Udaikaran Singh and Wesley Kwan

Gary Cottrell

CSE 190

11/11/2018

**PA3: Deep Convolutional Network for Thorax Disease Detection**

**Udaikaran Singh Wesley Kwan**

Department of Computer Science Department of Computer Science

University of California, San Diego. University of California, San Diego

La Jolla, CA 92093. 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 \_\_\_\_\_\_\_\_\_, with an overall accuracy of \_\_\_\_\_\_\_\_\_\_. In this model, the precision was \_\_\_\_\_\_\_\_\_\_\_\_\_, the recall was \_\_\_\_\_\_\_\_\_\_\_\_\_\_, and the balanced classification rate was \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

**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. Also, 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.

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 primary source we were inspired by is a Stanford paper that also studied the same dataset. The paper provided a rudimentary loss function, detailing how to deal the class imbalance. From this, we created a weighted loss function, similar to the one the Stanford group used (Rajpurkar et al. 2017). CAN ADD MORE IF WE USED OTHER PAPERS.

**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 = 121032  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. 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 |

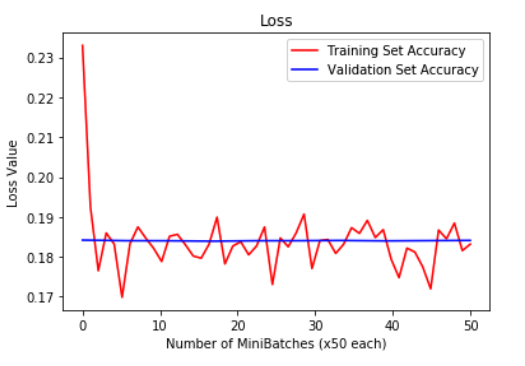
For both of these architectures, we used the Adam gradient descent optimizer, no regularization function, and Xavier weight initialization. The class imbalance was addressed by testing a weighted loss function that punishes false negatives heavily. The idea is that this punishes the model for outputting 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.

**Results**

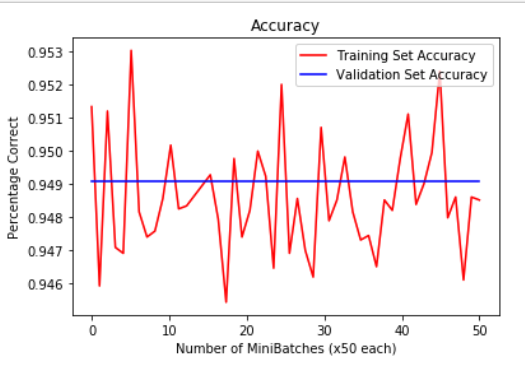
In experiment 1, we tested the baseline architecture described above. In experiment 2 and 3, we tested architecture 1 and 2 as described above. In experiment 4, we tested normalizing the inputs using transforms.Normalize with architecture 2. In experiment 5, we tested scaling down the inputs by a half using transforms.Scale with architecture 2. 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. IS THIS CORRECT?

Experiment 1: Baseline Architecture

(i) Loss Curves

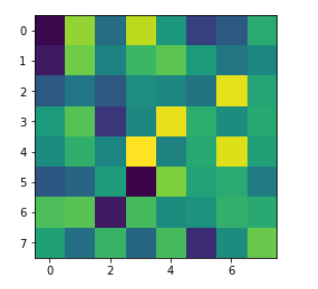
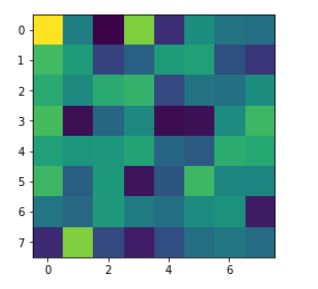


(ii) Accuracy Curves

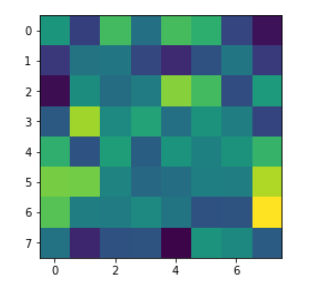
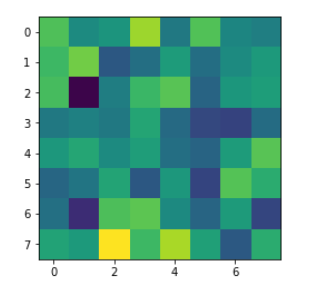


(iii) Visualization of filter maps

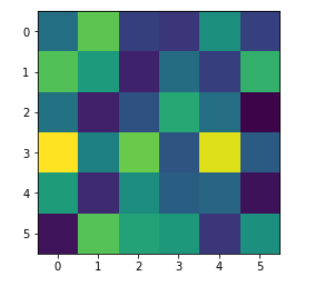
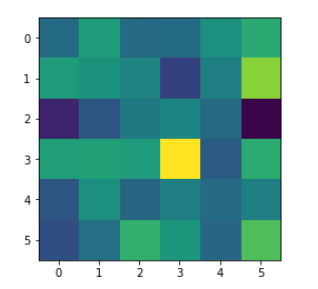
Convolution Layer 1

