Udaikaran Singh and Wesley Kwan
Gary Cottrell
CSE 190
11/11/2018

PA3: Deep Convolutional Network for Thorax Disease Detection

Udaikaran Singh
Department of Computer Science
University of California, San Diego.
La Jolla, CA 92093.

Wesley Kwan
Department of Computer Science
University of California, San Diego
La Jolla, CA 92093

Abstract

In this assignment, we used convolutional neural networks through PyTorch to detect diseases from x-ray images. With a dataset of 112,120 images and each image being labeled with no disease, a single disease, or multiple diseases, many factors and parameters were put into play. The primary concern regarding this dataset is the imbalance of images containing no disease and images containing diseases. Because of this, we had to make metrics such as precision and recall, since accuracy became less useful. In addition, we made several models to see which performs best, experimenting with preprocessed images, differing loss functions, and differing network architectures. Our best model came in experiment 7, with an overall accuracy of 94.1%. In this model, the precision was 0.076, the recall was 0.024, and the balanced classification rate was 0.045. We did find better results on the face of it in other experiments, however we found that this is mainly due towards the negative guessing in favor of the negative diagnosis rather than actually learning the patterns of diseases. We demonstrated in experiments 6 and 7 that a weighted loss function can be implemented to learn the internal representations necessary to identify rare disease patterns. This can be seen in how the model is converging towards a near 0 loss and an accuracy of over 98% on the training set. In the prior experiments, we show that without the weighted loss function, the model learns to guess a negative diagnosis, and hovers around a 94.9% accuracy. Therefore, getting a 98% accuracy on the training data shows that the weighted loss function can be used to resolve the class imbalance issue. However, for the sake of time and resources, we trained on a smaller training set on scaled down images. Given more time and resources, we show that the weighted loss function can be used to identify disease patterns in the imbalanced X-ray dataset.

Introduction

The problem this report aims to address is using x-ray imaging to detect and diagnose different diseases. The problem entails recognizing the different patterns that diseases tend to show.

In order to achieve this, we used a convolutional neural network, because of their known ability to perform well on images. The 1024 x 1024 input images in the dataset would only allow for a fully connected neural network that has too many parameters to train well on. The internal representations of such networks are too complicated for humans to understand. Convolutional neural networks, on the other hand, train well on large image inputs and the filters learned by the network can be used to by humans to better understand disease patterns.

We test 3 models initially, along with testing some other changes to aid the model learning from the such as reducing the input image size, normalizing the input images, and changing the objective function. The baseline architecture has 3 stacked convolutional layers, a maxpool layer, and then a 2 layer fully connected neural network. Architecture 1 tests the idea of reducing the numbers of filters within the convolutional layer and adding a fully connected layer. The idea of this is to see whether more parameters in the fully connected layer would aid the model. On the other hand, we added more convolutional layers in architecture 2 and a maxpool to reduce dimensionality within the convolutional layers. The idea behind this architecture is that depth is more important for learning the images, and the maxpool will serve to reduce learning too many parameters.

We initialized the weights in our network with Xavier weight initialization. This method is made to prevent the activation at each neuron in our neural network from exploding or shrinking. The Xavier weight initialization creates the weights based on a Gaussian distribution with a mean of 0 and variance of (1/N) where N is the average between number of neurons that weight connects. The motivation behind this is that we want to the neural network to not be greatly impacted by small changes. By having a Gaussian distribution with mean 0, we force the network to have an activation at each neuron that is close to 0. Therefore, as the network learns from data, it does not have many exploding or shrinking activations. This theoretically helps to smooth the learning curve.

The experiments we performed were to find the relative performance of the different neural architectures to gain a sense of how impactful the depth of the convolutional layers and the size of the fully connected layers is. Also, our experiments aim to see how image pre-processing would impact the effectiveness of the neural network. Lastly, we wanted to see the impact that a weighted loss function would have on the performance of the neural network.

One of the main issues faced when learning from this dataset is that there is a large imbalance between the occurrences of positive and negative outputs. Most people are not diagnosed with a disease, and if they are, they usually have less than half of the diseases. Therefore, based on the loss function, the neural network would learn this imbalance: it will predict a person does not have a disease a majority of the time because it leads to low loss. Even though this does allow the network to have a high accuracy, not much is learned. To remedy this,

we weighted the loss function such that occurrences of false negatives, meaning predictions of no disease when a person does have a disease, would be punished harshly within the loss function. The weights applied was based on finding the percentage of positive and negative labels on a per-class basis. The reason for this is that even though the positive diagnosis is uncommon for all the diseases, it is not equally uncommon.

To measure to results of our neural networks, we tested then on a test set of unseen x-ray images and tracked the statistics: accuracy, precision, recall, and balanced classification rate. The goal of this is to see if we can decrease the occurrences of false negatives (not predicting a disease, when one exists), while not causing a large rise in false positives (predicting a disease exits, when one does not). We also implemented a pseudo-confusion matrix that plots to see which diseases were mistaken for each other. Even though this does not follow the strict mathematical definition of a confusion table, it serves its purpose of allowing us to see what the neural network confuses, providing information that is not available in precision and recall statistics.

Related Works

The motivation for using a convolutional neural network for dealing with the X-ray imaging comes the research by Yann LeCun that demonstrated the effectiveness of convolutional filters in recognizing images of digits. We saw that the pattern recognition of digits as being a similar problem to disease pattern recognition on x-ray images. Therefore, we employed a similar neural network structure of stacked convolutional layers feeding into a fully-connected network. (LeCun et al. 1998)

We were also motivated by the research Xiaosong Wang, who demonstrated the effectiveness of a neural network at solving this problem by adding a sigmoid layer to a pre-trained resNet and GoogLeNet neural networks. This demonstrated that a neural network can be trained to solve this problem. However, we found that the loss function suggested in this paper was not very effective at solving the class imbalance issue present in the dataset. (Xiaosong et. Al 2017).

The primary source of inspiration for solving the class imbalance problem by is a Stanford paper that also studied the same dataset. The paper provided a rudimentary loss function, detailing how to add weights to the loss function to correct for the class imbalance issue. From this, we implemented a weighted loss function that works on the same principles as the one detailed in Rajpurkar's paper. (Rajpurkar et al. 2017).

Methods

(i) Baseline Architecture

Baseline Architecture

Layer (from input to output)	Description of Layer:
Input Layer	The input image is 1024 x 1024 x 1. The image is in greyscale.
Convolutional Layer 1	In-channel = 1 Out-channel = 12 Kernel size = 8 Zero-Padding = 0 Stride = 1 Activation Function = ReLU Batch Normalization Applied
Convolutional Layer 2	In-channel = 12 Out-channel = 10 Kernel size = 8 Zero-Padding = 0 Stride = 1 Activation Function = ReLU Batch Normalization Applied
Convolutional Layer 3	In-channel = 10 Out-channel = 8 Kernel size = 6 Zero-Padding = 0 Stride = 1 Activation Function = ReLU Batch Normalization Applied
Max-Pool Layer	Kernel size = 3 Stride = 3
Fully Connected Layer 1	In-features = 215168 Out-features = 128 Activation Function = ReLU Batch Normalization Applied
Fully Connected Layer 2 (output)	In-features = 128 Out-features = 14 Activation Function = Sigmoid

The loss criterion used is a binomial cross entropy function (without weights to counter the class imbalance issue). The weight parameters were initialized using Xavier weight initialization. The gradient descent optimization used was the Adam optimizer. No regularization was added to the model. We dealt with the class imbalance issue by implementing a weighted loss function that punishes false negatives. The motivation of this is that this motivates the model to learn rare cases, instead of predicting negative all

the time. We implemented cross validation by leaving out 10% of the training set and testing against it to determine whether the model is overtraining.

