CS230

Diagnosing Chest Abnormalities with Deep Learning

Philip J. Lambert, Ryan D. Ludwick, Kyle P. Orciuch
Department of Computer Science
Stanford University
{plambert, rludwick, korciuch} @stanford.edu

Abstract

Detecting thoracic (i.e. chest) abnormalities via convolutional neural networks has recently been of piquing interest to the computer vision community. Our project seeks to build upon related work done already with AlexNet, by contributing 14 binary classifiers which can individually discern the most common chest issues diagnosed via X-ray imaging. Furthermore, we attempt an all-in-one multi-class classifier model from scratch and eventually via transfer learning with the first 10 layers of Inception V3 frozen. We utilize a dataset set of roughly 18,000 radiologist-labeled DICOM images to evaluate our models.

1 Introduction

Throughout the year-long duration of the COVID-19 pandemic, access to healthcare has been emphasized unlike any other time in American history. Our system has been stretched to the max while providing care for millions of Americans.

One way that we can ease this stress is through reducing diagnosis time. Throughout the medical field, patient diagnosis is regarded as perhaps one of the most important areas where technology excels. From the very first innovations like the invention of the x-ray, doctors and patients alike are thankful for the emergence of technological help.

In recent years, Neural Networks have arisen as a popular medical aid in image interpretation and abnormality detection; in our case, this applies specifically to Chest X-rays. We use multiple different forms of Neural Networks to identify 14 types of disease or abnormality from input Chest X-ray images. These abnormalities include cardiomegaly (an enlarged heart), pleural effusion (fluid buildup between the lungs and the chest), and pulmonary fibrosis (scarred and diseased lung tissue). Our Neural Network detection systems are composed of different forms of Deep Learning models, like Multi-Class Classifiers (to identify the presence of any of the 14 abnormalities) and Binary Classifiers (to identify the presence or absence of 1 specific abnormality). We utilize the Convolutional Neural Network framework to work with the X-ray image data most effectively. Each of our proposed models has its strengths and weaknesses; we feel that it's important to try a large breadth of models on this problem in order to fully gauge the success capability of any one classifier.

Ultimately, we feel that these advanced detection systems will be extremely useful for streamlining the Chest X-ray diagnosis process and improving hospital efficiency. It's important to note that even the most advanced Neural Network systems can't substitute for an educated doctor's diagnosis, and we aren't proposing that our model should do this. Rather, we believe that our models can be used as a supplement, guiding the practitioner towards likely abnormalities and speeding up the workflow. Any improvement in diagnosis efficiency will, in the long run, allow more patients to be diagnosed, treated, and hopefully cured.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

2 Related work

As is to be expected, the Chest X-ray domain has been studied intensely by deep learning experts. A consensus has seemingly been formed around four models being the most effective: ResNet-50, AlexNet, the Inception Network, and Dense-121.

Rajpukar et al. argue for the Dense-121 model, noting that they can achieve results even better than typical doctor diagnosis with a single model[1]. This model is also known as the ChexNet.

Another common model in use is the AlexNet. This network consists of 5 convolutional layers, 5 max pooling layers, and a fully connected layer, similar to traditional Convolutional Neural Networks. AlexNet differs in that it uses the ReLU activation function, rather than the commonly-used sigmoid activation function, for each layer. Multiple Chest X-ray diagnosis models have used the AlexNet with great success, both in classifying chest abnormalities[2] and in detecting COVID-19[3].

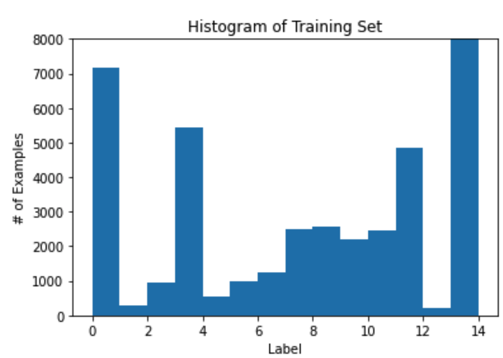
The Inception V3 model is another successful CNN that is 48 convolution layers deep. It was trained on over a million images from ImageNet and can classify over 1000 different images. This model is popularly used in transfer learning, specifically with multi-classifications.

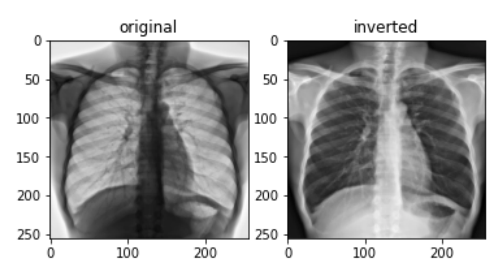
The final commonly used model is ResNet-50. Many researchers have leveraged the Residual Network structure to great success. Baltruschat et al., though, argue that Resnet-50 depth is not necessary to achieve optimal results, and propose a modified Resnet-38 model for Chest X-ray classification[4].

We will draft our work to expand upon some of these four different approaches. The AlexNet model acts as a baseline for our binary classifiers, although we can expand upon its minimal architecture for improved results. We can also utilize pretrained weights for the Inception Network model , together with our novel dataset, to produce significant classifiers.

3 Dataset and Features

Our group sourced a dataset from the Vingroup Big Data Institute's prediction competition, found on Kaggle's website, which consists of roughly 191.82 GB of postero-anterior CXR scans - 15,000 labeled images for the training phase and 3,000 labeled images for the testing phase[1]. Their `train.csv` file contains the radiologist's findings for classification and the coordinates of their bounding boxes for localization. The 14 possible output labels are as follows: Aortic enlargement, Atelectasis, Calcification, Cardiomegaly, Consolidation, ILD, Infiltration, Lung Opacity, Nodule/Mass, Other lesion, Pleural effusion, Pleural thickening, Pneumothorax, and Pulmonary fibrosis[2]. The distribution of the training data set is as follows:





Not all of the Xrays were the same image type, some had black backgrounds and other had white. We inverted those with a white background. Therefore, all xrays will be uniform with a black background and white highlighting.