Convolution Layer 2

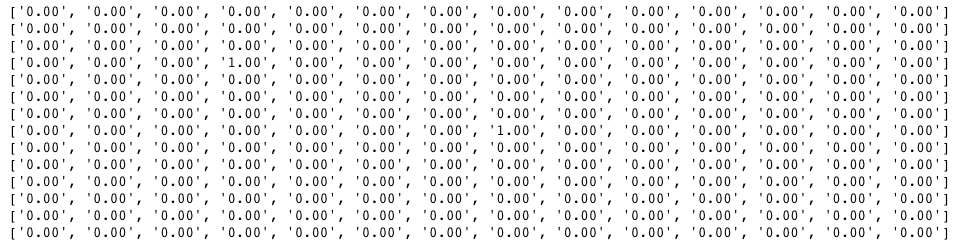
Convolution Layer 3

(iv) Model Results

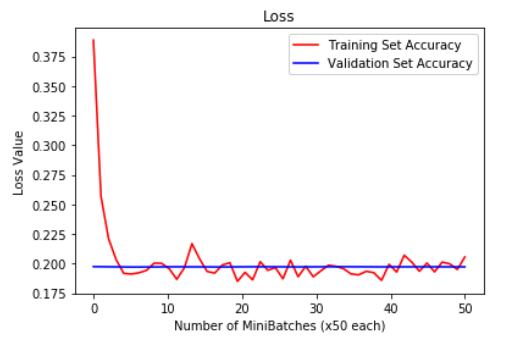
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Disease | Accuracy | Precision | Recall | BCR |
| Atelectasis | 0.896 | 0.5 | 0.0 | 0.25 |
| Cardiomegaly | 0.976 | 0.5 | 0.002 | 0.251 |
| Effusion | 0.881 | 0.5 | 0.0 | 0.25 |
| Infiltration | 0.82 | 0.667 | 0.001 | 0.334 |
| Mass | 0.95 | 0.5 | 0.001 | 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:

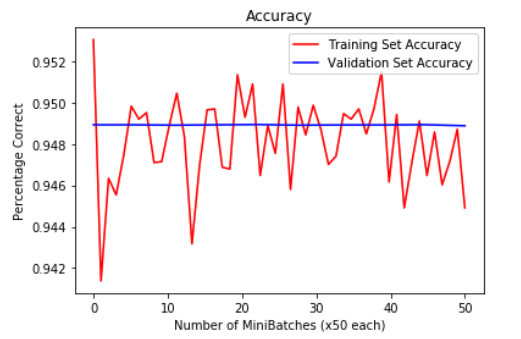


Experiment 2: Architecture 1 (w/o addressing class imbalance)

(i) Loss Curves

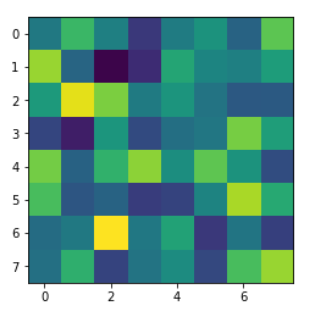
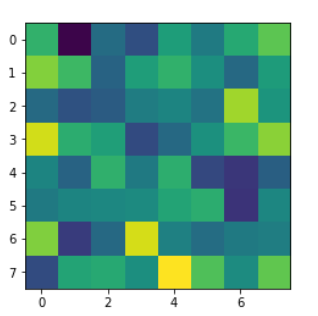