(ii) Experimental Architecture

Experimental Architecture 1

Layer (from input to output)	Description of Layer:
Input Layer	The input image is 1024 x 1024 x 1. The image is in greyscale.
Convolutional Layer 1	In-channel = 1 Out-channel = 4 Kernel size = 8 Zero-Padding = 0 Stride = 1 Activation Function = ReLU Batch Normalization Applied
Convolutional Layer 2	In-channel = 4 Out-channel = 8 Kernel size = 8 Zero-Padding = 0 Stride = 1 Activation Function = ReLU Batch Normalization Applied
Convolutional Layer 3	In-channel = 8 Out-channel = 12 Kernel size = 6 Zero-Padding = 0 Stride = 1 Activation Function = ReLU Batch Normalization Applied
Max-Pool Layer	Kernel Size = 4 Stride = 4
Fully Connected Layer 1	In-features = 178608 Out-features = 512 Activation Function = ReLU Batch Normalization Applied
Fully Connected Layer 2	In-feature = 512 Out-feature = 128 Activation Function = ReLU Batch Normalization Applied
Fully Connected Layer 3 (output)	In-features = 128 Out-features = 14 Activation Function = Sigmoid

Experimental Architecture 2

Layer (from input to output)	Description of Layer:
Input Layer	The input image is 1024 x 1024 x 1. The image is in greyscale.
Convolutional Layer 1	In-channel = 1 Out-channel = 16 Kernel Size = 8 Zero-Padding = 0 Stride = 1 Activation Function = ReLU Batch Normalization Applied
Convolutional Layer 2	In-channel = 16 Out-channel = 14 Kernel Size = 8 Zero-Padding = 0 Stride = 1 Activation Function = ReLU Batch Normalization Applied
Convolutional Layer 3	In-channel = 14 Out-channel = 12 Kernel Size = 8 Zero-Padding = 0 Stride = 1 Activation Function = ReLU Batch Normalization Applied
Max-Pool Layer	Kernel Size = 3 Stride = 3
Convolutional Layer 4	In-channel = 12 Out-channel = 10 Kernel Size = 6 Zero-Padding = 0 Stride = 1 Activation Function = ReLU Batch Normalization Applied
Convolutional Layer 5	In-channel = 10 Out-channel = 8 Kernel Size = 6 Zero-Padding = 0 Stride = 1 Activation Function = ReLU Batch Normalization Applied
Max-Pool Layer	Kernel Size = 3 Stride = 3
Fully Connected Layer 1	In-feature = 20808 Out-feature = 128 Activation Function = ReLU

	Batch Normalization Applied
Fully Connected Layer 2 (output)	In-feature = 128 Out-feature = 14 Activation Function = sigmoid

For both of these architectures, we used the Adam gradient descent optimizer, no regularization function, and Xavier weight initialization. The class imbalance was addressed by testing a weighted loss function that punishes false negatives heavily on a per-class basis. The idea is that this punishes the model for outputting negative all the time, which will motivate better learning of the rare classes. We implemented cross validation by leaving out 10% of the training set and testing against it to determine whether the model is overtraining.

To address the class imbalance issue, we implemented a loss function that goes weighs false negatives more heavily in order to motivate the network towards learning rare disease patterns.

$$D = Data, y = output\ of\ model$$

$$L(D, y) = -w^{+} * ylog(Y = 1\,|D) - w^{-} * (y - 1)log(Y = 0|D)$$

$$w^{+} = \frac{|N|}{|P| + |N|}$$

$$w^{-} = \frac{|P|}{|P| + |N|}$$

$$|N| = the\ number\ of\ negative\ results\ for\ class\ i$$

$$|P| = the\ number\ of\ positive\ results\ for\ class\ i$$

Note that the loss function is applied to all outputs nodes of the model, which each node having its own $w^{+}$and w^{-} values. This takes into account that the imbalance of negative and positive diagnosis is different for each respective class.

Results

In experiment 1, we tested the baseline architecture described above.

In experiment 2 and 3, we tested architecture 1 and 2 as described above.

In experiment 4, we tested normalizing the inputs before putting them into architecture 2.

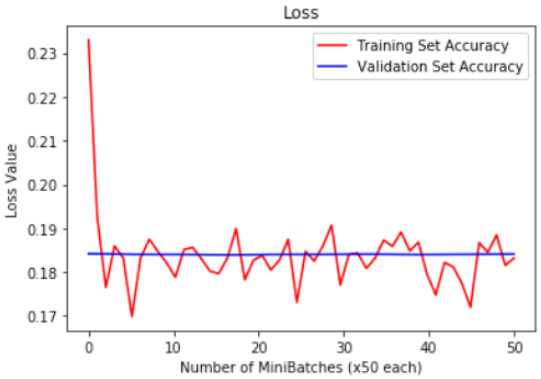
In experiment 5, we tested scaling down the inputs by a half before training on architecture 2. The goal of this was to see if scaling down the images would impact the performance of the model, because theoretically information is lost.

In experiment 6, we tested a weighted loss function with architecture 2.

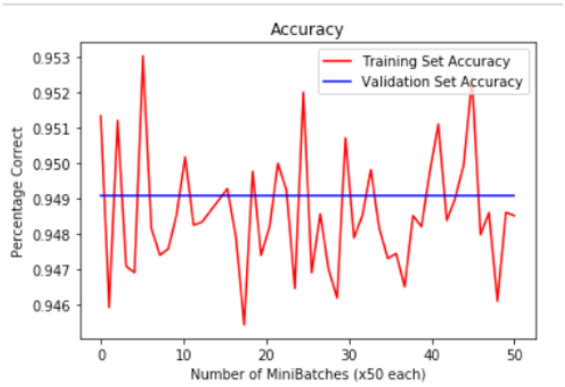
In experiment 7, we tested a weighted loss function, normalized inputs, and scaled down input (to a quarter this time) with architecture 2.

