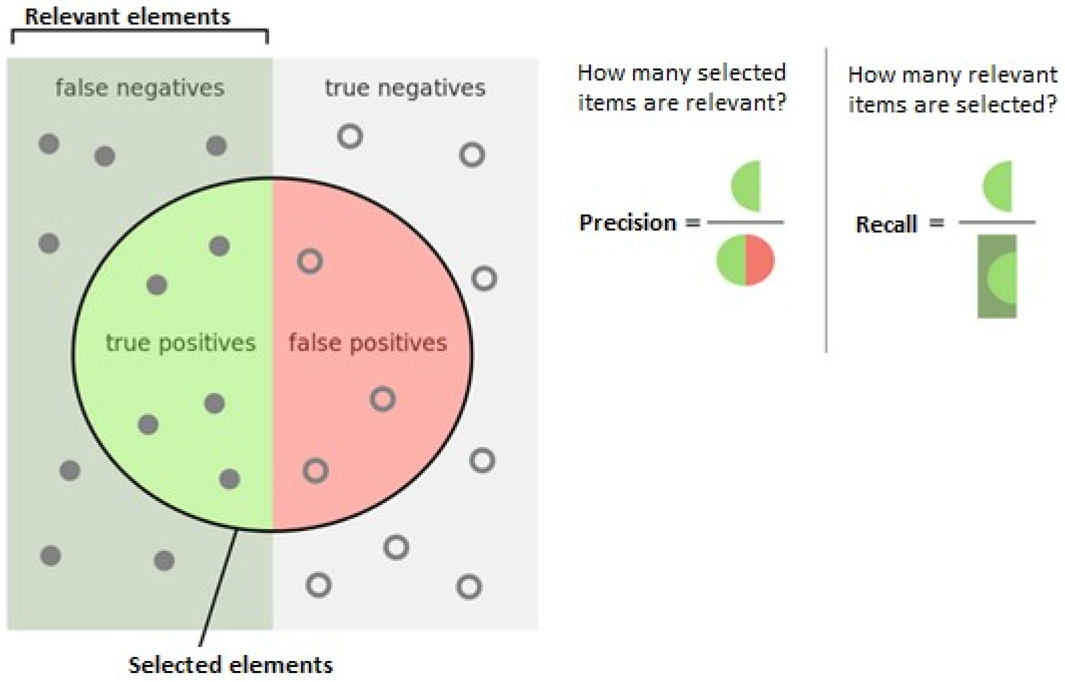
An improved approach : Selvaraju, R.R. et al,*“Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization”*, Link : https://arxiv.org/abs/1610.02391

Evaluation Metrics

$$precision = \frac{TP}{TP + FP}$$

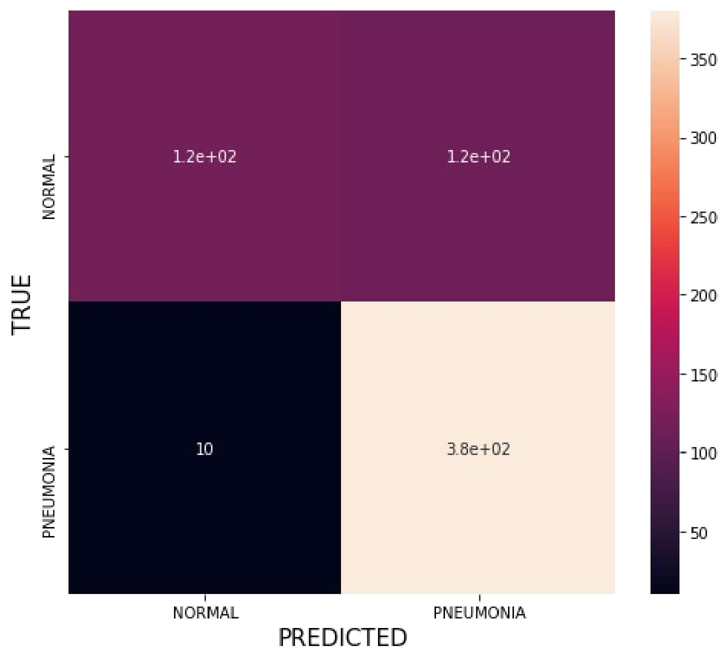
$$recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$



