CSE583/EE552 Pattern Recognition and Machine Learning: Term Project Final Report

Due on May 4, 2020 at 11:59pm

PROFESSOR Yanxi Liu Spring 20

Srikanth Banagere Manjunatha **sxb5973@psu.edu**

Pneumonia Detection using X-Net

Abstract

Pneumonia is a lung infection which can be Bacterial or Viral. The infection is widely found in children below ten years. Pneumonia is the largest cause of death in children worldwide. There is a necessity of detection of the infection in early stages to start the treatment in the early stages to cure the disease. Chest X-rays serve as the primary source of detection even to these days. There is a need for prediction of the presence of Pneumonic infection in large quantity. In this course project, we realize the Baseline model and also propose a novel architecture to predict the presence of Pneumonic infection or not in the several chest X-ray images. Various experiments are conducted justifying the use of the various methods followed in the course of this project. A detailed report of the various methods incorporated and the background is presented to support our claim. The model's performance is largely dependent on the available Data set and hence a detailed study of the Data set is presented to validate the model's generalizability. The results are presented as scores as well as visualized in graphs and figures for better understanding.

Contents

Introduction **3**

Problem Statement **3**

Goals **4**

Motivation and challenges **4**

Related Works **4**

Dataset Description **5**
Is the dataset sufficient? Does it have enough samples for training? 5

State-of-the-art - Baseline **6**

Proposed Methods **8**

Proposed Model - XNet **8**

Data Augmentation **11**

Novelty of the Term Project **12**

Evaluation metrics **12**

Experiments **14**
Data Study using KNN 14
Unsupervised Learning approach: K-Means 15
Baseline Implementation 15

Results and Observations **16**
Data Study using K-Nearest Neighbor 16
Inference 17
Unsupervised Learning approach: K-Means 17
Inference 17
Baseline Implementation without Data Augmentation 17
Inference 18
Baseline Implementation with Data Augmentation 21
Inference 25
X-Net Model 25
Inference 28

Issues faced during Term Project **29**

Future Work **29**

Conclusions **29**

Introduction

Pneumonia is an infection caused in lung(s) and the infection can be bacterial, fungal or viral. The infection results in the filling up of alveoli with fluid or pus which makes it hard for the person to breathe. People with weakened immune system and young children and elder people are at higher risk of acquiring Pneumonia. Pneumonia infections can be fatal. According to the World Health Organization (WHO), "Pneumonia is the single largest cause of death in children worldwide. Every year, it kills an estimated 1.4 million children under the age of five years, accounting for 18% of all deaths of children under five years old worldwide." It is important to note that the chest X-rays are a very good source for the detection of Pneumonia.

It is very important that the chest X-rays are properly processed. Employing Deep Learning models to efficiently classify the Pneumonic infected lungs and the non-infected lungs using the chest X-rays is hence a very interesting topic in Machine Learning. Prior to outbreak of Deep Learning, traditional pattern recognition tasks were performed using an initial pre-processing, followed by a careful-intelligent feature extraction, and then the features are used for the classification task. After the deep learning models started gaining popularity with high computation, the feature extraction step has been bypassed in many applications and the input is directly fed to the classifier. For the image processing task, the Convolution Neural Networks gained lot of popularity, as the number of parameters to be learnt effectively reduces as the convolution kernels scan through the whole image.

In this project, we employ Convolutional Neural Network architecture as baseline and improvement over the architecture based on the proposed method in the context of classification of the Pneumonic infected and non-infected lungs using chest X-rays. The open source dataset from the Kaggle website has been employed for the classification task. The classification is performed based on the famous Adam optimizer using Tensorflow, Keras and Python libraries. The performance will be evaluated on the test set which was divided prior to training. Also, the data augmentation techniques make sure that the model sees more samples from the training set that are different versions of the same image with some transformation or some noise added to the original image. These techniques have proved to be very effective in terms of the performance improvement.

The document is organized as follows. The next section discusses the title of the proposed project, followed by the Problem statement and description. The next section discusses the goals and later a separate section is dedicated to Motivations and challenges. A brief related work discussion is also carried out which covers various research works carried out in context of Pneumonia classification using deep learning, and in context of the dataset employed in the project. A small discussion in terms of the prior research regarding the proposed methods are also included in the same section of related works. A section is dedicated to discuss the State-of-the-art work carried out through the project followed by the novel ideas which are planned to be implemented. A separate section is dedicated to discuss the data set employed to carry out the classification and important questions pertaining to the dataset are answered in this section. The methods planned in course of the project are briefly discussed including the various techniques that are implemented. Specific steps that will be implemented is discussed in the next section with a separate discussion on the evaluation metrics and expected outcome. If the proposed methods do not go as planned, the alternatives planned for the evaluation is also discussed, followed by the timeline.

Problem Statement

Building a deep learning model to effectively classify the Pneumonia patients and the non-Pneumonia patients using the chest X-ray scans.

Proposing a deep convolutional network model for the task of Pneumonia detection. A study is done comparing the baseline and the proposed architecture. The baseline for Pneumonia detection is around 93%. With the proposed model, we expect to see an improvement in the baseline. The architecture proposed is a hybrid of various techniques implemented separately. This proposed architecture is employed using the

X-ray scans for the Pneumonia detection. To evaluate the model's generalizability, a study is done on the distribution of the training and test dataset, based on unsupervised learning algorithm of the k-Nearest Neighbors. The evaluation is performed based on the classification accuracy on the training and test set.

Goals

- We aim to build a deep learning model which can be used to efficiently classify a Pneumonia infected – and non-infected based on the X-ray scan of the patients. The model is supposed to be an improvement over the baseline which achieves a classification accuracy of around 93%.
- To carry out Data distribution study of how far the train and test sets are from one another.
- To make a comparative study of how better the proposed model is with the baseline
- To make a comparative study of multiple models with each implementing a sub part of the proposed model and evaluating each model's performance

Overall, the final goal is to build a new model which achieves end-to-end learning and efficiently classify the chest X-ray images as Pneumonic/non-Pneumonic.

Motivation and challenges

Pneumonia affects children and families everywhere. It is most prevalent in South Asia and sub-Saharan Africa. And the chest X-rays serve as one of the main sources of detection techniques for Pneumonia even to this day. Doctors primarily recommend for an X-ray scan when person is suffering from symptoms of Pneumonia. X-ray scans do provide a lot of information about the infection. The chest X-rays show decreased lung expansion and patchy opacity on the affected side with ill-defined margins. Hence, it is very important that the X-ray scans are properly processed. The aim is to use the X-ray scans to let the machine predict the presence of Pneumonic infection. It is of very high importance to detect and treat the infection in primary stages to reduce the mortality rates. As it is the primary reason for infant mortality, it is of very high priority to build a model which can detect the infection efficiently, so that the patient starts with the treatment as early as possible.

Related Works

Andrew NG led team worked on CheXNet [7] model to perform Pneumonia Detection where they used a huge 121-layer convolutional neural network that inputs a chest X-ray image and outputs a heatmap localizing the areas of image which indicates pneumonia. The heatmap is an indicator of the likelihood of the presence of the infection. The training set employed is also very large: 112,120 frontal-view X ray images (which contains 14 different disease labels; however the samples are relabeled as Pneumonia (positive examples) and non-Pneumonia (negative examples)) of 30,805 different patients. They achieve an AUROC-score of 76.80% compared to the previous works of 71.3% (by Yao et al.) and 63.3% (by Wang et al.).

Wang et al. [8] worked on the same data set with GoogLeNet, VGGNet-16, ResNet, AlexNet and classify 8 different Thorax diseases (among the existing 14 disease labels in the dataset). A small number of images with infection which have hand labeled bounding boxes are used as ground truths for the localization of the infection. The prediction is again a heatmap indicative of the location of one of the 8 thoracic disease. They achieve an AUCROC of 63.33% for Pneumonia.

Yao et al. [9] work on classifying 14 different thoracic diseases using densely connected convolutional neural

networks on the same data set used by Wang et al. They achieve an AUCROC score of 71.3%.

Jameson Merkow [6] and team work on 3000 chest X-rays with tightly labeled bounding boxes as part of 2018 RSNA Pneumonia challenge. They employ a CoupleNet, a fully convolutional neural network incorporating global and local features for object detection. They report an IoU average C-score (based on averaging over various thresholds per image) of 0.2310.

Stephen [10] and team employed Convolutional Neural Networks to classify the chest X-ray scans of normal and Pneumonic patients to two classes (non-Pneumonia and Pneumonia). The dataset employed contains 5856 X-ray images of anterior-posterior chests. They produce a classification accuracy of 93.73% on the validation set and a 95.31% on training set.

Yu et al. [12] propose multi scale context aggregation by Dilated Convolutions. There has been significant improvement with the usage of the dilated convolutions over the conventional convolutional blocks, which support exponential expansion of the receptive fields without losing resolution.

He et al. [11] propose a technique of spatial pyramid pooling on Deep convolutional networks which generates a fixed length representation regardless of the image size/scale. They produce better accuracy with faster running time than R-CNN.

Dataset Description

The Dataset [5] employed in the project consists of Chest X-Ray images with two categories (Pneumonia and non-Pneumonia) and is available on Kaggle. This dataset was updated by Paul Mooney and is available at https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia. The data set contains Bacterial and Viral tags on the Pneumonia patient scans as well as chest X-ray scans of normal individuals. The dataset is divided to training, validation and test folders each having both positive and negative samples in separate folders.