Experiment 1: Baseline Architecture

(i) Loss Curves

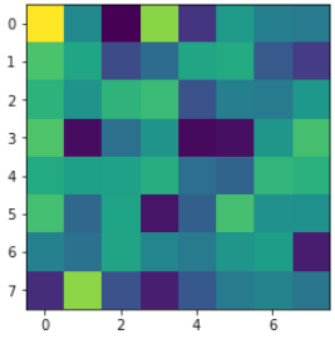


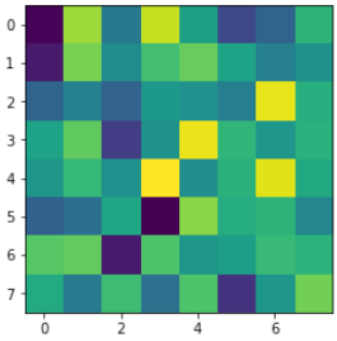
(ii) Accuracy Curves



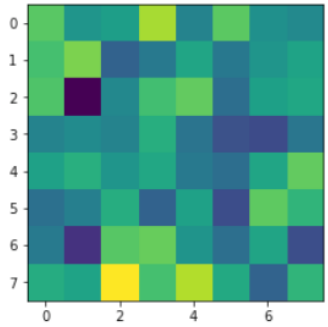
(iii) Visualization of filter maps

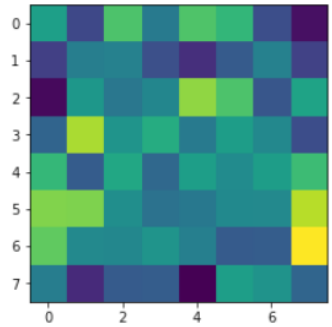
Convolution Layer 1



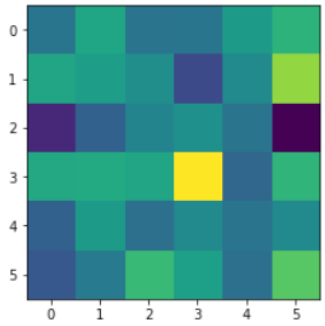


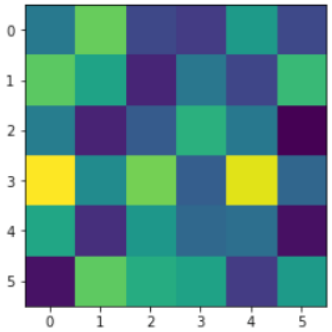
Convolution Layer 2





Convolution Layer 3





(iv) Model Results

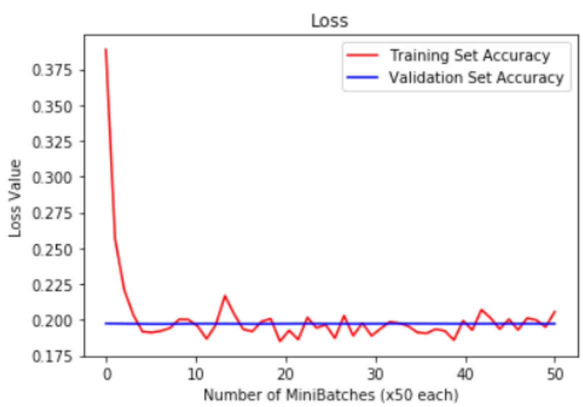
Disease	Accuracy	Precision	Recall	BCR
Atelectasis	0.896	0.5	0.0	0.25
Cardiomegaly	0.976	0.5	0.002	0.251
Effusion	0.881	0.5	0.0	0.25
Infiltration	0.82	0.667	0.001	0.334
Mass	0.95	0.5	0.001	0.25
Nodule	0.941	0.5	0.001	0.25
Pneumonia	0.986	0.5	0.004	0.252
Pneumothorax	0.953	0.333	0.001	0.167
Consolidation	0.958	0.5	0.001	0.251
Edema	0.979	0.5	0.002	0.251
Emphysema	0.977	0.5	0.002	0.251
Fibrosis	0.986	0.5	0.003	0.252
Pleural Thickening	0.97	0.5	0.002	0.251
Hernia	0.251	0.5	0.027	0.264
Model Average Performance	0.948	0.5	0.003	0.252

Confusion Matrix:

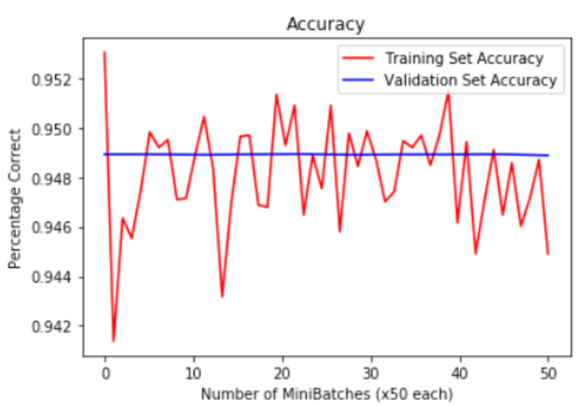
```
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '1.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '1.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
```

Experiment 2: Architecture 1 (w/o addressing class imbalance)