(ii) Accuracy Curves

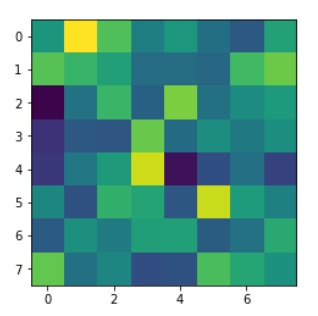


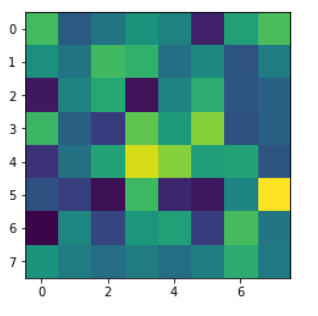
(iii) Visualization of filter maps

Convolution Layer 1

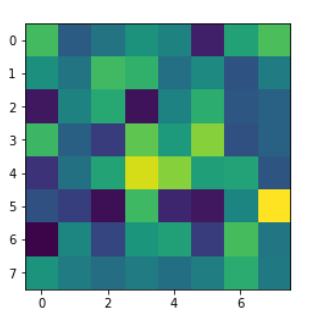
 

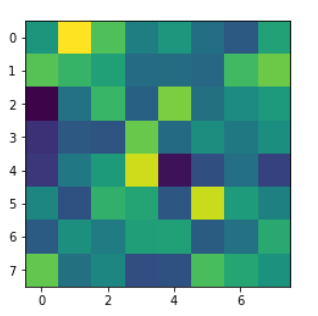
Convolution Layer 2





Convolution Layer 3

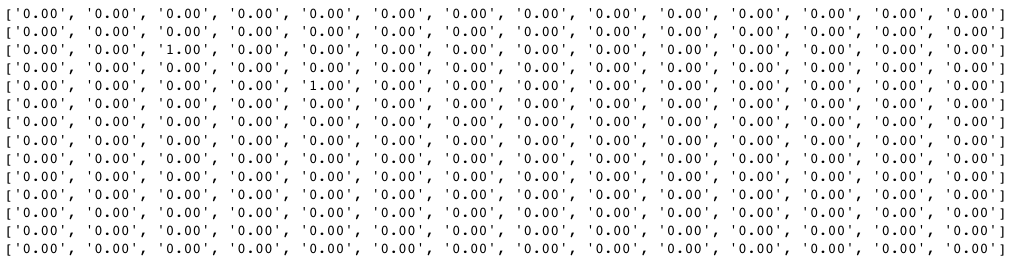




(iv) Model Results

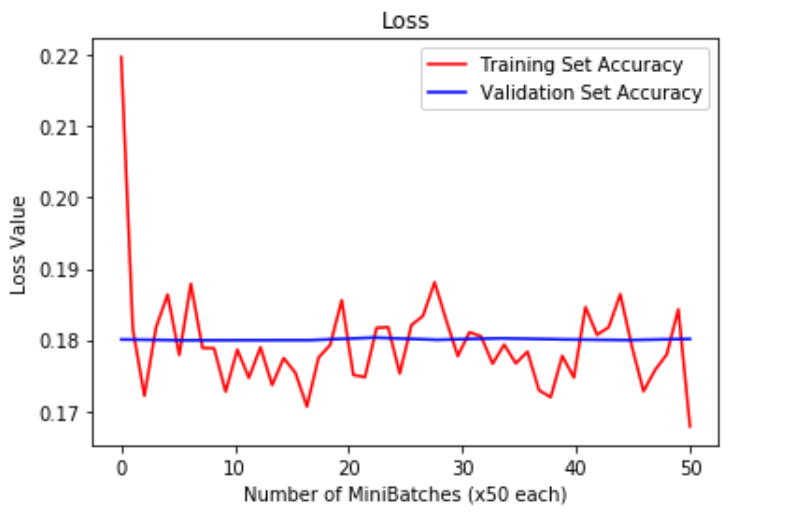
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Disease | Accuracy | Precision | Recall | BCR |
| Atelectasis | 0.897 | 0 | 0.0 | 0.0 |
| Cardiomegaly | 0.975 | 0 | 0 | 0 |
| Effusion | 0.881 | 0.25 | 0.0 | 0.125 |
| Infiltration | 0.826 | 0 | 0 | 0 |
| Mass | 0.951 | 0.5 | 0.004 | 0.252 |
| Nodule | 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:

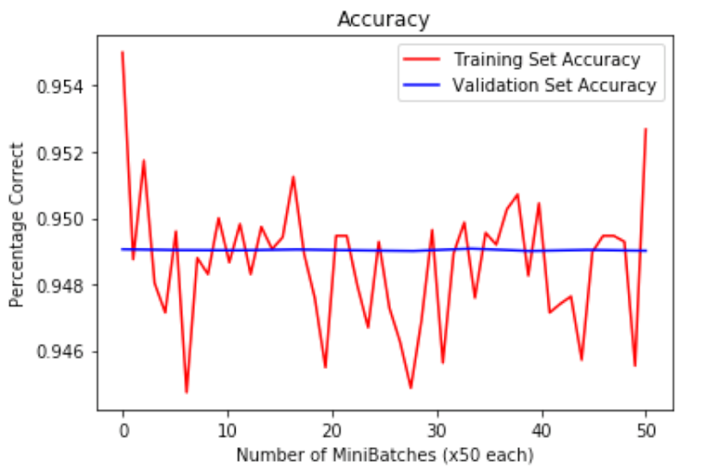


Experiment 3: Architecture 2 (w/o addressing class imbalance)