The images are in jpeg format and the labels for each image has to be generated based on the separate folders maintained for positive and negative examples maintained. The images are of different dimensions in resolution with a depth of 8 bits. A sample visualization of all the three tags can be found in figure 1. The labels for the data are to be generated based on the tag of the X-ray scans. All the chest X-ray scans are of individuals in the **age group 1-5 years old**. All the chest X-rays were initially screened for quality control and the unreadable scans were removed from the dataset. The evaluation set was verified by a **third expert**. There are a total of 5863 X-ray scanned images in the dataset including the training and test samples. The number of "Pneumonia" samples and the number of "Normal" patient samples in the Training and Test set are not well balanced. There are 1341 "Normal" X-ray scan samples while there are 3875 "Pneumonia" X-ray scan samples in Training set. Also, we observe that there are 234 "Normal" X-ray scan samples and 390 "Pneumonia" X-ray scan samples in Test set. Hence, a Data Augmentation technique will be employed to balance the different samples per each class, along with making sure the model sees different samples of the same class.

A very important question when dealing with Machine Learning problems is, if the data set chosen is sufficient. A small justification can be found in the coming subsection.

Is the dataset sufficient? Does it have enough samples for training?

The same dataset is employed by the authors Stephen et al. for the Journal of Healthcare Engineering. They have obtained a state-of-the-art result on the same data set. Also, employing data augmentation can significantly improve the size of the dataset, hence providing sufficient samples for training. To summarize,

Figure 1: Visualization of sample X-ray images

the dataset available and after augmentation, the overall data will be a set of sufficient samples for training, validation and testing purpose. The evaluation is performed on the Train set and Validation + Test set and the values will be reported separately. Also, a data distribution study will be employed using an unsupervised learning KNN, to evaluate the spread and overlap of images in training and test set. This can aid in understanding the dataset and statistical significance of the dataset better. Also, from this study, we will be getting an idea of how diverse the Test set and Train set are, so that we can understand if the model is memorizing or generalizing.

State-of-the-art - Baseline

As discussed in the related work, Stephen and team [10] published a paper in Journal of Healthcare Engineering on employing Convolutional Neural Networks for the classification of the chest X-ray scans of normal and Pneumonic infected individuals. The dataset used by them is same as the dataset employed in the course of our project. This paper is the closest among the other research works to our project and we will be considering this as our baseline. The authors claim to achieve a classification accuracy of 93.73% on the validation data set and a 95.31% on training data set.

The authors employ a simple Convolution Neural Network architecture which they group as two parts: the convolutional filters carrying out feature extraction, and the classifier employing a sigmoid activation. They employ a 3x3 convolution filters in every layer of feature extraction. In each layer of feature extraction ReLU activation, and a max pooling of 2x2 was used. A dropout of 0.5 was used in the final feature extraction process after flattening. Two hidden layers are employed before connecting them to a single node for prediction as Positive (1), that is Pneumonia or Negative (0), that is Normal label.

Figure 2 depicts the overall architecture of the model, the authors employ for the Pneumonia detection. The 2 figure is directly taken from the authors publication for depicting process in the document. The figure 2 gives a fair idea of the Baseline architecture [10]. The baseline model consists of series of Convolutional layers to extract high dimensional filters and a fully connected layer to perform classification. The feature extractor part is basically a series of Convolutional layers: 3 x 3 Convolutional filters with 32, 64, 128, 128 filters in different layers, and intermediate ReLU activations and max pooling layers. The output of the final Convolutional layers is a narrow high dimensional feature representation. This feature is flattened and fed to the fully connected layers for classification. A dropout layer has been employed after flattening before feeding it to the Dense layer.

The details of each layer in the Baseline implementation can be found in the figure 3

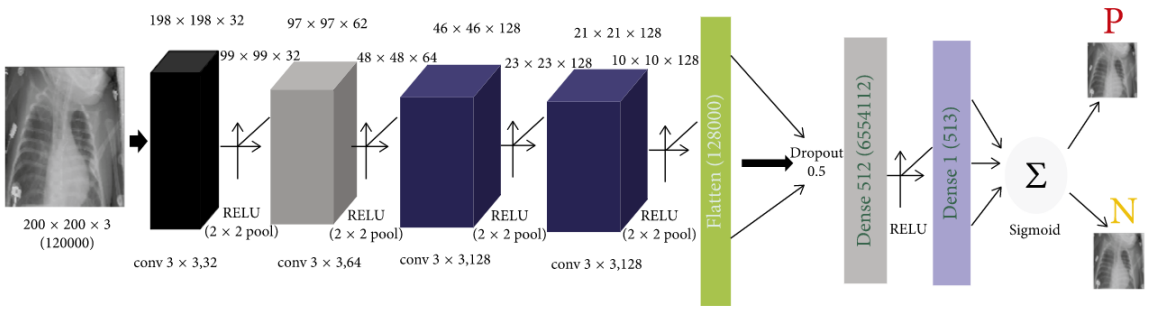

Figure 2: The Baseline architecture

Layer (type)	Output shape	Turtles
conv2d_9 (conv2D)	(None, 198, 198, 32)	896
max_Pooling2d_9 (MaxPooling2)	(None, 99, 99, 32)	0
conv2d_10 (conv2D)	(None, 97, 97, 64)	18496
max_Pooling2d_10 (MaxPooling2)	(None, 48, 48, 64)	0
conv2d_11 (conv2D)	(None, 46, 46, 128)	73856
max_Pooling2d_11 (MaxPooling2)	(None, 23, 23, 128)	0
conv2d_12 (conv2D)	(None, 21, 21, 128)	147584
max_Pooling2d_12 (MaxPooling2)	(None, 10, 10, 128)	0
flatten_3 (Flatten)	(None, 12800)	0
dropout_3 (Dropout)	(None, 12800)	0
dense_5 (Dense)	(None, 512)	6554112
dense_6 (Dense)	(None, 1)	513

Figure 3: Details of the layers in the Baseline architecture

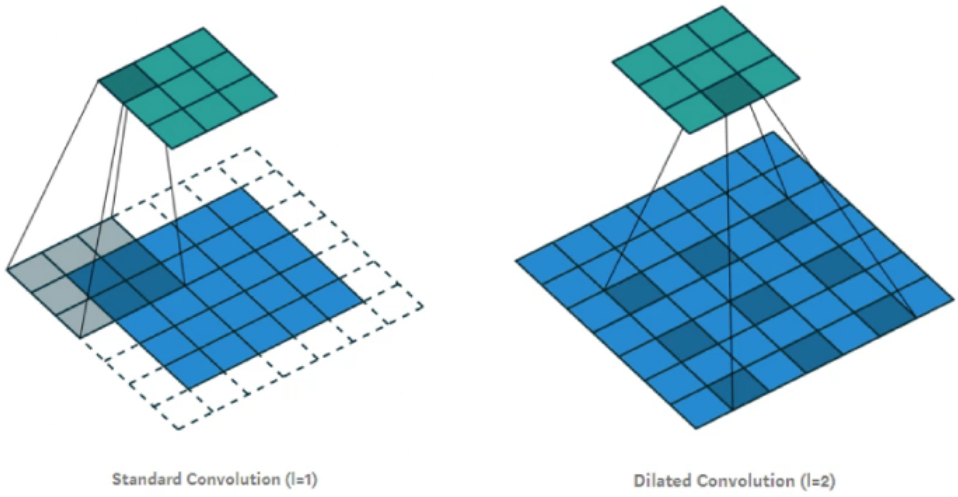

Figure 4: Implementation of Dilated Convolutions (right) compared to Standard Convolutions (left) [1]

Proposed Methods

As discussed in the related works [1], the dilated convolutions show a significant improvement over the conventional convolutional blocks. Also, implementation of the pyramid pooling [2] approach showed significant improvement in the performance.

Inspired by the above two ideas, in each layer of convolutional block we propose the following modifications:

1. Instead of the conventional convolutional filters, dilated convolutions are employed.
2. Different levels of features are extracted in each layer (using various level of dilated convolutions (3 x 3, 5 x 5, 7 x 7).
3. These obtained features are pooled and concatenated, and the high-dimensional, high-information feature blocks are traversed through a long similar chain of network. Finally, a fully connected network is employed to perform classification.

Proposed Model - XNet

The dilated convolutions support exponential expansion of the receptive fields without losing resolution[1]. Figure 4 gives a general idea of how the dilated convolutions are implemented. The enlargement of field of view is similar to including more pixel information during the learning of the kernel weights. This is in line with how humans perceive images with varied field of view. Also, dilated convolutions or "atrous convolutions" comes from the idea of wavelet decomposition. The wavelet decomposition is an integral part of signal and image processing, where wavelets are capable of deconstructing complex signals into basis signals and carry most information with very less loss [4] which will be useful in reconstruction. However, in our case, we use the high information content (with very low loss) as features for classification task.

Also, different level of feature extraction is performed based on the idea of pyramid pooling [2], and these different level features are concatenated as high information features. These features are extracted in multiple non-linear layers, to further use for classification. With the new architecture, the idea is to extract different level features after multiple non-linear transformations. We expect that there will be a significant

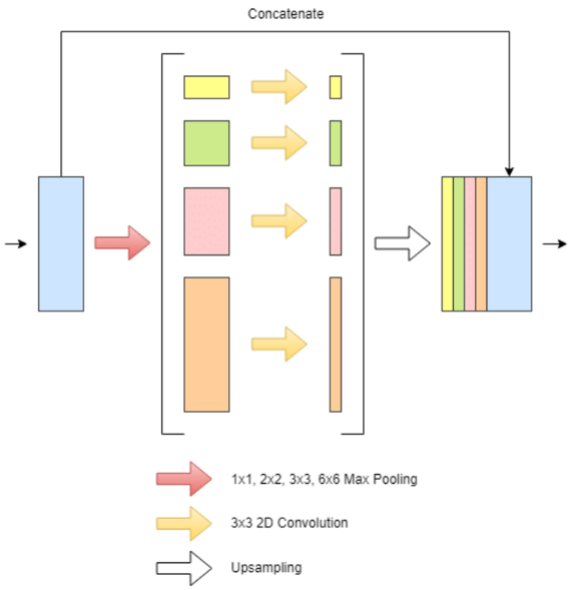

Figure 5: Example of Pyramid pooling [2]

improvement in the performance of the model with the modifications. The basic idea of pyramid pooling is shown in figure 5.