(i) Loss Curves

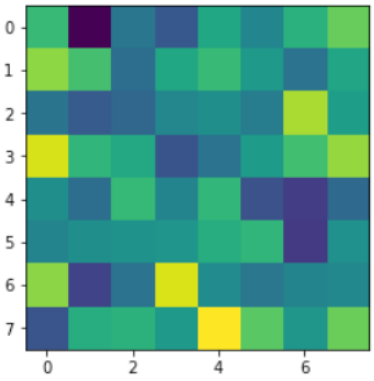


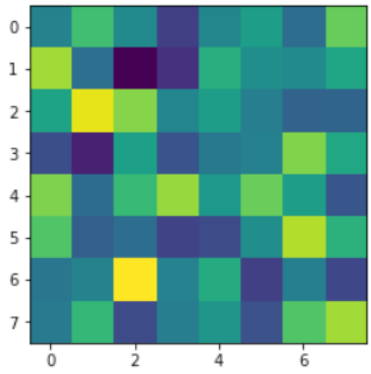
(ii) Accuracy Curves



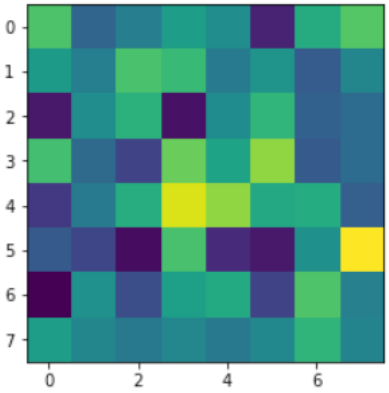
(iii) Visualization of filter maps

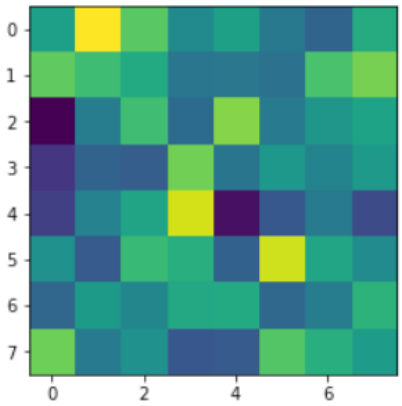
Convolution Layer 1



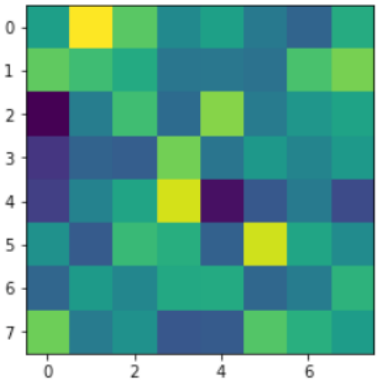


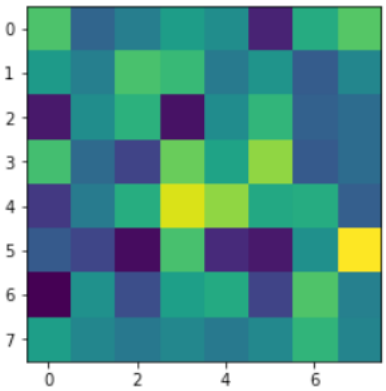
Convolution Layer 2





Convolution Layer 3





(iv) Model Results

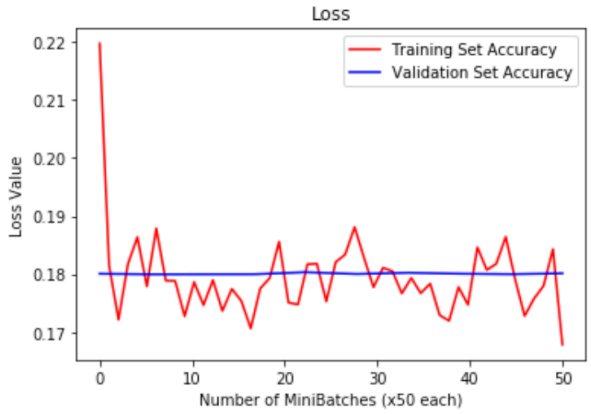
Disease	Accuracy	Precision	Recall	BCR
Atelectasis	0.897	0	0.0	0.0
Cardiomegaly	0.975	0	0	0
Effusion	0.881	0.25	0.0	0.125
Infiltration	0.826	0	0	0
Mass	0.951	0.5	0.004	0.252
Nodule	0.946	0	0	0
Pneumonia	0.988	0	0	0
Pneumothorax	0.954	0	0	0
Consolidation	0.957	0	0	0
Edema	0.98	0	0	0
Emphysema	0.979	0	0	0
Fibrosis	0.984	0	0	0
Pleural Thickening	0.97	0	0	0
Hernia	0.998	0	0	0
Model Average Performance	0.949	0.054	0.0	0.027

Confusion Matrix:

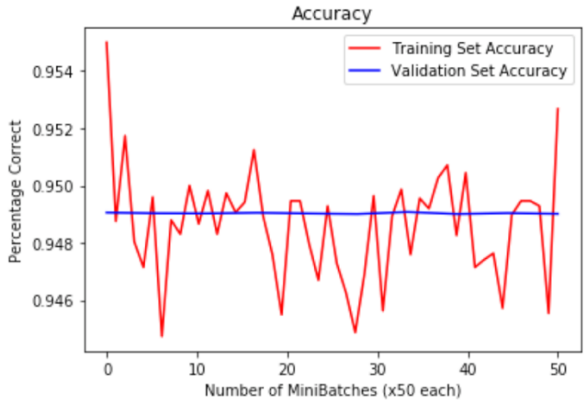
```
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '1.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '1.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
```

Experiment 3: Architecture 2 (w/o addressing class imbalance)

(i) Loss Curves

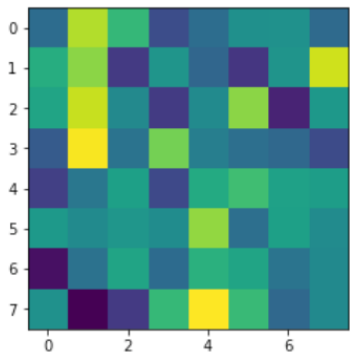


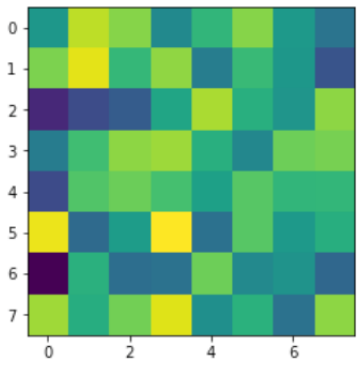
(ii) Accuracy Curves



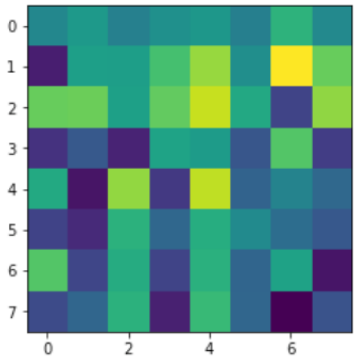
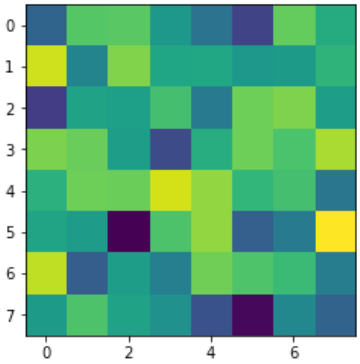
(iii) Visualization of filter maps

Convolution Layer 1

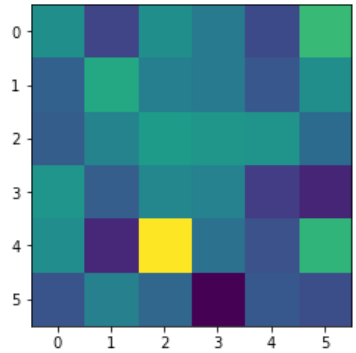
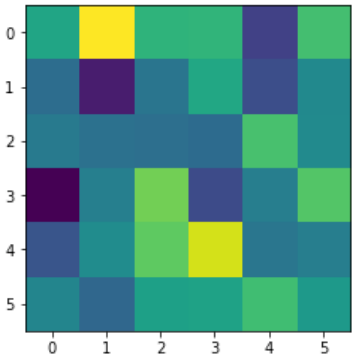




Convolution Layer 3

Convolution Layer 5

(iv) Model Results

Disease	Accuracy	Precision	Recall	BCR
Atelectasis	0.89559	0	0	0
Cardiomegaly	0.97235	0	0	0
Effusion	0.87924	1.0	0.00041	0.50021
Infiltration	0.82236	0.30172	0.00989	0.15581
Mass	0.94951	0.2	0.00098	0.10049
Nodule	0.93984	0	0	0
Pneumonia	0.98731	0	0	0
Pneumothorax	0.95253	0	0	0
Consolidation	0.95887	0	0	0
Edema	0.97968	0	0	0
Emphysema	0.97968	0	0	0
Fibrosis	0.98419	0	0	0
Pleural Thickening	0.97002	0	0	0
Hernia	0.99767	0	0	0
Model Average Performance	0.948	0.107	0.001	0.054

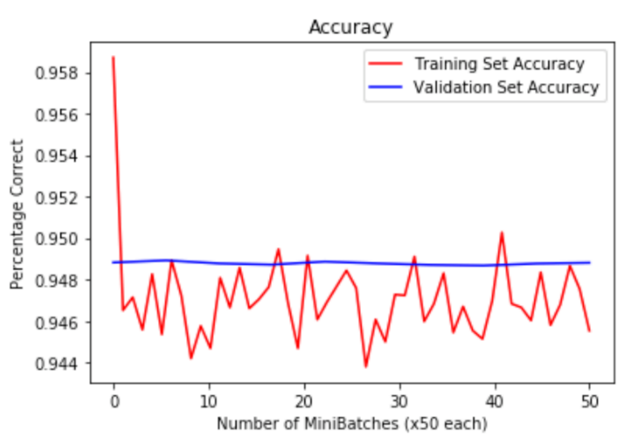
Confusion Matrix:

```
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '1.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '1.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '1.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
```