(i) Loss Curves

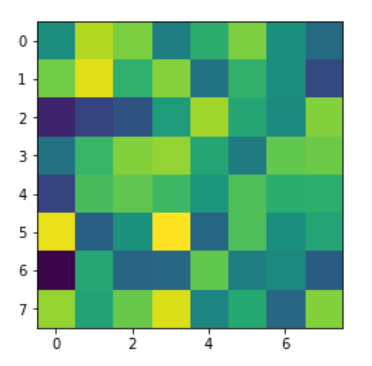
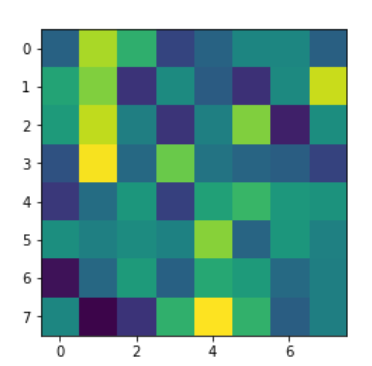


(ii) Accuracy Curves

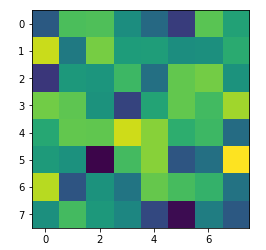
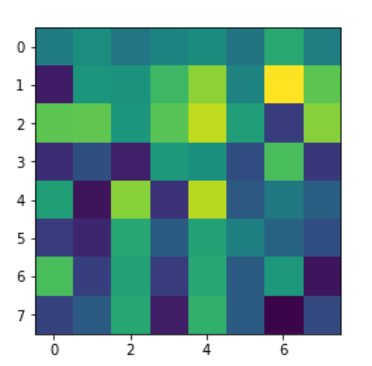


(iii) Visualization of filter maps

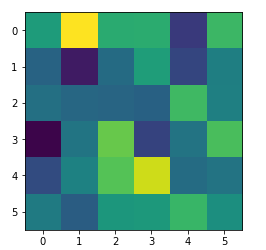
Convolution Layer 1

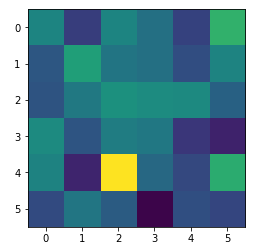
 

Convolution Layer 3

Convolution Layer 5

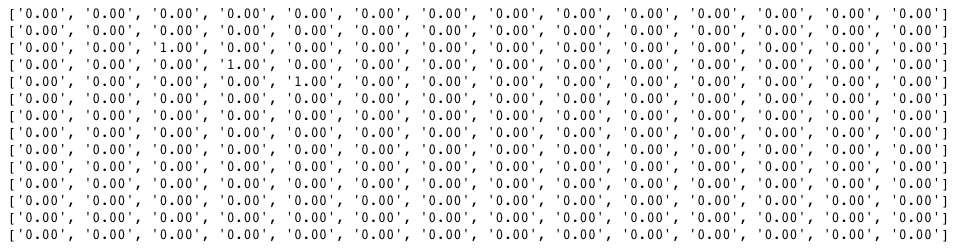




(iv) Model Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 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:

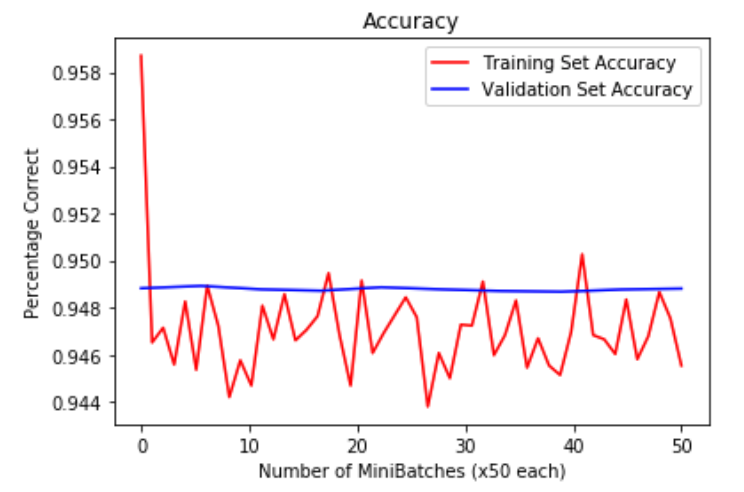


Experiment 4: Architecture 2 with Normalized Inputs

(i) Loss Curves

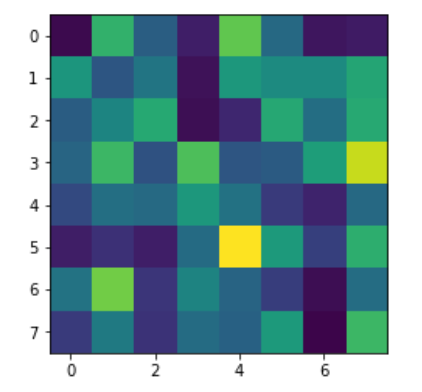


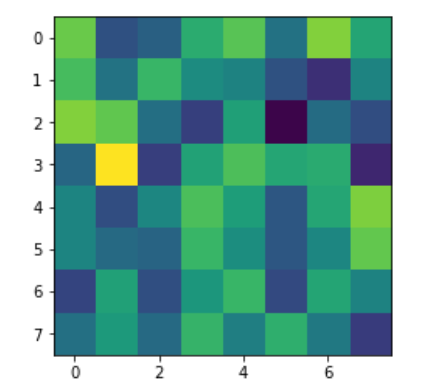
(ii) Accuracy Curves



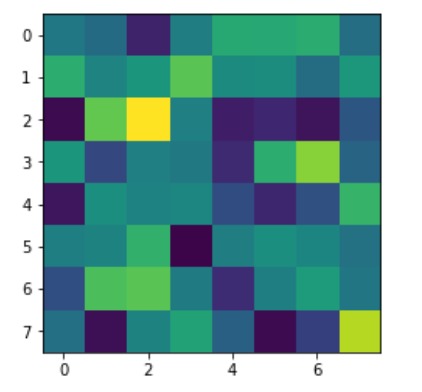
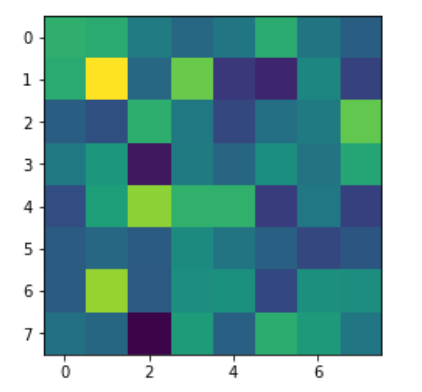
(iii) Visualization of filter maps

Convolution Layer 1

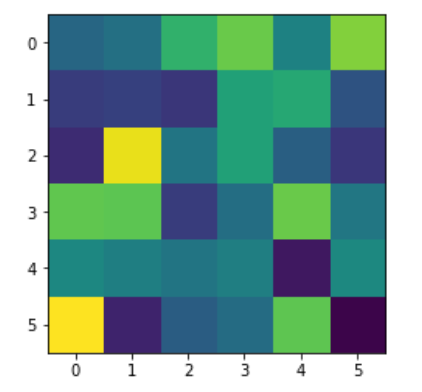
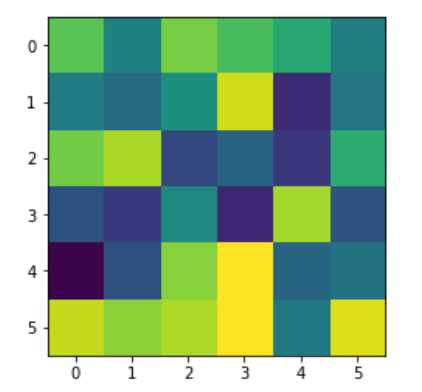




Convolution Layer 3

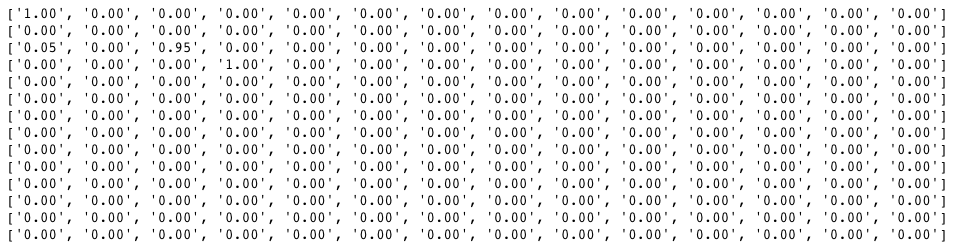
Convolution Layer 5

(iv) Model Results

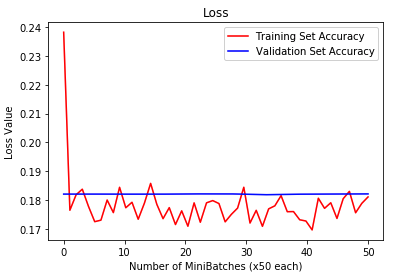
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Disease | Accuracy | Precision | Recall | BCR |
| Atelectasis | 0.881 | 0.232 | 0.057 | 0.144 |
| Cardiomegaly | 0.977 | 0 | 0 | 0 |
| Effusion | 0.881 | 0.727 | 0.003 | 0.365 |
| Infiltration | 0.82 | 0.364 | 0.047 | 0.205 |
| Mass | 0.947 | 0 | 0 | 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:

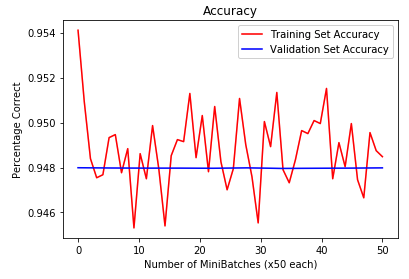


Experiment 5: Architecture 2 with Scaled Down Image (by ½)

(i) Loss Curves

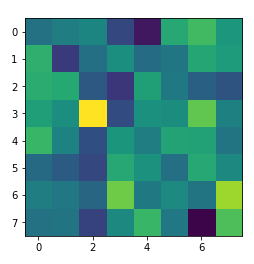


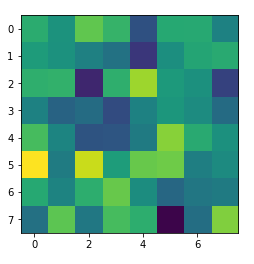
(ii) Accuracy Curves



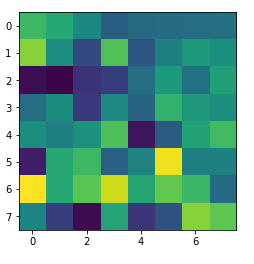
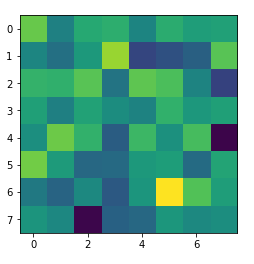
(iii) Visualization of filter maps

Convolution Layer 1

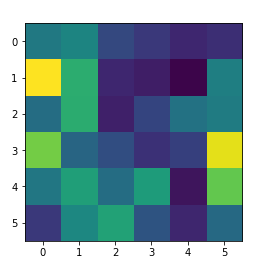


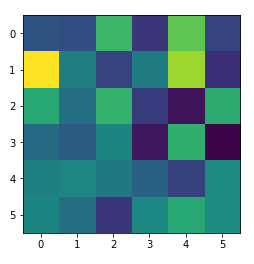


Convolution Layer 3

Convolution Layer 5

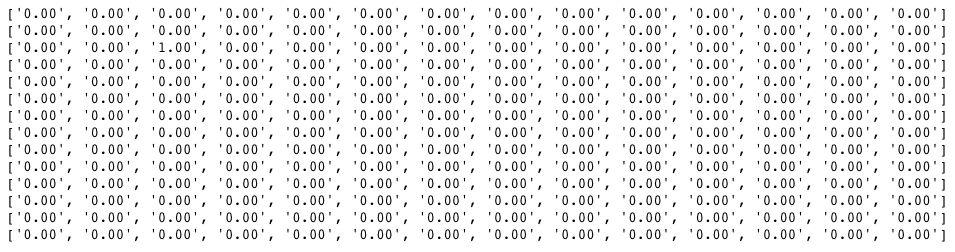




(iv) Model Results

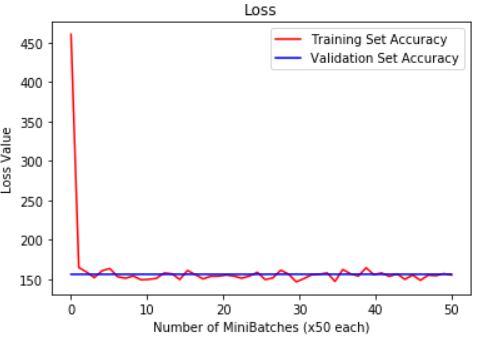
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Disease | Accuracy | Precision | Recall | BCR |
| Atelectasis | 0.897 | 0 | 0 | 0 |
| Cardiomegaly | 0.976 | 0 | 0 | 0 |
| Effusion | 0.883 | 1.0 | 0.001 | 0.501 |
| Infiltration | 0.819 | 0 | 0 | 0 |
| Mass | 0.948 | 0 | 0 | 0 |
| Nodule | 0.944 | 0 | 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:

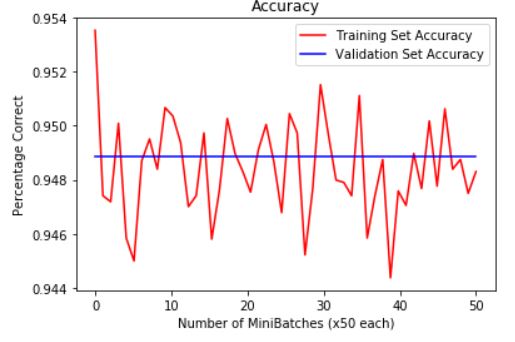


Experiment 6: Architecture 2 with weighted objective function

(i) Loss Curves

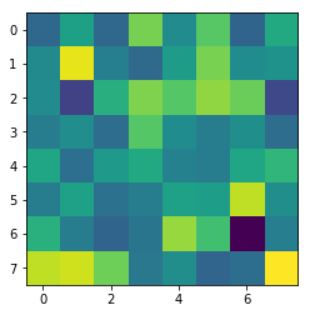
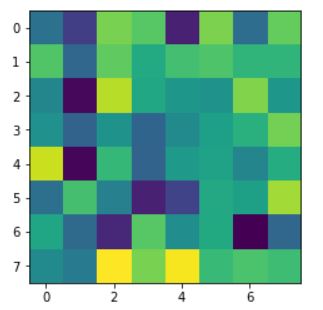