Inspired by these two ideas, we incorporate them in our proposed model. The basic architecture is depicted in the figure 6. We employ an X-net block which contains 4 different dilated convolution blocks of different filter size in parallel, that is, they extract different level of features from the same input. The output of these dilated convolutions are passed through activation, pooled down and concatenated for further processing. The output is batch normalized and similar process is executed till we obtain a narrow block of high dimensional feature. In our case, we start off with an input dimension of 200 x 200 x 3, and end with a narrow block of 12 x 12 x 128. This is flattened for the Classifier and series of fully connected block operates on this to obtain a single node output after Sigmoid activation. This is matched with the label, and the loss function based on the mismatch is back propagated, and all the weights and filter weights are optimized. The number of layers and specific details are as shown in the table 1.

Other details pertaining to X-Net:

- The activations functions ReLU and sigmoid are employed for the classification in Feature extractor part and Classifier part respectively.
- Binary cross entropy loss function and Adam optimizer are employed.
- Learning rates of around 0.00001 and 0.0001 are employed in combination in multiple epochs.
- Batch Normalization layers are also employed after each level of feature extraction.
- A combination of Dropouts are chosen based on Trial and error basis, to simulate noise in each layer.
- The images are resized to 200 x 200 x 3 (based on Baseline and for comparison purpose)
- In general, the features are extracted as narrow high dimensional features before employing the fully connected network.

Layer	**Output shape**	**Turtles**
Image	(None, 200, 200, 3)	0
block1_conv1_pre (Conv2D)	(None, 200, 200, 4) x 4, Dilation rate: 3, Filter size: 3 x 3, 5 x 5, 7 x 7, 9 x 9	112 + 304 + 592 + 976 = 1984
Concatenate	(None, 200, 200, 16)	0
Activation	(None, 200, 200, 16)	0
Batch_Normalization_56	(None, 200, 200, 16)	64
block1_pool_post (MaxPooling2D)	(None, 100, 100, 16)	0
dropout_72 (Dropout)	(None, 100, 100, 16)	0
block2_conv1_pre (Conv2D)	(None, 100, 100, 8) x 4, Dilation rate: 3, Filter size: 3 x 3, 5 x 5, 7 x 7, 9 x 9	1160 + 3208 + 6280 + 10376 = 21024
Concatenate	(None, 100, 100, 32)	0
Activation	(None, 100, 100, 32)	0
Batch_Normalization_57	(None, 100, 100, 32)	128
block2_pool_post (MaxPooling2D)	(None, 50, 50, 32)	0
dropout_73 (Dropout)	(None, 50, 50, 32)	0
block3_conv1_pre (Conv2D)	(None, 50, 50, 16) x 4, Dilation rate: 3, Filter size: 3 x 3, 5 x 5, 7 x 7, 9 x 9	4624 + 12816 + 25104 + 41488 = 84032
Concatenate	(None, 50, 50, 64)	0
Activation	(None, 50, 50, 64)	0
Batch_Normalization_58	(None, 50, 50, 64)	256
block3_pool_post (MaxPooling2D)	(None, 25, 25, 64)	0
dropout_74 (Dropout)	(None, 25, 25, 64)	0
block4_conv1_pre (Conv2D)	(None, 25, 25, 32) x 4, Dilation rate: 3, Filter size: 3 x 3, 5 x 5, 7 x 7, 9 x 9	18464 + 51232 + 100384 + 165920 = 336000
Concatenate	(None, 25, 25, 128)	0
Activation	(None, 25, 25, 128)	0
Batch_Normalization_59	(None, 25, 25, 128)	512
block4_pool_post (MaxPooling2D)	(None, 12, 12, 128)	0
dropout_75 (Dropout)	(None, 12, 12, 128)	0
flatten_9 (Flatten)	(None, 18432)	0
dropout_76 (Dropout)	(None, 18432)	0
dense_36 (Dense)	(None, 256)	4718848
dropout_77 (Dropout)	(None, 256)	0
dense_37 (Dense)	(None, 100)	25700
dropout_78 (Dropout)	(None, 100)	0
dense_38 (Dense)	(None, 25)	2525
dropout_79 (Dropout)	(None, 25)	0
dense_39 (Dense)	(None, 1)	26

Table 1: Details of the Proposed model

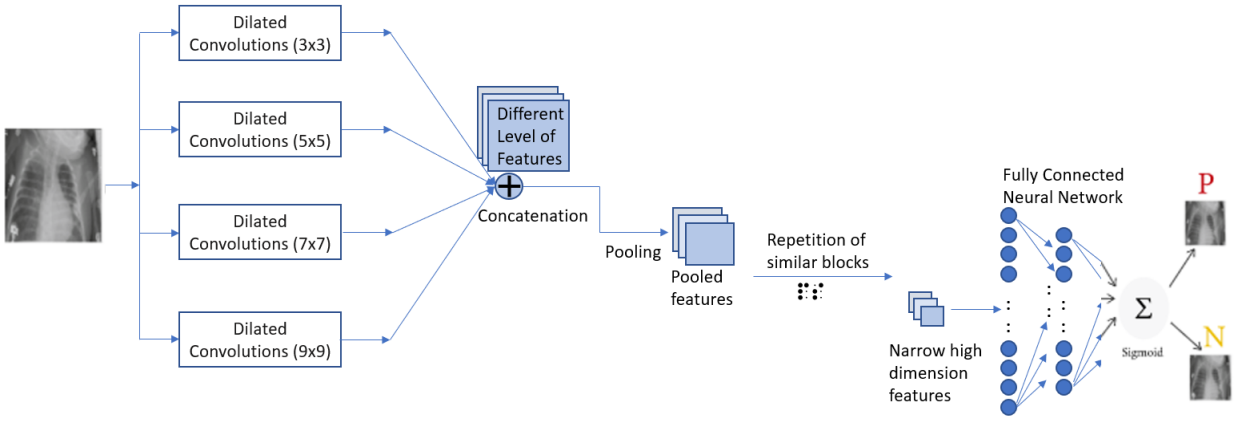

Figure 6: Proposed Architecture

Augmentation Method	Augmentation Value
Rescale	1/255
Rotation Range	40
Shear Range	0.2
Zoom Range	0.2
Width Shift	0.2
Height Shift	0.2
Horizontal flip	TRUE

Table 2: Data Augmentation Details

Data Augmentation

Data augmentation techniques are employed to increase the dataset size and introduce more samples for training as depicted in figure 7. In our project, the data augmentation will be performed based on the Augmentation techniques in the baseline. Data augmentation techniques like rotation, flips, zoom are employed and the augmented data is added to the training samples. The Data Augmentation techniques are also employed to balance the number of training samples per class. Another important problem that needs attention is the size of the overall augmented data. This could pose problems to the resource and could affect the training process. Hence, it is very important to choose an optimal number of augmentations per class to ease the Training process. The Data Augmentations are employed to simulate the uneven transformations found in the X-ray scans like, flip or rotate in the Test set. The Data Augmentations generally tend to increase the Genaralizability and Learning ability of a Machine Learning model and often results in improvement in the performance. However, there is a need to verify this with our dataset, and our experiments and implementations. The results of these experiments are discussed in detail in the later part of the report.

The data augmentation techniques are implemented based on the works of Stephen and team [10]. Data augmentation is implemented to artificially increase the size and quality of the dataset. The process helps in overcoming the problem of overfitting. The augmentation is performed in terms of Rescale, Rotation, Shift, Zoom and Horizontal flips. The details of the augmentation performed by the authors of the baseline journal [10] and in the course of the project to realize Baseline is presented in table 2.

We perform data augmentation such that the previously unequally distributed NORMAL and PNEU-

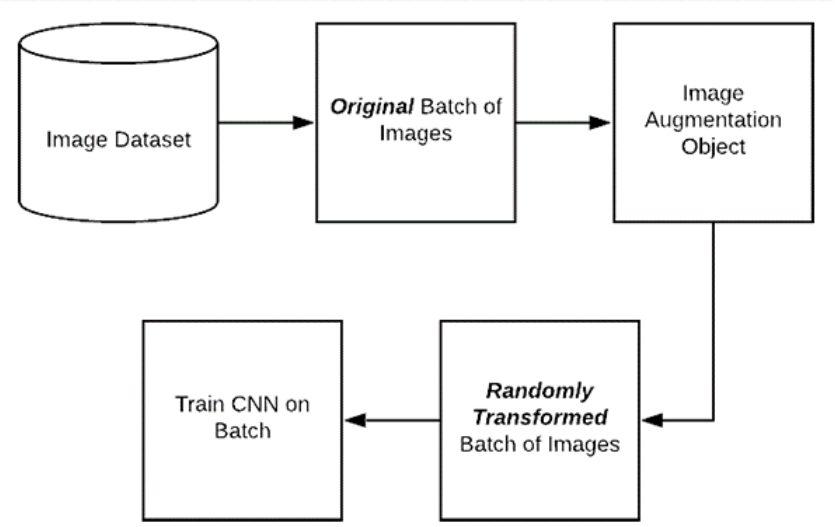

Figure 7: Basic block diagram of Data Augmentation

MONIA cases are now **approximately equal to 10000 images each**. Hence, **overall, there are approximately 20000 samples including the original samples**. Hence, this is **approximately 4-fold increase**, that is approximately, there are three times the original data apart from the original data. With this operation there are sufficient samples to perform further experiments. However, there is still a need of a study of how much data augmentation is sufficient.