Experiment 4: Architecture 2 with Normalized Inputs

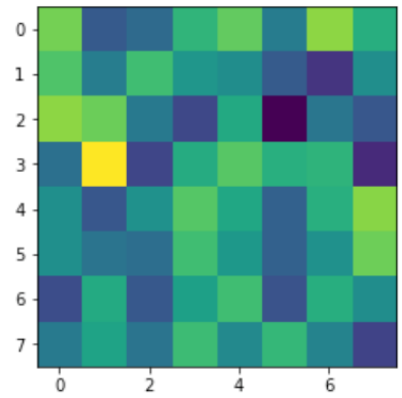
(i) Loss Curves

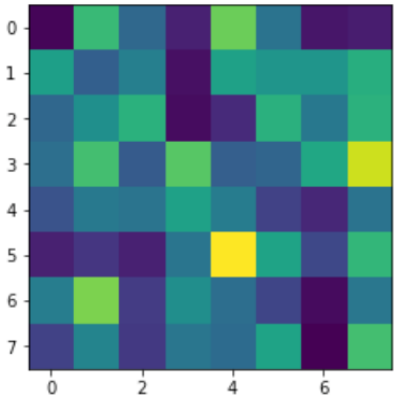
(ii) Accuracy Curves



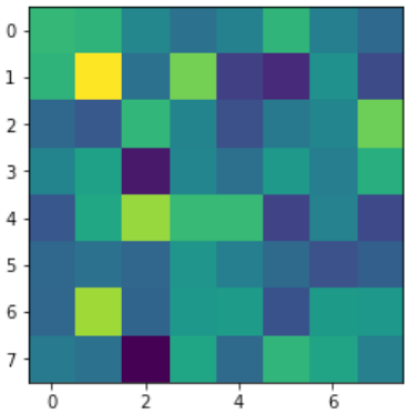
(iii) Visualization of filter maps

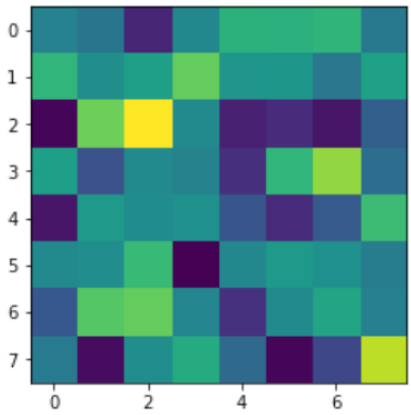
Convolution Layer 1



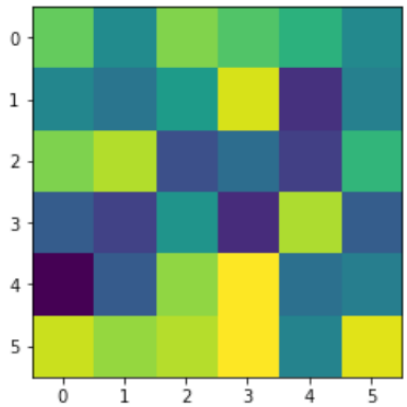


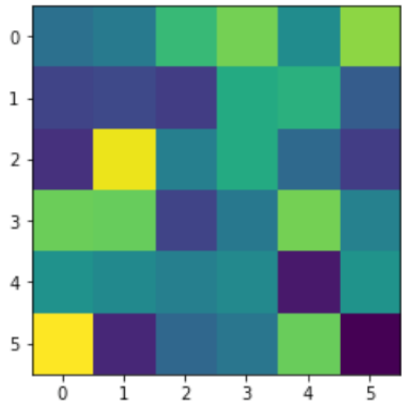
Convolution Layer 3





Convolution Layer 5





(iv) Model Results

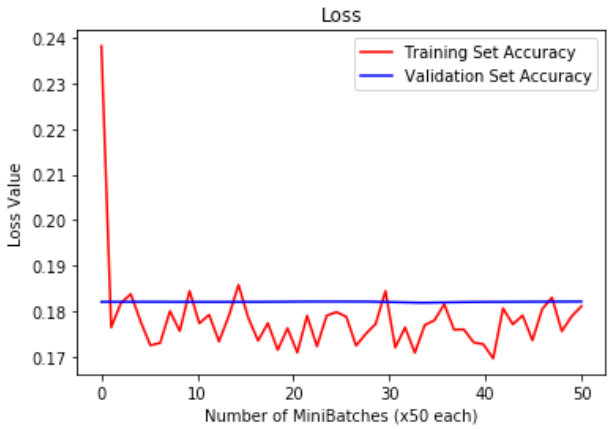
Disease	Accuracy	Precision	Recall	BCR
Atelectasis	0.881	0.232	0.057	0.144
Cardiomegaly	0.977	0	0	0
Effusion	0.881	0.727	0.003	0.365
Infiltration	0.82	0.364	0.047	0.205
Mass	0.947	0	0	0
Nodule	0.944	0	0	0
Pneumonia	0.989	0	0	0
Pneumothorax	0.953	0	0	0
Consolidation	0.958	0	0	0
Edema	0.979	0	0	0
Emphysema	0.978	0	0	0
Fibrosis	0.986	0	0	0
Pleural Thickening	0.97	0	0	0
Hernia	0.998	0	0	0
Model Average Performance	0.947	0.094	0.008	0.051

Confusion Matrix:

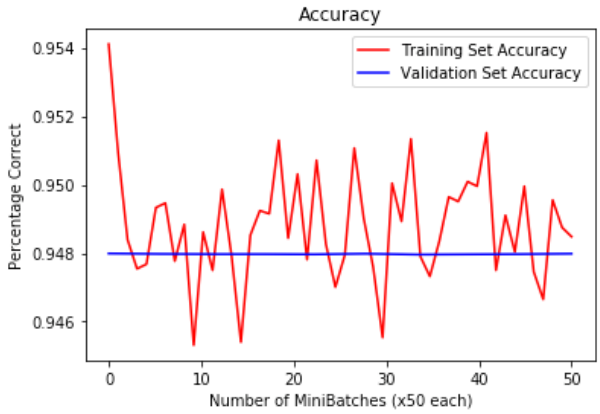
```
['1.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.05', '0.00', '0.95', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '1.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
```

Experiment 5: Architecture 2 with Scaled Down Image (by ½)