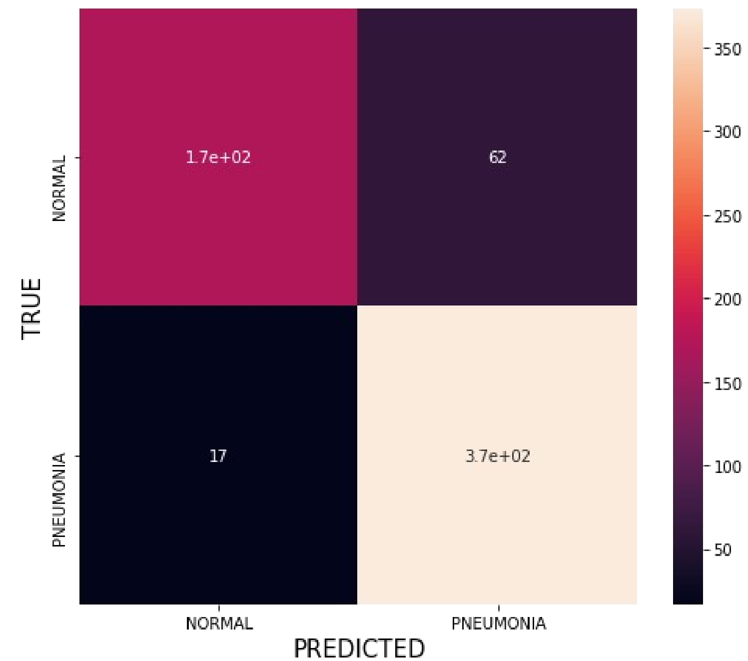
Model Evaluation

Confusion Matrix obtained from
8-layer deep CNN model



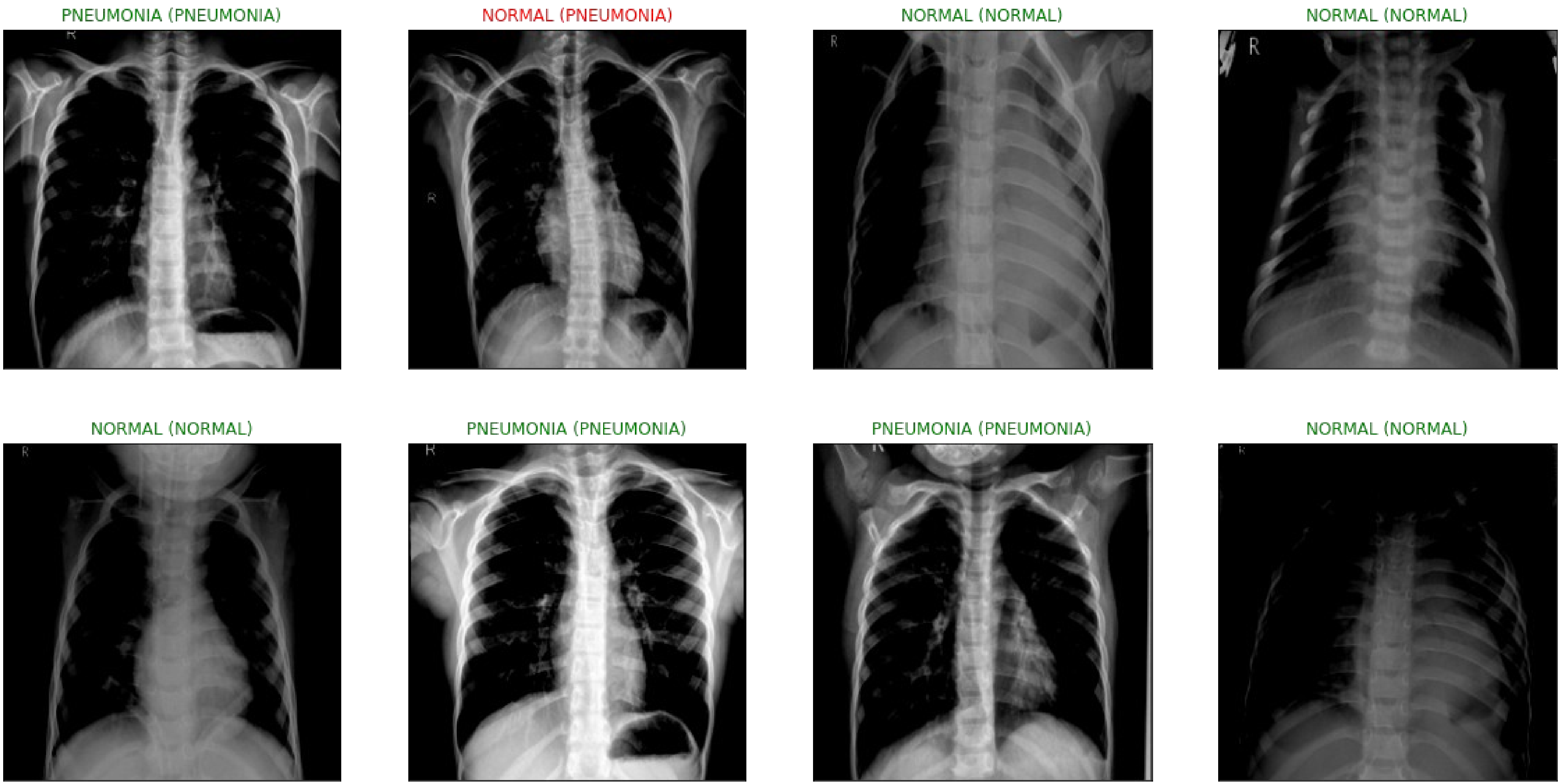
Precision: 0.7677
Recall: 0.9744
F1 Score: 0.8588

Confusion Matrix obtained from
ResNet50 model



Precision: 0.8575
Recall: 0.9564
F1 Score: 0.9042

Prediction Results



Thank You !