Sample Augmentations can be observed from the figures 8 and 9

Novelty of the Term Project

The model employed in the baseline is a simple convolutional neural network architecture. As part of this project, a new architecture is being proposed which modifies the basic convolutional filters block and also implements pyramid pooling after extraction of different level of high dimensional features. Also, the data distribution study is an important part of our project, which indicates the dissimilarity of the test and train images. This can help in generalizing the model's generalizability. Also, we justify the use of Supervised Learning. We verify if Unsupervised Learning approach can be employed to obtain similar or better results. We perform experiments indicating the influence of Data Augmentation on model's performance and justify the use of Data Augmentations too. Finally, the proposed model is experimented with the Augmented data. The details of the novel methods employed are discussed in detail in the coming sections.

Evaluation metrics

The binary cross entropy loss function will be employed and the model learns to reduce the loss in each iteration. Adam optimizer is employed for the optimization. The validation will be performed quantitatively based on the classification accuracy in each of the data subsets, namely, training, validation + testing sets. Accuracy is calculated as 1. We evaluate all our implementations based on the metrics: Accuracy, Precision, Recall, F1 score. The Accuracy is calculated based on the formula 1

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \tag{1}$$

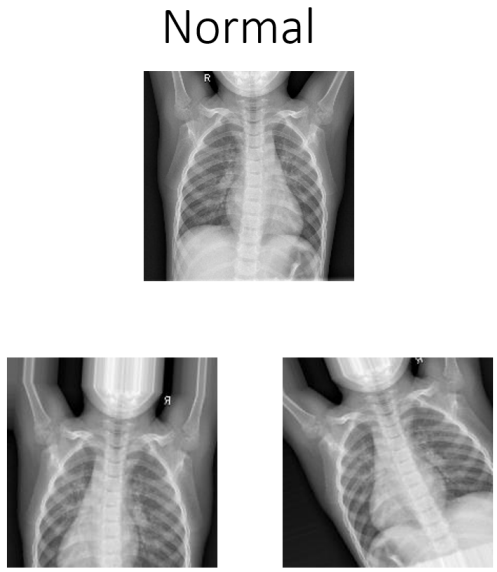

Figure 8: Sample Augmentations on Normal X-ray scans

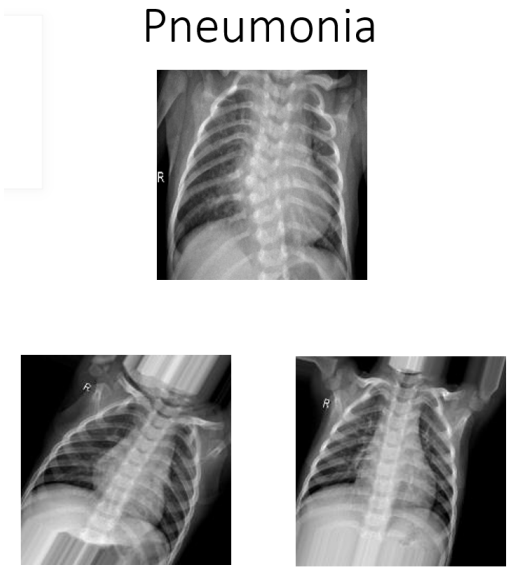

Figure 9: Sample Augmentations on Pneumonia X-ray scans

Also, Confusion matrices are published for each experiment to indicate the mis-classification details, along with the plots.

The precision is a good measure used to determine the effect of False Positive and is effective when the total False Positives are high. Precision gives a measure of how precise/accurate our model is compared to the predicted positives, and gives a measure of how many of them are actual positives. The precision is calculated based on the formula 2.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \tag{2}$$

Recall helps in calculation of how many of the Actual Positives in our model capture the True Positives. The Recall is a metric used to select the best model when there is a high total number of False Negatives.The recall is calculated based on the formula 3.

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} \tag{3}$$

We plot the Precision Vs Recall curves to get an estimate of our model performance in the course of this project.

F1 Score is a metric which balances between the Precision and Recall of the model. The F1 score is basically the harmonic mean of the metrics Precision and Recall. The F1-score is calculated based on the formula 4.

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{4}$$

Experiments

Data Study using KNN

To understand if the training and test data are different is an important part of the study to justify that the Machine Learning model is not overfitting. The model can generalize better if it performs better on the Test data which is quite different from the seen Training data. One of the ways to understand how far is the Test data, is to compute the L2 distance measure (Euclidean distance) between the Test image and the Training images. The L2 distance measure is computed as (5) below.

$$d_E(p, q) = \sqrt{\sum_{i=1}^{N} (p_i - q_i)^2} \tag{5}$$

In our experiment in the course of the project, the least L2 measure is displayed indicating the closest Train image for the corresponding Test image, and how far is the closest Training image helps us understand if the model is seeing new Test data or not. The average distance, that is, a distance measure per pixel is comparatively less, due to the common part of the Test X-ray and Train X-ray images. Each X-ray image has a common black background, and a lot of common rib cage part. Hence, distance per pixel reduces the score and might not give a general idea about the study. The difference between the Train and Test data are the small patches present, which the model needs to learn to generalize. However, in our study, we have presented the results as an average over the total number of pixels, that is, the results presented is an indicator of how far each pixel in the Test data is from the corresponding pixel of the closest Training data in the high dimensional space. This serves as a lower limit. The distance metric is far higher in the high dimensional space, due to the small areas of infection, and large common background and rib cage regions.

To obtain the closest Train image for a corresponding Test image based on the L2 measure, we incorporate the K-Nearest Neighbor algorithm. The algorithm computes the L2 distance measure between the Test image and every Train image, and chooses the lowest distance Train image, indicating the closest Training

image. Hence, this serves as a lower limit. The Test-Train image pair might not seem visually different, but the raw pixel-level difference, can be computed as an L2 distance score. This difference, is what effectively the model needs to generalize, as these are the pixel level intensities, the convolutional model sees and learns. A statistical study on the available L2 distance metric is performed, and average, minimum and maximum over the entire Test data is obtained and presented in the next subsection.

Another important distance metric is the L1 distance measure or the Manhattan distance. This distance metric is particularly important for high dimensional data. In our case, the input to the convolutional models is 200 x 200, and hence, the overall pixels are 40000 per image, which can be classified as high-dimensional raw data. Hence, Manhattan distance is also an important distance measure. A statistical study on the computed L1 distance metric is performed, and average, minimum and maximum over the entire Test data is obtained and presented in the next subsection. Similar to the L2 distance metric, the L1 distance metric is also presented as an average over the total number of pixels, that is, the results presented is an indicator of how different each pixel in Test data is from the corresponding pixel of the closest Training data in the high dimensional space (in terms of pixel intensities). The difference in the pixel intensities, the model learns is much higher than the lower limit, due to the small areas of infection and large areas of background and ribcage.

The L1 distance is calculated as 6

$$d_M(p, q) = \sum_{i=1}^{N} |(p_i - q_i)| \tag{6}$$

The results of the statistical data study on the closest Test and Train image pair is presented in the later part of the report in the subsection .

Unsupervised Learning approach: K-Means

Many a times, for binary classification problem, an unsupervised learning approach of K-Means is sufficient to solve the problem. A K-Means model is employed to understand if the model can classify the data distribution based on the distance between the similar samples. These models turn out to be very effective in various problems. In the course of this project, we experiment with the K-means and publish the results in the coming sections. The Training data is presented as high dimensional vector of size 40000, which is obtained after reshaping the image of shape 200 x 200. A two-class mapping is obtained and two local means of the data is obtained based on the training data. Based on the two classes, training accuracy is obtained. The same approach is performed on the unseen Test data. The Test data is presented and classification is performed based on the obtained mean and the Test data samples. The results of this approach are discussed in the subsection .

Baseline Implementation

The baseline is implemented based on the works of Stephen and team [10] as discussed earlier. The baseline model consists of the feature extractor part and Classifier part. The feature extractor part consists of series of convolutional filters. The classifier part consists of fully connected neural network architecture. Each of the layers were obtained from the output of the previous layer except for the input layer, which takes in the raw X-ray image of size 200 x 200 x 3 and normalized by division of the max value of the pixel intensities (i.e., 255).

The baseline is implemented in two versions in the project. The baseline is tested without Data Augmentation and with Data Augmentation. The results of the baseline implementations are discussed in the later part of the report. The average training and test accuracies of the baseline models are discussed in the subsection and .

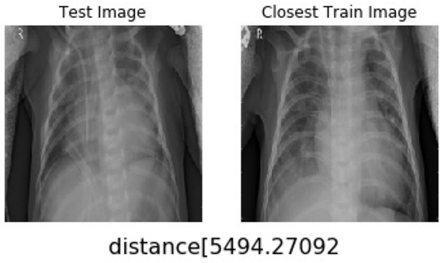

Figure 10: Sample test-train pair

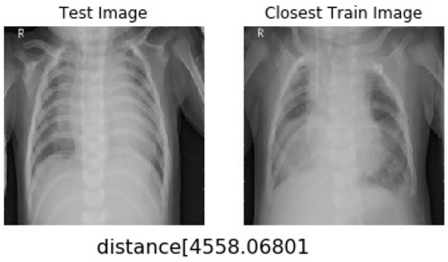

Figure 11: Sample test-train pair

Results and Observations

Data Study using K-Nearest Neighbor

For our experiment, we have concatenated the Validation data and Test data. The images part of these datasets are not seen by the model, and we need the model to generalize better on these image samples. The table 3 presents the results of the statistical study performed on the L2 distance measure of the Test-closest Train image pair over the entire Test data. The table 3 presents the results of the statistical study performed on the L1 distance measure of the Test- closest Train image pair over the entire Test data.

We observe that on an average, each pixel is at least 20.60 units different from the corresponding pixel intensity of the corresponding closest Training image. Based on the minimum statistics, we observe that each pixel is 12.06 units different than the corresponding pixel intensity of the corresponding closest Training image. When we study the distance in the high dimensional space, each pixel is at least 0.1528 units far from the corresponding pixel of the corresponding closest Training image, on an average. Overall, on an average, each Test image is approximately 5812.36 units far from the corresponding closest Train image.