(i) Loss Curves

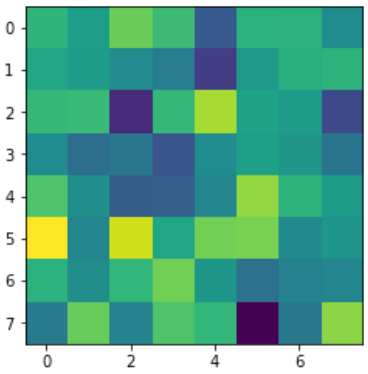


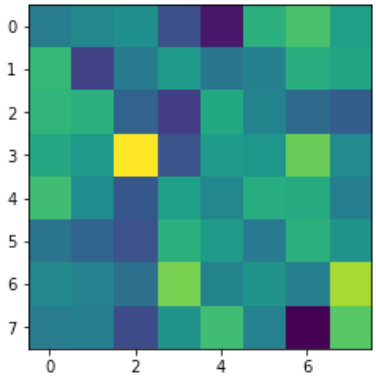
(ii) Accuracy Curves



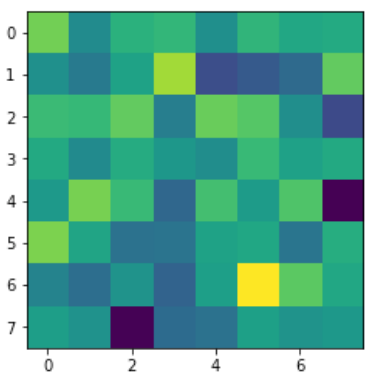
(iii) Visualization of filter maps

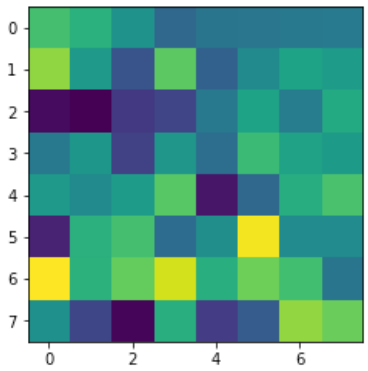
Convolution Layer 1



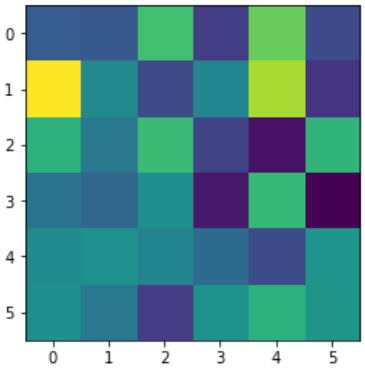


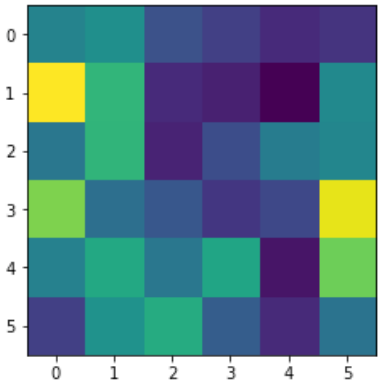
Convolution Layer 3





Convolution Layer 5





(iv) Model Results

Disease	Accuracy	Precision	Recall	BCR
Atelectasis	0.897	0	0	0
Cardiomegaly	0.976	0	0	0
Effusion	0.883	1.0	0.001	0.501
Infiltration	0.819	0	0	0
Mass	0.948	0	0	0
Nodule	0.944	0	0	0
Pneumonia	0.988	0	0	0
Pneumothorax	0.955	0	0	0
Consolidation	0.958	0	0	0
Edema	0.978	0	0	0
Emphysema	0.977	0	0	0
Fibrosis	0.985	0	0	0
Pleural Thickening	0.971	0	0	0

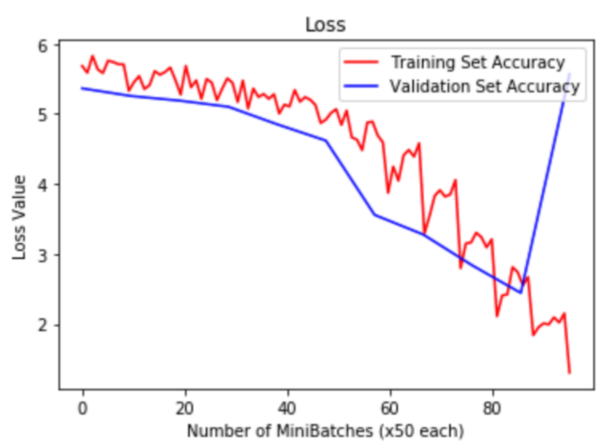
Hernia	0.998	0	0	0
Model Average Performance	0.948	0.071	0	0.036

Confusion Matrix:

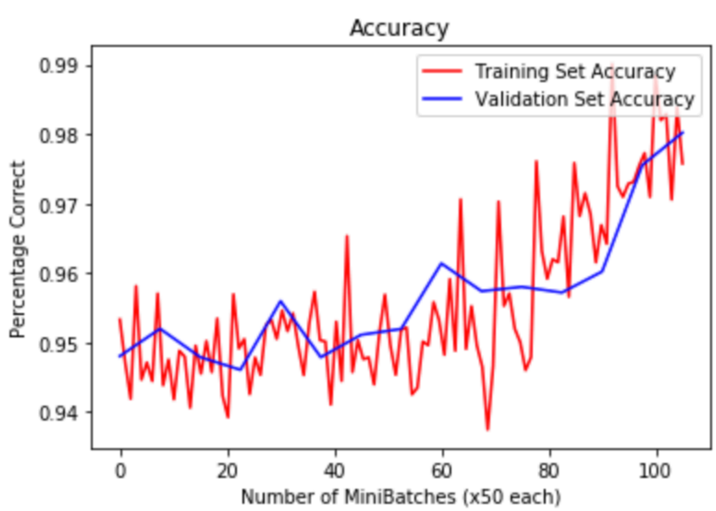
```
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '1.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
```