(ii) Accuracy Curves

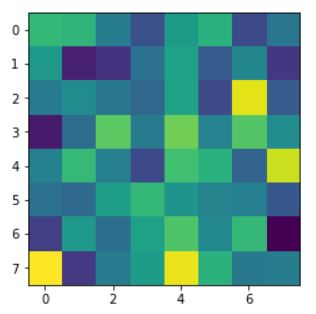
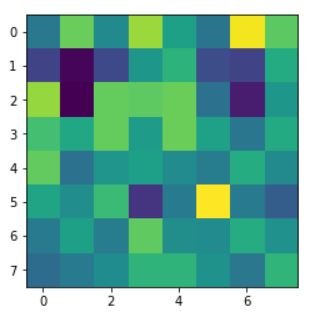


(iii) Visualization of filter maps

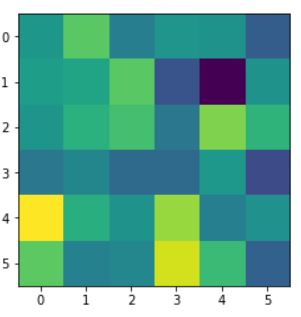
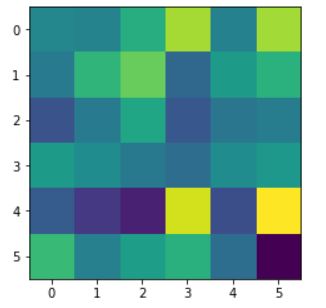
Convolution Layer 1



Convolution Layer 3



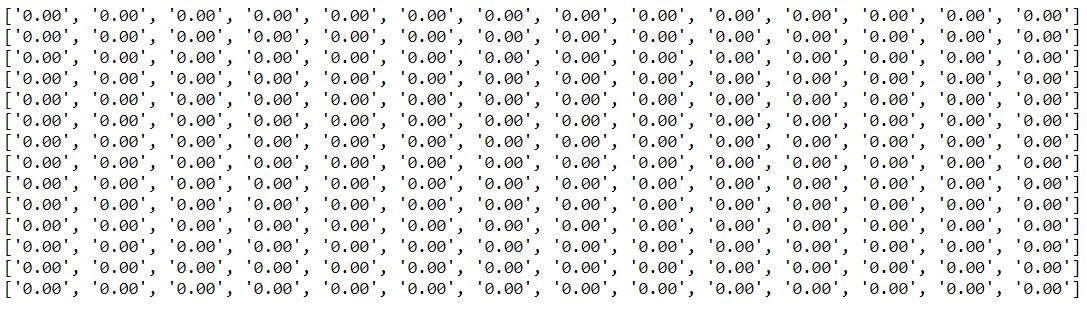
Convolution Layer 5



(iv) Model Results

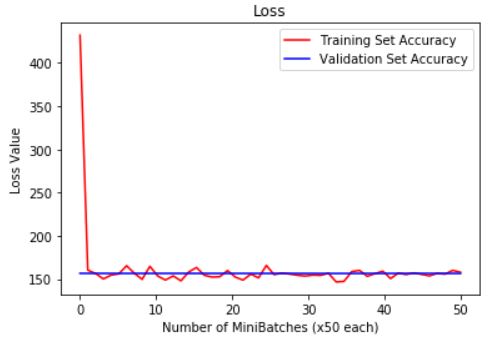
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Disease | Accuracy | Precision | Recall | BCR |
| Atelectasis | .898 | 0 | 0 | 0 |
| Cardiomegaly | 0.977 | 0 | 0 | 0 |
| Effusion | 0.886 | 0 | 0 | 0 |
| Infiltration | 0.823 | 0 | 0 | 0 |
| Mass | 0.949 | 0 | 0 | 0 |
| Nodule | 0.947 | 0 | 0 | 0 |
| Pneumonia | 0.987 | 0 | 0 | 0 |
| Pneumothorax | 0.956 | 0 | 0 | 0 |
| Consolidation | 0.957 | 0 | 0 | 0 |
| Edema | 0.979 | 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 | 0 | 0 |

Confusion Matrix:

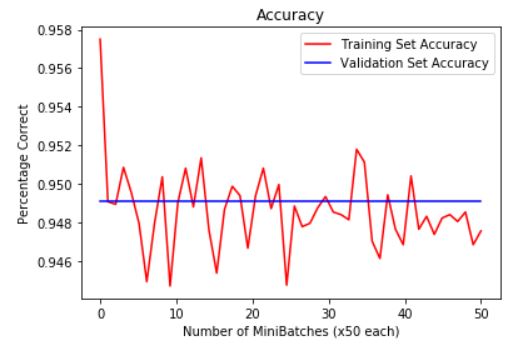


Experiment 7: Architecture 2 with weighted objective function & Normalized Inputs

(i) Loss Curves

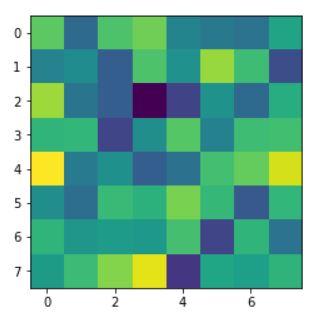


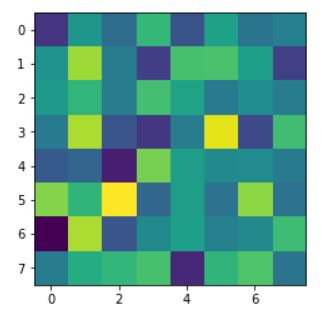
(ii) Accuracy Curves



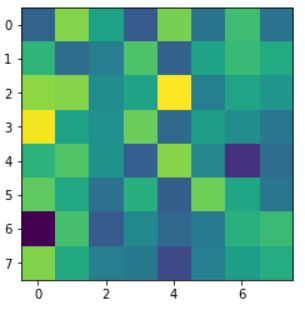
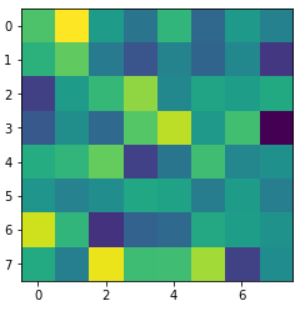
(iii) Visualization of filter maps

Convolution Layer 1

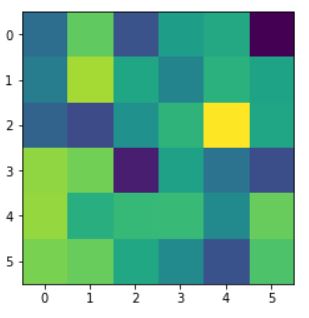
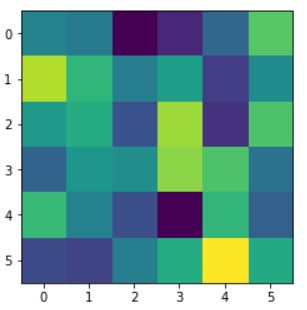




Convolution Layer 3



Convolution Layer 5



(iv) Model Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Disease | Accuracy | Precision | Recall | BCR |
| Atelectasis | 0.898 | 0 | 0 | 0 |
| Cardiomegaly | 0.974 | 0 | 0 | 0 |
| Effusion | 0.879 | 0 | 0 | 0 |
| Infiltration | 0.827 | 0 | 0 | 0 |
| Mass | 0.95 | 0 | 0 | 0 |
| Nodule | 0.944 | 0 | 0 | 0 |
| Pneumonia | 0.988 | 0 | 0 | 0 |
| Pneumothorax | 0.952 | 0 | 0 | 0 |
| Consolidation | 0.957 | 0 | 0 | 0 |
| Edema | 0.982 | 0 | 0 | 0 |
| Emphysema | 0.976 | 0 | 0 | 0 |
| Fibrosis | 0.985 | 0 | 0 | 0 |
| Pleural Thickening | 0.97 | 0 | 0 | 0 |
| Hernia | 0.998 | 0 | 0 | 0 |
| Model Average Performance | 0.949 | 0 | 0 | 0 |

**Discussions**

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 95%. However, the low precision, recall, and BCR statistics show that these models were inept at recognizing diseases. In contrast, experiment 6 and 7 fared much better, with higher accuracy at around \_\_\_\_\_\_\_\_\_\_\_\_, and having higher precision, recall, and BCR statistics.

Common confusions \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. These 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.

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. 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. 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, meaning if this metric is high, the network is performing well on both precision and recall.

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. In contrast, experiments 6 and 7 have high precision and recall, in addition to good accuracy and loss plots. This means the models in these experiments are actually capable of identifying diseases, as opposed to simply predicting negatives.

The visualizations shown in experiments 1 to 5 are rather meaningless, since the network is not learning how to predict diseases. This is reflected in the visualizations of the convolutional layers, which look random and pattern-less. In contrast, the visualizations in experiments 6 and 7 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

**References**

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

**Authors’ Contributions**

**(…)**