Sample Test image and the closest train image from the Data set can be found in the figures 10 through 12.

Statistical measures (per pixel)	L1 (Manhattan) distance (lower limit)	L2 (Euclidean) distance (lower limit)
Mean	20.60	0.1528
Median	19.93	0.1495
Standard Deviation	3.855	0.034
Minimum	12.06	0.095
Maximum	33.41	0.2534

Table 3: Statistical analysis of the Manhattan (L1) and Euclidean (L2) distance measures per pixel for the Test-Train image pair

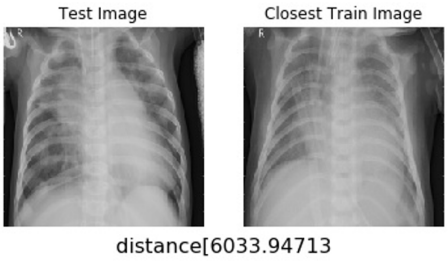

Figure 12: Sample test-train pair

Data set	Accuracy
Train Data set	52.03%
Validation Data set	50%
Test Data set	40.54%

Table 4: Accuracy of Train, Validation and Test Data set for K-means approach

Inference

We can infer from the above data study that the Test Image data set is quite distinct from the Train image data set, when considering the fact that the pixels representing the Pneumonia infection is far less than the background and rib cage pixels. However, the pixels are not too far too, that the model fails to predict on the unseen Test Images. **Based on the Data study, we can conclude that the provided Train and Test splits are good enough to carry out the further experiments.**

Unsupervised Learning approach: K-Means

The K-means model was presented with the high dimensional Training data with two-class constrain. A similar approach on the Test data is also performed and the results are discussed. The table 4 presents the training, validation and test accuracy of the K-means model.

We observe that the test data classification accuracy of 40.54% is achieved in our implementation and a training data classification accuracy of 52.03% is achieved.

Inference

The K-means model performs poorly compared to the published results of the supervised learning model. This might be due to the high-level feature extraction required to perform classification of normal and Pneumonia infected patients. Hence, we can justify the usage of supervised learning models, especially the Convolutional Neural Network models for the task. In the coming sections, let us discuss the results of the Baseline model built using the Convolutional Neural Network architecture. We also examine the effect of data augmentation based on the performance of the baseline model with and without data augmentation.

Baseline Implementation without Data Augmentation

The model was trained with a learning rate of 0.001 and binary crossentropy loss function. Adam optimizer was implemented and training was performed for 50 epochs on NVIDIA RTX 2060 GPU. The table 5 presents the training and test accuracy of the Baseline model.

We observe that the test accuracy mentioned in the paper is not reached in our implementation. The training accuracy, as quoted in the journal [10] is reached, however, the best test accuracy of our baseline model implementation is 83%, while the test accuracy quoted in the journal [10] is 93%. We observe a training accuracy of 96%.

Data set	Accuracy (Given split from Kaggle)	Average accuracy of five different splits	Standard Deviation
Train Data set	96.35%	97.39%	0.06%
Test Data set	84.00%	83.45%	0.14%

Table 5: Accuracy of Train and Test Data set for the Baseline implementation without Data Augmentation

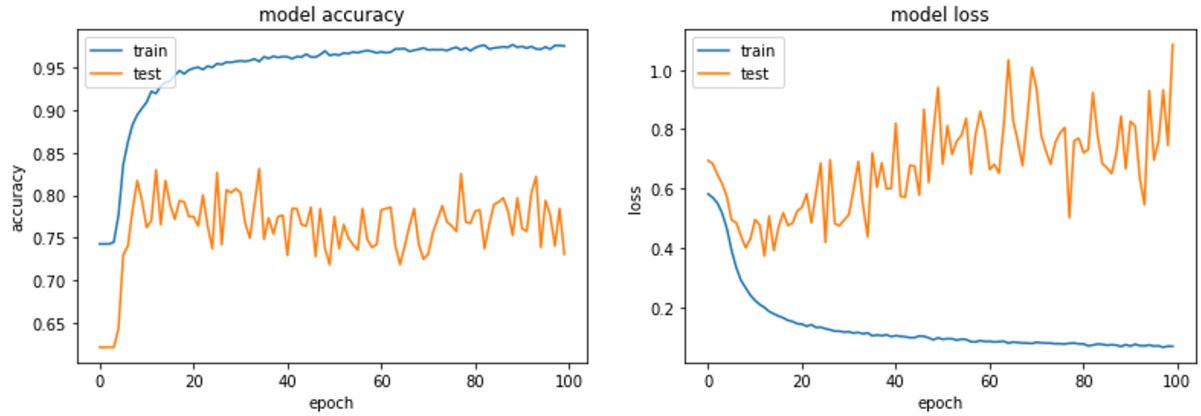

Figure 13: Accuracy Vs epochs and Loss Vs epochs for Baseline implementation without Data Augmentation

We can observe the variation of Training and Test Accuracy with epochs in the figure 13. Another plot can be observed depicting the variation of the Training and Test loss with epochs in the figure 13.

We can also observe the AUC curve and Precision recall curve for this case from the figures 14 and 15. We observe that the area under curve is around 0.83 and shows less precision with more recall. We observe an F1 score F1 score of **86%** for the Baseline model without augmentation.

The Train and Test Confusion matrix for the baseline implementation without Data Augmentation can be found in the tables 6 and 7.

We also visualize the convolution filters in the figure 16. We observe that the deeper filters extract finer features and hence resulting in the higher classification accuracy. The filters are concatenated next to each other for visualization purposes.

We also visualize the output of each convolution layer using the TSNE approach in the figure 17. We observe that as we go deeper, the class separation is more evident. The TSNE visualization of each layer is concatenated next to each other for visualization purposes.

Inference

The model reaches a test accuracy of 83% while the training accuracy of 96% is achieved (due to the fact that the model has not enough samples for it to generalize better) on the given split by the Kaggle. With our five different splits, we observe a mean classification of 97% and standard deviation of 0.06%. Also, the test accuracy of 84% is obtained with a standard deviation of 0.14%.. Also, we observe from the filter visualization

	Class I	Class II
Class I	1221	120
Class II	70	3805

Table 6: Train Confusion Matrix for Baseline Implementation without Data Augmentation

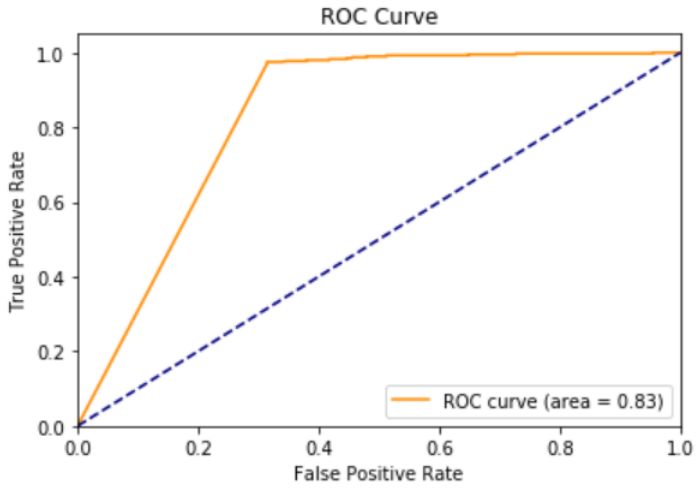

Figure 14: AUC curve for Baseline model without data augmentation

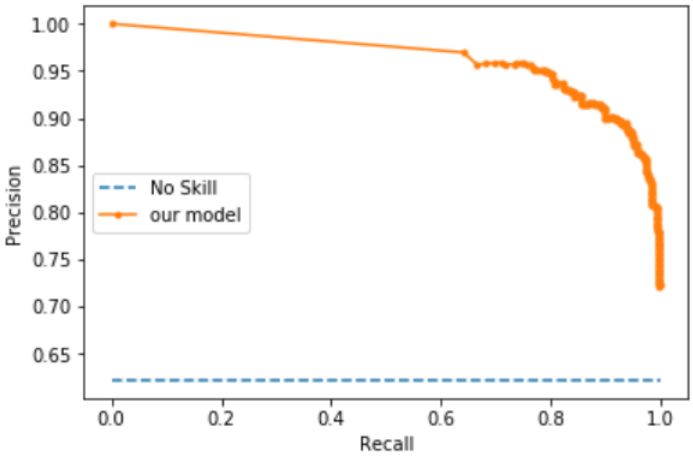

Figure 15: Precision curve for Baseline model without data augmentation

	Class I	Class II
Class I	142	100
Class II	3	395

Table 7: Test Confusion Matrix for Baseline Implementation without Data Augmentation

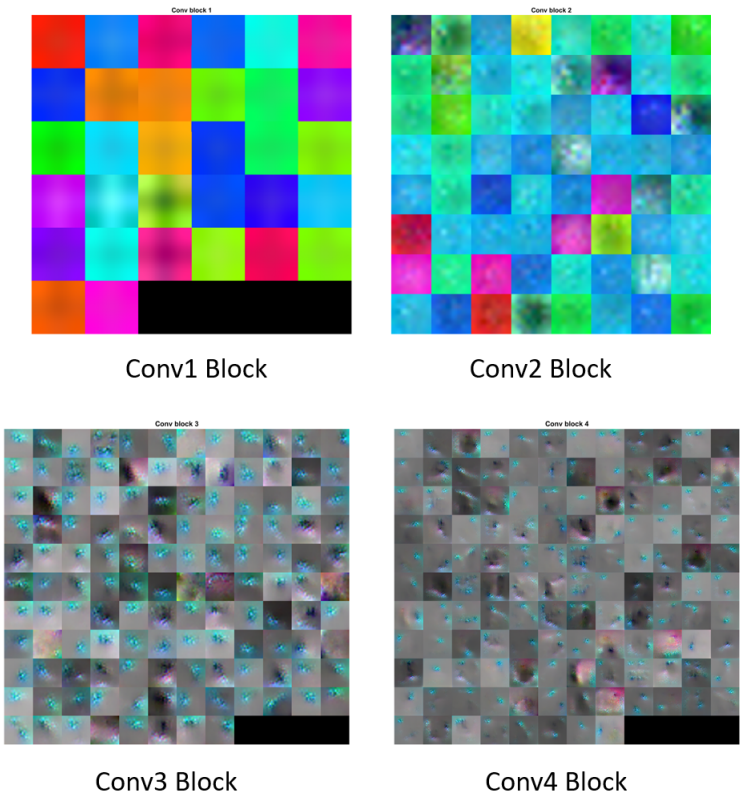

Figure 16: Convolution Filter Visualization for Baseline Implementation without Data Augmentation