<u>Experiment 6: Architecture 1 with weighted objective function</u>

(i) Loss Curves

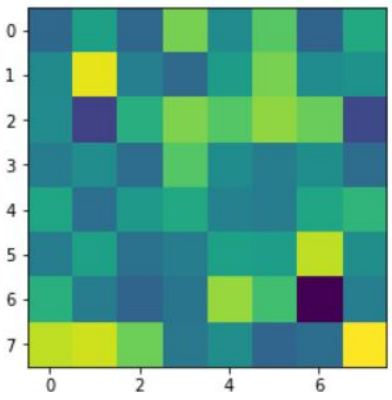


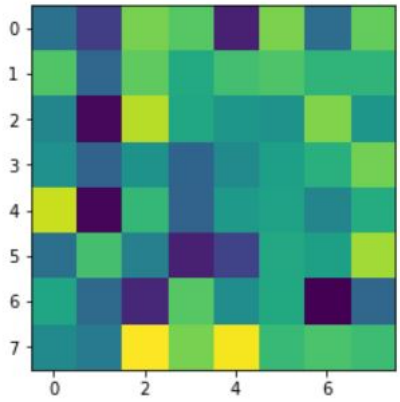
(ii) Accuracy Curves



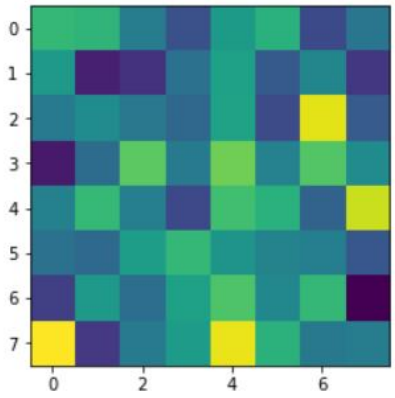
(iii) Visualization of filter maps

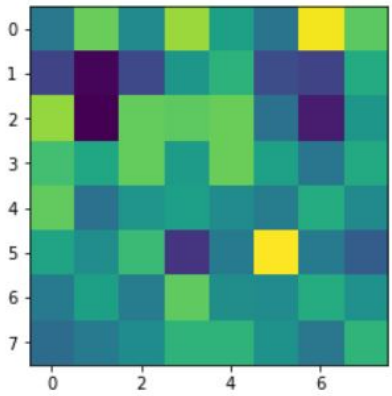
Convolution Layer 1



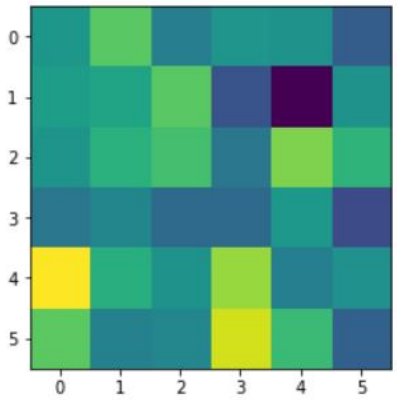


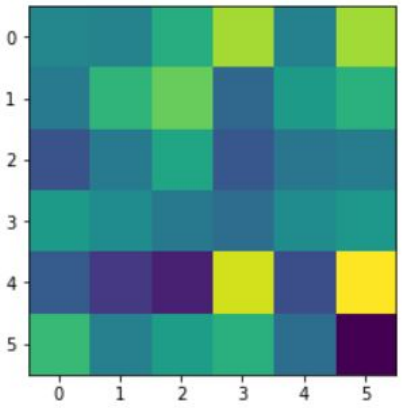
Convolution Layer 3





Convolution Layer 5





(iv) Model Results

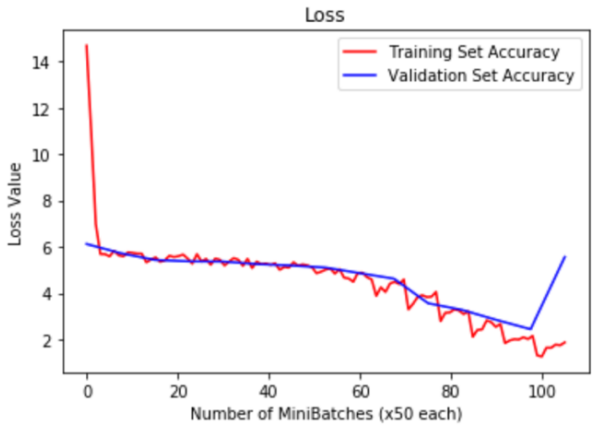
Disease	Accuracy	Precision	Recall	BCR
Atelectasis	0.879	0.232	0.054	0.143
Cardiomegaly	0.972	0.054	0.007	0.031
Effusion	0.845	0.239	0.153	0.196
Infiltration	0.785	0.223	0.088	0.156
Mass	0.948	0.255	0.022	0.138
Nodule	0.942	0.048	0.005	0.027
Pneumonia	0.989	0	0	0
Pneumothorax	0.949	0.044	0.004	0.024
Consolidation	0.954	0.009	0.002	0.006
Edema	0.979	0	0	0
Emphysema	0.978	0	0	0
Fibrosis	0.986	0	0	0
Pleural Thickening	0.966	0.023	0.003	0.013
Hernia	0.998	0	0	0
Model Average Performance	0.941	0.076	0.024	0.05

Confusion Matrix:

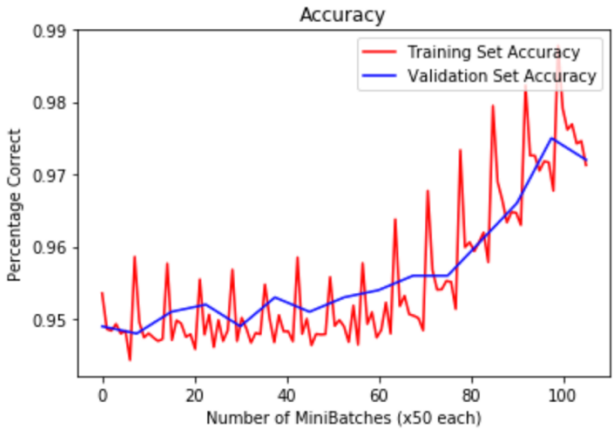
```
['0.83', '0.10', '0.01', '0.05', '0.01', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.11', '0.80', '0.08', '0.01', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.02', '0.11', '0.86', '0.01', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.04', '0.00', '0.10', '0.86', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.13', '0.11', '0.01', '0.87', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.02', '0.01', '0.05', '0.92', '0.02', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.03', '0.01', '0.35', '0.01', '0.00', '0.60', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.01', '0.01', '0.03', '0.12', '0.02', '0.01', '0.80', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.01', '0.04', '0.00', '0.95', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.10', '0.04', '0.00', '0.01', '0.00', '0.00', '0.00', '0.00', '0.00', '0.80', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.07', '0.00', '0.00', '0.00', '0.00', '0.00', '0.08', '0.85', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
```

Experiment 7: Architecture 2 with weighted objective function

(i) Loss Curves

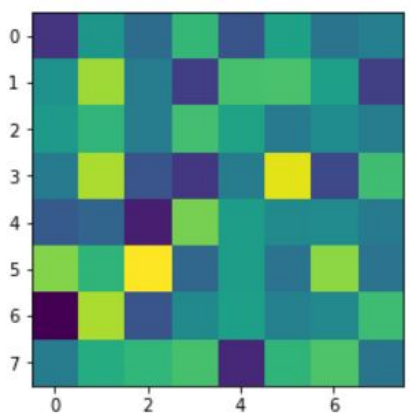


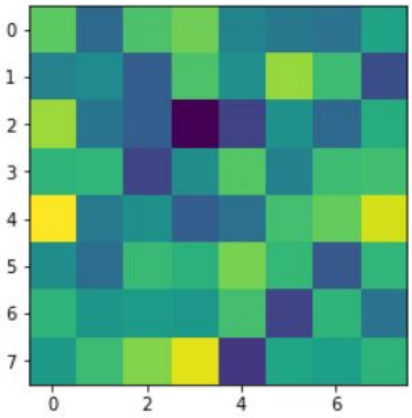
(ii) Accuracy Curves



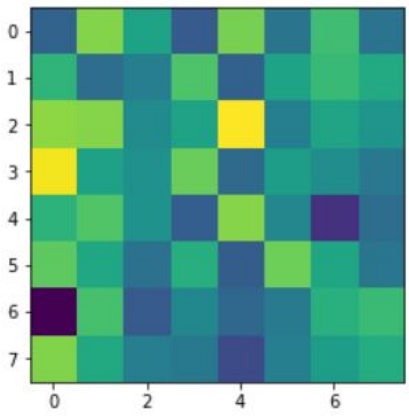
(iii) Visualization of filter maps

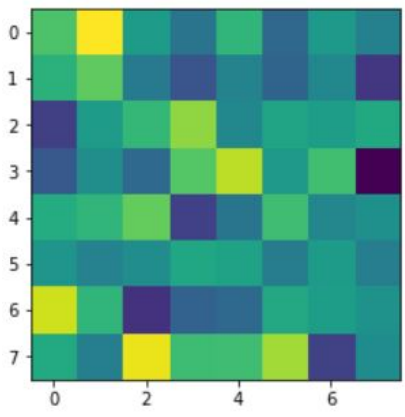
Convolution Layer 1



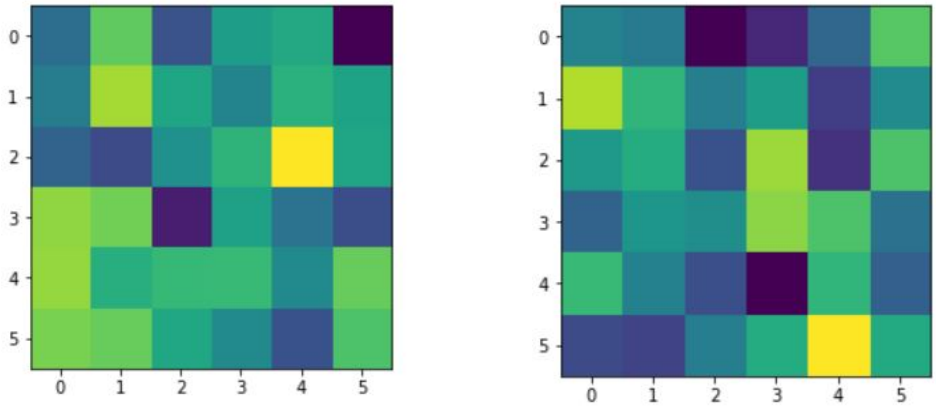


Convolution Layer 3





Convolution Layer 5



(iv) Model Results

Disease	Accuracy	Precision	Recall	BCR
Atelectasis	0.879	0.172	0.054	0.113
Cardiomegaly	0.972	0.108	0.014	0.061
Effusion	0.845	0.239	0.153	0.196
Infiltration	0.785	0.223	0.088	0.156
Mass	0.947	0	0	0
Nodule	0.942	0.048	0.005	0.027
Pneumonia	0.988	0	0	0
Pneumothorax	0.952	0.044	0.004	0.024
Consolidation	0.954	0.055	0.014	0.035
Edema	0.982	0	0	0
Emphysema	0.979	0	0	0
Fibrosis	0.985	0	0	0
Pleural Thickening	0.966	0.023	0.003	0.013
Hernia	0.998	0	0	0
Model Average Performance	0.941	0.065	0.024	0.045