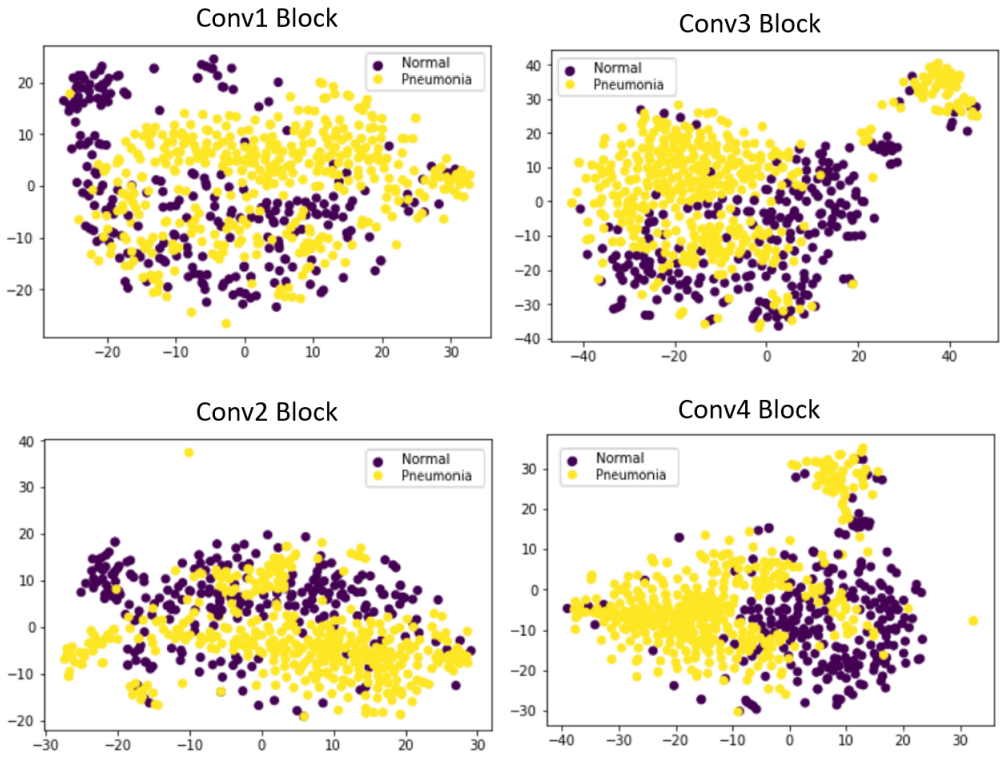

Figure 17: TSNE Visualization for Baseline Implementation without Data Augmentation

Data set	Accuracy (Given split from Kaggle)	Average accuracy of five different splits	Standard Deviation
Train Data set	96.92%	96.40%	0.12%
Test Data set	87.63%	86.14%	0.55%

Table 8: Performance of the Baseline model with Data Augmentation

that the filters extract finer features in the deeper layers and hence resulting in high separation in classes which can be visualized in TSNE plots. The misclassification still persists with this implementation. The Baseline implementation from the paper quotes a 93% accuracy, however, with the same set of parameters quoted in the Baseline paper, the results in our implementation do not reach the Baseline accuracies.

Baseline Implementation with Data Augmentation

The baseline architecture is trained with the augmented data set and tested on the same test data employed. The same test data is used for comparison. The model was trained with a learning rate of 0.0001 and binary cross entropy loss function. Adam optimizer was employed and training was performed for 50 epochs on NVIDIA RTX 2060 GPU. The model reaches a training accuracy of 96.92%, however, a test accuracy of 87.63%. The performance has still not reached as quoted in the Baseline.

The results of the Baseline with Data augmentation is discussed in the table 8

We can observe the variation of Training and Test Accuracy with epochs in the figure 18. Another plot can be observed depicting the variation of the Training and Test loss with epochs in the figure 18.

We can also observe the AUC curve and Precision recall curve for this case from the figures 19 and 20. We observe that the area under curve is around 0.94 and shows less precision with more recall similar to the behaviour of the Baseline model without data augmentation. We observe an improvement in the area under

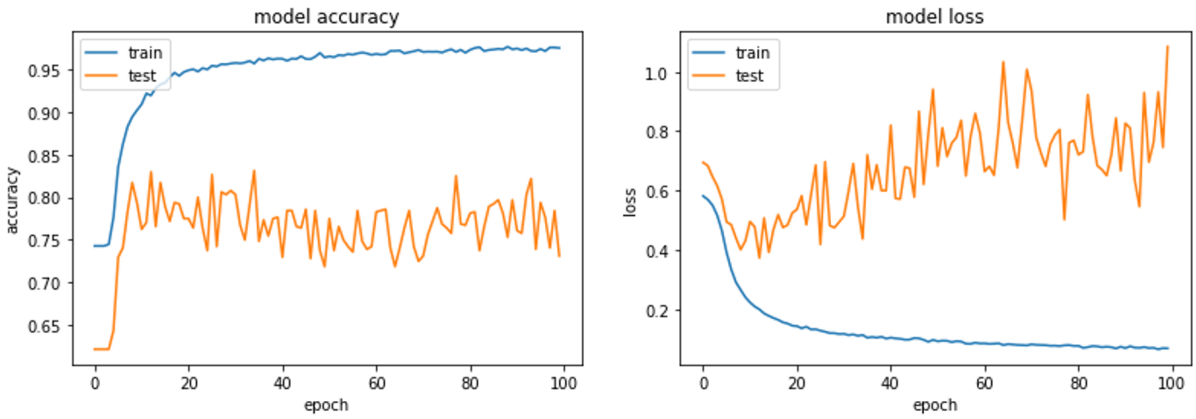

Figure 18: Accuracy Vs epochs and Loss Vs epochs for Baseline implementation with Data Augmentation

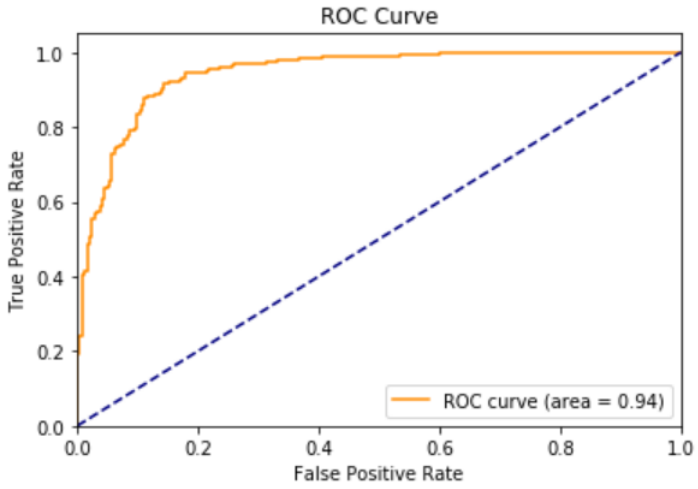

Figure 19: AUC curve for Baseline model with data augmentation

the ROC curve indicating an improvement in the performance. We also observe an improvement in the F1 score over the Baseline model without data augmentation where the F1 score for the Baseline model with data augmentation is **87%**.

The Train and Test Confusion matrix for the baseline implementation with Data Augmentation can be found in the tables 9 and 10.

We visualize the convolution filters in the figure 21. We observe a similar behaviour that the deeper filters extract high level features and hence resulting in a better classification accuracy. The filters are concatenated next to each other for visualization purposes. We observe that the extracted features are much finer compared to the Baseline counterpart without Augmentation.

We also visualize the output of each convolution layer using the TSNE approach in the figure 22. We observe that as we go deeper in the feature extractor, the class separation is more evident. The TSNE visualization of each layer is concatenated next to each other for visualization purposes.

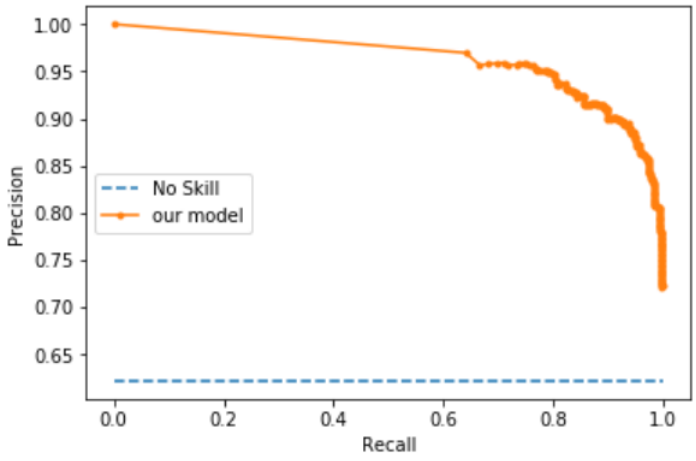

Figure 20: Precision curve for Baseline model with data augmentation

	Class I	Class II
Class I	9617	370
Class II	429	9613

Table 9: Train Confusion Matrix for Baseline Implementation with Data Augmentation

	Class I	Class II
Class I	163	79
Class II	4	394

Table 10: Test Confusion Matrix for Baseline Implementation with Data Augmentation

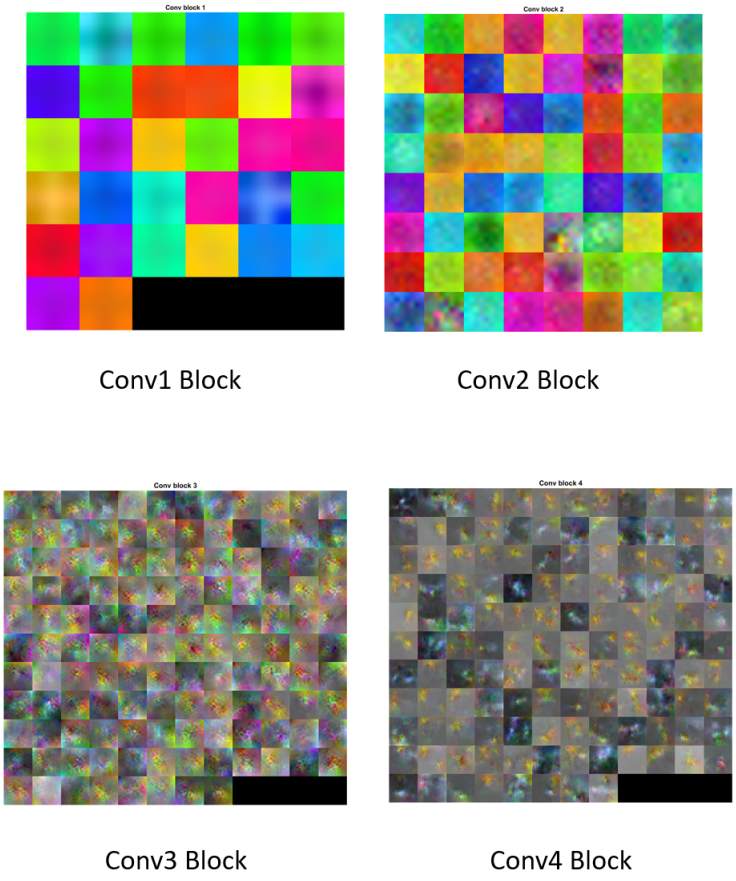

Figure 21: Convolution Filter Visualization for Baseline Implementation with Data Augmentation

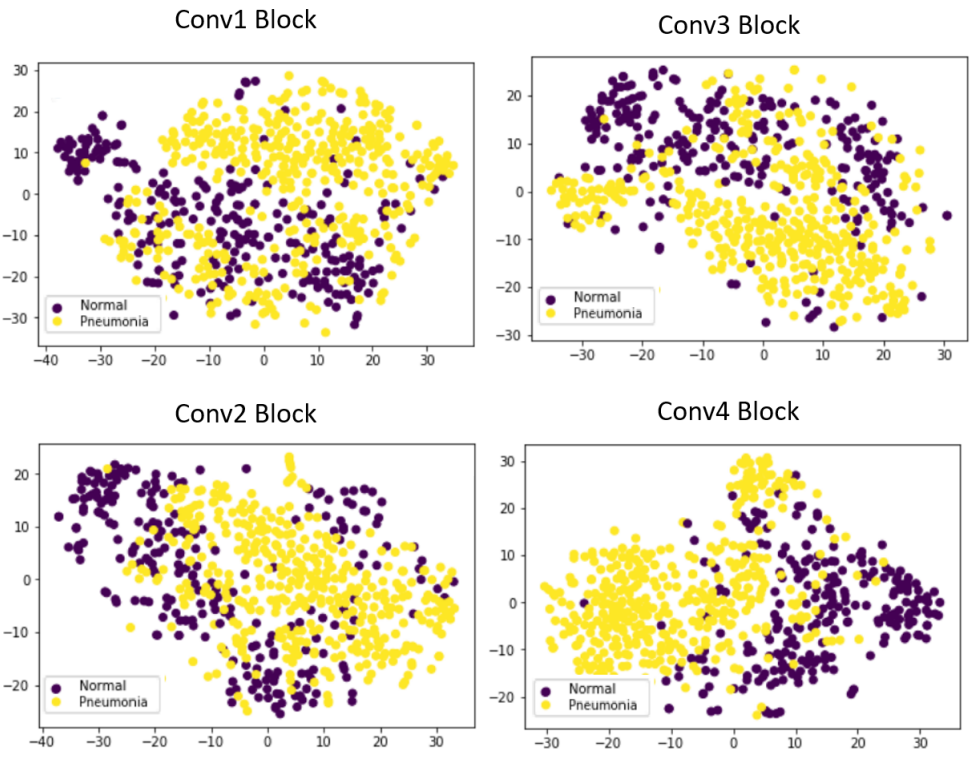

Figure 22: TSNE Visualization for Baseline Implementation with Data Augmentation

Inference

The model reaches a test accuracy of 87% while the training accuracy of 96% on the given split by the Kaggle. With our five different splits, we observe a mean classification of 96% and standard deviation of 0.12%. Also, the test accuracy reaches an average of 86% accuracy score with a standard deviation of 0.55%. Also, we observe from the filter visualization that the filters extract much finer features in the deeper layers compared to the case without data augmentation and hence resulting in high separation in classes which can be visualized in TSNE plots. The misclassification still persists with this implementation. The Baseline implementation from the paper quotes a 93% accuracy, however, with the same set of parameters quoted in the Baseline paper, the results in our implementation do not reach the Baseline accuracies. Contrary to the results of the Baseline model without data augmentation, where the test accuracy reached a peak of 84% and dropped on further training. We observe that the baseline model performance has improved slightly with data augmentation. Hence, we can justify the usage of Data Augmentation.

X-Net Model

The X-Net Model is trained with the augmented data set and tested on the same test data employed for easy comparison. The same test data is used for comparison. The model was trained with a learning rate of 0.0001 and binary cross entropy loss function. Adam optimizer was employed and training was performed for 80 epochs on NVIDIA RTX 2060 GPU. The model reaches a training accuracy of 94.15%, however, a test accuracy of 89.9%.

The results of the X-Net Model is discussed in the table 11.

We can observe the variation of Training and Test Accuracy with epochs in the figure **??**. Another plot can be observed depicting the variation of the Training and Test loss with epochs in the figure 18.

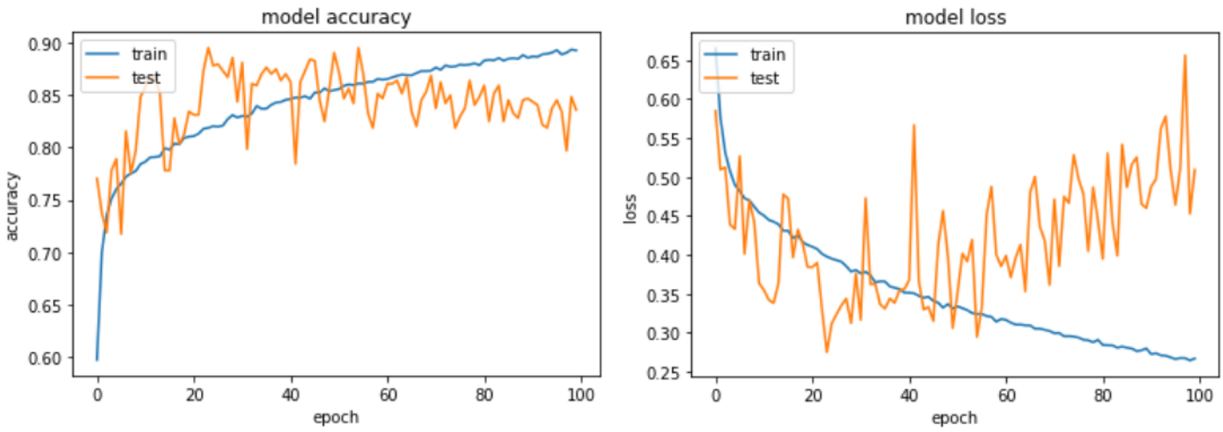

Figure 23: Accuracy Vs epochs and Loss Vs epochs for X-Net Model

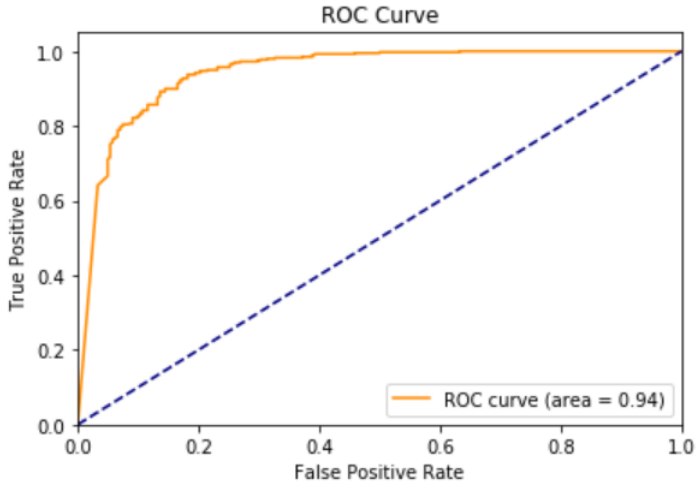

Figure 24: AUC curve for X-Net model

We can also observe the AUC curve and Precision recall curve for this case from the figures 24 and 25. We observe that the area under curve is around 0.94 and shows less precision with more recall similar to the behaviour of the Baseline model with data augmentation. We observe not much improvement in the area under the ROC curve indicating a slight improvement in the performance. However, we observe an improvement in the F1 score where the F1 score for the X-Net model is **88.2%** over the baseline models.