Confusion Matrix:

```
['0.83', '0.10', '0.01', '0.05', '0.01', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.11', '0.80', '0.08', '0.01', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.02', '0.11', '0.86', '0.01', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.04', '0.00', '0.10', '0.86', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.13', '0.11', '0.01', '0.87', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.02', '0.01', '0.05', '0.92', '0.02', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.03', '0.01', '0.35', '0.01', '0.00', '0.60', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.01', '0.01', '0.03', '0.12', '0.02', '0.01', '0.80', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.01', '0.04', '0.00', '0.95', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.10', '0.04', '0.00', '0.01', '0.00', '0.00', '0.00', '0.00', '0.00', '0.80', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.07', '0.00', '0.00', '0.00', '0.00', '0.00', '0.08', '0.85', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
['0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00', '0.00']
```

Discussions

(i)

With the exception of experiment 6 and 7, where the weighted loss function was used, the networks we tested were generally not great at detecting the diseases. In experiments 1 to 5, the accuracy was high, hovering around 94.8%. However, the low precision, recall, and BCR statistics show that these models were inept at recognizing diseases. While we used a weighted loss function in experiments 6 and 7 to address the class imbalance problem, we had to reduce the size of training data in order to allow the model to converge, due to time and resource problems. Because of this, it did not generalize well to the test set, hence the mediocre accuracy and precision, recall, and balanced classification rate statistics. However, the higher accuracy on the training set and the existence of non-zero values on the confusion matrix for these experiments show that by addressing the class imbalance problem, the model was learning to identify diseases, rather than simply predicting negatives.

We think the reason that the networks in experiments 1-5 did not perform well is that the class imbalance promoted the network to simply learn the majority output, which is a negative diagnosis. This shows up in how the accuracy and loss on the validation did not change dramatically. We think this is because the model in each validation test simply guessed 0 for most, if not all the classes regardless of the input. Also, the testing accuracy seems to oscillate heavily because the network seems to perform better or worse on each minibatch based on the number of rare diagnosis in the minibatch.

In experiment 6 and 7, we had the same variation based on the difficulty of the minibatch, but there is a general upwards trend in terms of accuracy, and general downwards trend in terms of loss. So, even though the network performed better or worse on any given minibatch, it generally performed better as the network trained due to the weighted loss function.

(ii)

The most common confusion in our model was between infiltration and pneumonia. This makes sense since infiltration is some excess accumulation of foreign substances and pneumonia is the excess accumulation of fluid in the lungs. Given that these definitions are so similar and that all the images are chest x-rays (so the lungs are in view), it makes sense that our model is confusing between the two, since the images classifying infiltration and pneumonia likely look very similar.

Confusions were only exhibited in experiment 6 and 7, with the weighted loss function. In experiments 1-5, where class imbalance was not addressed, the confusion matrix outputted all zeroes. This is because the confusion matrix was implemented in a way such that it would only add values if the output of the network matched the label (output = 1 and label = 1), meaning the network was correct, or if the output of the network guessed there was a disease and there was none (output = 1 and label = 0), meaning the network was wrong by guessing something was there when there was not anything. When class imbalance was not addressed, the network learned to predict negative (0) all the time, which led to nothing being added to the confusion

matrix. In experiments 6 and 7, where class imbalance was addressed, the confusion matrix had values on it, since it learned to predict positive cases.

(iii)

Accuracy statistic is used to simply measure the number of correct diagnosis relative to the total number of diagnoses. The issue with simply following this value is that it does not take into account whether the correct diagnoses are true positives and true negatives, and whether the incorrect diagnoses are false positives or false negatives. Therefore, this stat is heavily biased by class imbalances within the dataset.

Precision addresses the question: of all the images the network predicts has a disease, what fraction of the images actually show a disease? Having high precision is desirable, since we want the network to be able to predict diseases when the images shows a disease. Precision is hurt by high number of false positives.

Recall addresses the question: of all the images that actually contain a disease, what fraction did the network correctly detect as having a disease? Having high recall is desirable as well, since we want the network to be able to correctly detect as many diseases as possible. Recall is hurt by having a high number of false negatives.

Balanced classification rate combines the two to see how well the network does on both precision and recall. Given the nature of precision and recall, if one goes up, the other goes down. Using balanced classification rate allows us to determine how well the network is doing on both; this helps us to give a more detailed view on how false negatives or false positives taking from the total accuracy of the model. In a sense, the precision, recall, and BCR values give us a sense of whether the model is too liberal or restrictive in producing positive or liberal outputs relative to the training data.

As stated above, precision and recall allows us to answer different questions on how well the model performs, as opposed to raw accuracy or plotted loss. In experiments 1 to 5, we see that the accuracy is relatively high near 95% and the loss plots look good. However, we know the models in these experiments are not good at predicting diseases, since they learn to predict negatives all the time, because of the class imbalance. We can see that these models are not good because the precision and recall are very low. Theoretically experiments 6 and 7 should have had high precision and recall statistics, but as mentioned above, the time and resource problems did not allow the model to fully train on the entirety of the training data, thus leading to a model that does not generalize well. These higher statistics would have shown that these models were much more capable of identifying diseases.

(iv)

Theoretically, the feature maps are very simple for earlier layers, and before more complicated deeper into the network as the convolutional layers are learning the internal representations of the model. However, the filters we used were fairly small, as the largest filter is 7x7 pixels. So, this makes it difficult to see a significant difference in the filters. However, something we do notice in the filters of experiment 6 and 7 is that the filters later in the network tend to be more structures, as there is as dramatic shift in weights of adjacent pixels. This shows

up the visualization as the lighter colors are closer together rather than randomly on the filter. This may be explained by the idea that the filters of earlier layers are learned to identify very basic shapes, therefore causing the erratic weights next to each other, while the later filters are looking to identify larger and more complicated shapes in the images.

References

LeCun, Yann, Leon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. "Gradient-Based Learning Applied to Document Recognition." *Proceedings of the IEEE*86 (11): 2278–2324. doi:10.1109/5.726791.

Rajpurkar, Pranav, Irvin, Jeremy, Zhu, Kaylie, Yang, Brandon, Mehta, Hershel, Duan, Tony, Ding, Daisy, Bagul, Aarti, Ball, Robin L., Langlotz, Curtis, Shpanskaya, Katie, Lungren, Matthew P., and Ng, Andrew Y. *CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning*, 2017

Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri: "ChestX-ray8: Hospital-scale Chest X-ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases", 2017; arXiv:1705.02315.

Authors' Contributions

Wesley:
For the code, I worked on implementing the confusion matrix and precision recall statistics. For the report, I worked on formatting and writing up the discussion.

Udai:
On the programming side, I worked on implementing the 2 new architectures to test, and created the experiment files to run. I also implemented the weighted loss function to resolve the class imbalance issue. In the write up, I worked on creating the descriptions of the architectures, and gathering the data presented in the results section. I also worked on writing the related works and discussion section of the report.