The Train and Test Confusion matrices for the X-Net Model can be found in the tables 12 and 13.

We visualize the convolution filters in the figure 26. We observe that the features extracted are much higher level and have very fine features. This might be a reason for the resulting better classification accuracy. The filters are concatenated next to each other for visualization purposes. The filters of each block are concatenated and presented for better visualization.

We also visualize the output of each X-Net block using the TSNE approach in the figure 27. We observe that, the class separation increases compared to the previous two models, as we go deeper in the feature extractor. The TSNE visualization of each block is concatenated next to each other for visualization purposes.

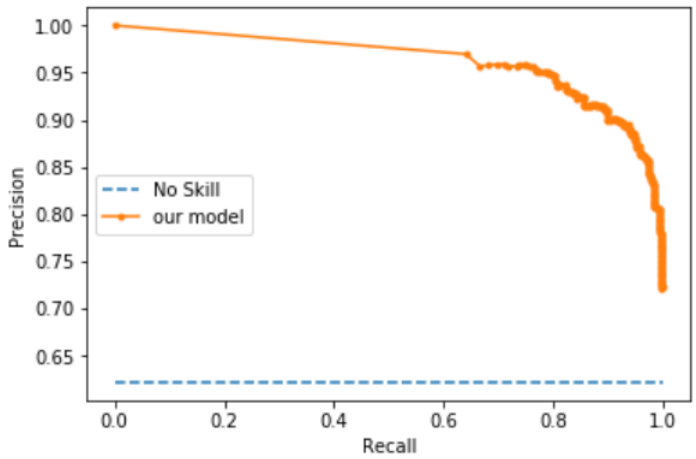

Figure 25: Precision curve for X-Net model

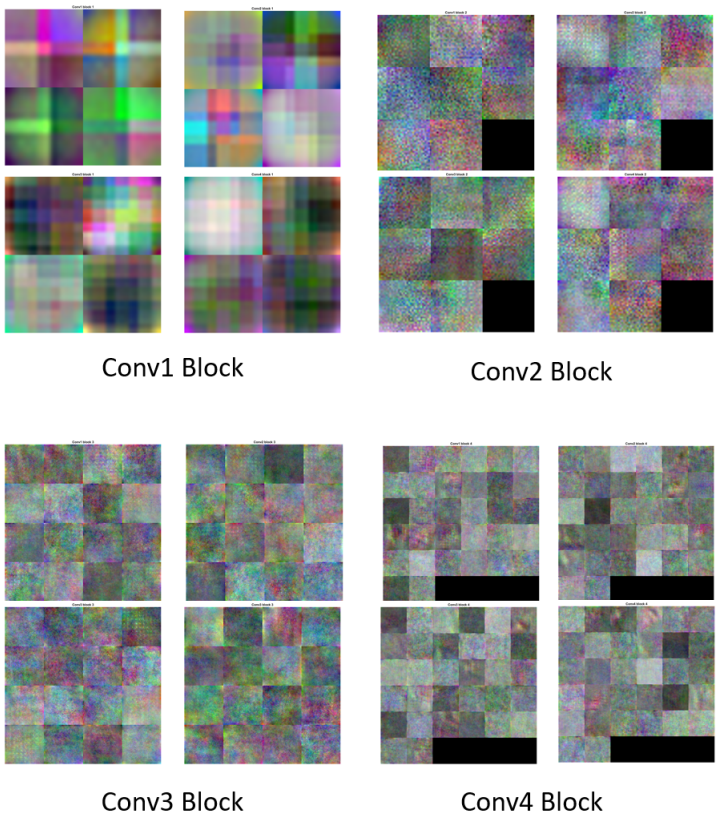

Figure 26: Convolution Filter Visualization for X-Net Model

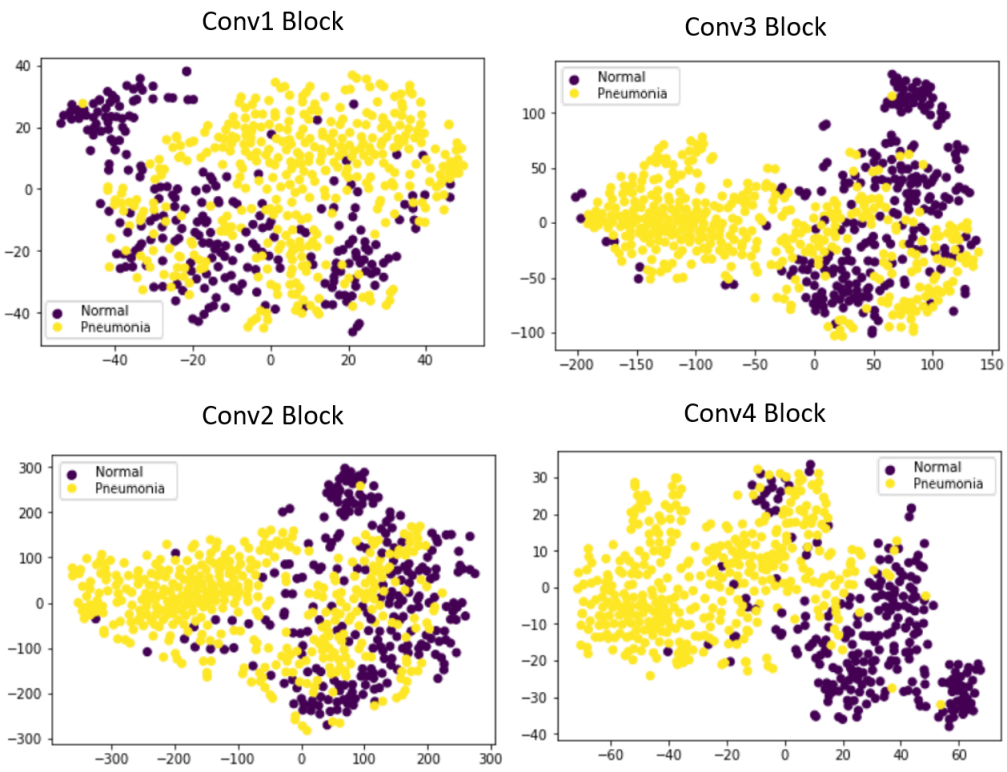

Figure 27: TSNE Visualization for X-Net Model

Data set	**Accuracy (Given split from Kaggle)**	**Average accuracy of five different splits**	**Standard Deviation**
Train Data set	94.15%	95%	0.73%
Test Data set	89.9%	89%	0.46%

Table 11: Performance of the X-Net Model

Inference

The model reaches a test accuracy of 89% while the training accuracy of 94% on the given split by the Kaggle. With our five different splits, we observe a mean classification of 95% and standard deviation of 0.73%. Also, the test accuracy still maintained at 89% accuracy score with a standard deviation of 0.46%. Also, we observe from the filter visualization that the filters extract higher level features in the deeper layers and hence resulting in high class separation between the two classes, which is also evident from the TSNE plots. The approach taken is novel, and with much finer tuning of hyper parameters, the accuracy might further improve.

	Class I	Class II
Class I	9433	554
Class II	618	9424

Table 12: Train Confusion Matrix for X-Net Model

	Class I	Class II
Class I	181	61
Class II	3	395

Table 13: Test Confusion Matrix for X-Net Model

Issues faced during Term Project

- We observe that the model does not achieve the same performance as claimed in the baseline even with the same set of hyper parameters and the same model.
- Due to the large data samples of augmentation, training takes more time and hence, finetuning takes more time.

Future Work

- Further finetuning can be employed to obtain a higher accuracy than presented for the proposed X-Net model
- Including Semi-Supervised learning along with the proposed architecture.
- Including Squeeze-Excitation block and experimenting to check for improvement.

Conclusions

Pneumonia is a serious lung infection, especially found in infants and kids that needs at most care and treatment and the detection using X-rays needs more reach. In this course project, we aimed at building a model that could predict the presence of Pneumonic infection based on the X-ray images. We presented the model with several positive and negative samples of X-ray images of children aged less than 5 years, and the model learnt what features to extract to effectively classify the Pneumonic infected X-rays from the Normal ones. In large scale, these machines help overcome the need of manually checking for the presence of Pneumonic infection. The proposed architecture of X-Net is inspired by two very-strong Machine Learning algorithms and we see that the proposed model achieves a high classification accuracy compared to the Baseline models presented in the course of the term project. Each of the model is also tested with various other splits and find that the standard deviation is not very large indicating that the test set and train set in the given split is widely variant, justifying the results obtained from the data study performed using the K-Nearest Neighbor approach. The study can be carried forward with incorporation of Semi-Supervised Learning and further finetuning.

References

[1] "https://towardsdatascience.com/review-dilated-convolution-semantic-segmentation-9d5a5bd768f5"

[2] "https://www.researchgate.net/figure/An-illustration-of-the-Pyramid-Pooling-Module-PPM_fig2_330700867"

[3] "https://www.who.int/"

[4] "https://www.researchgate.net/post/What_are_the_advantages_of_wavelet_for_filtering_compared_to_conventional_filters"

[5] Kermany, Daniel; Zhang, Kang; Goldbaum, Michael (2018), "Large Dataset of Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images", Mendeley Data, v3 http://dx.doi.org/10.17632/rscbjbr9sj.3

[6] Pneumonia Detection in Chest Radiographs: The DeepRadiology Team

[7] Pranav Rajpurkar, Andrew Y. Ng et al., "CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning"

[8] Wang et al., "ChestX-ray8: Hospital-scale Chest X-ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases"

[9] Yao et al., "LEARNING TO DIAGNOSE FROM SCRATCH BY EXPLOITING DEPENDENCIES AMONG LABELS"

[10] Stephen et al., "An Efficient Deep Learning Approach to Pneumonia Classification in Healthcare"

[11] He et al., "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition"

[12] Yu et al., "Multi-Scale Context Aggregation by Dilated Convolutions"