4 Methods

4.1 Binary Classifiers

Each binary classifier was trained on some variant of the AlexNet model, which consists of 5 convolutional layers, 5 max pooling layers, and 2 densely connected layers. We found through trial and error that adding more Convolutional layers increased model performance, especially with regards to smaller and harder-to-detect abnormalities. We also found that decreasing the Convolutional kernel size to 3x3 improved model performance across all abnormalities, albeit with an increase in computational time. Finally, we observed that adding either 1 or 2 Dropout layers greatly improved performance across all abnormalities. We added 1 Dropout layer for all abnormalities as the last layer prior to the Densely Connected layers, in an attempt to reduce model variance; this addition worked splendidly, and improved test accuracy by as much as 15% in some models. Additionally, for some abnormalities we added another Dropout layer in the middle of our Convolutional layers. This was motivated by previous work that utilized Dropout prior to increasing the number of channels in Convolutional layers. The results of this second Dropout layer varied across classifiers, but it significantly benefited a few.

Thus, we ended up with a slightly modified version of the AlexNet model for each binary classifier. Each classifier has between 13 and 19 Convolutional layers, 5 Max Pooling layers, 2 Densely Connected layers, and 1 or 2 Dropout layers. In addition to this model structure, we utilized the binary cross-entropy loss function to assess our results. For abnormality a_i, where $i \in \{0, ..., 13\}$, the loss function can be expressed as follows:

$$L(y_{a_i}, \hat{y_{a_i}}) = -\sum_{j=1}^{n} y_{a_i}^{j}(log(\hat{y_{a_i}}^{j})) + (1 - y_{a_i}^{j})(log(1 - \hat{y_{a_i}}^{j}))$$

4.2 Transfer learning

Another method we used to classify the xrays was transfer learning. We utilized the Inception V3 model as our base, which is a CNN with 48 layers, that consists of many convolution layers paired with either max or average pooling. After these convolution blocks, the model ends with a dropout, fully connected, and softmax layer that can predict 1000 images. For our transfer learning application, we want to remove the last softmax in order to train it to our classifications using a 15 softmax classifier instead. The idea of transfer learning is to train over top of the pre-trained Inception V3 layers. Because the model is pre-trained, it will already be very good at recognizing image features like edges. Therefore, we will have much better weight initialization when training our own images. Lastly, we added some dense layers of our own to the end of Inception model. The dense layers we added at the end serve to use the features extracted from the convolution layers and better predict the xray classification. We used the categorical loss function to test our results. This loss is specifically used for multi-classifiers where the output is a one-hot encoding vector. $loss = -\sum_{i=0} y_i log \hat{y}_i$

5 Experiments/Results/Discussion

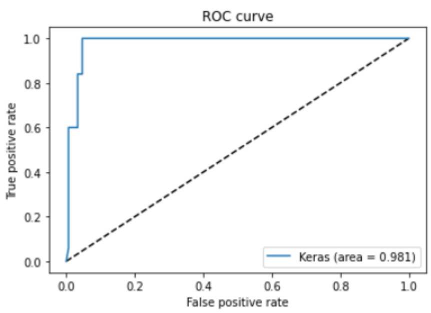
5.1 Gateway Classifier

We thought that a classifier which could discern between healthy and unhealthy would be a great start for a potential clinical workflow. If we can say with a high degree of confidence that the X-ray is normal, then we don't have to waste computational power by sending the image to the other classifiers. If the X-ray is abnormal in any way, we can send it to the binary classifiers and/or multiclass classifiers for further investigation. We achieved roughly 90% training accuracy with our Gateway Classifier via an Xception network (Appendix 8.5). However, the main focus of our project lies in the next two sections below.

5.2 Binary Classifiers

Each binary classifier was assessed using a ROC curve, which measures the performance of the classifier using different prediction thresholds. Additionally, we found the optimal threshold for each classifier (based on maximizing the F1 score) and reported this score and threshold. These detailed results for all 14 classifiers can be found in the appendices. For this section, we'll talk in depth about one specific classifier. Though tuning leads to different results for each classifier, the overarching process is very similar for each one: try a model, see what went wrong, tune parameters, and repeat.

Let's take an in-depth look at the Cardiomegaly classifier. We first assessed a model with the vanilla AlexNet framework and trained for 150 epochs. This did not produce ideal results; we had both a high false positive and high false negative rate, despite scoring well on the training data. This immediately suggested a model with high variance, and to counteract this effect we began iteratively adding Convolutional layers to our Neural Network and retesting. 17 Convolutional layers ended up being the sweet spot for this classifier, with an increase in the number of Convolutional channels occurring every 4 layers. We also added a Dropout layer after these Convolutional layers in an attempt to reduce variance. This model performed significantly better, reaching 0.914 AUROC, but was still struggling with false negatives. To combat this, we upsampled positive image examples of Cardiomegaly in training; instead of using 10% positive examples in the training set, which mimicked the actual distribution, we upsampled to 55% positive examples in the training set. We kept our test set consistent with the data distribution - 10% positive and 90% negative examples. To account for the large upsampling, we also increased the number of epochs from 150 to 250. The combined effects of these changes produced an even stronger classifier, which reached 0.981 AUROC, a 0.75 F1 score, and 0.94 accuracy on the test set.



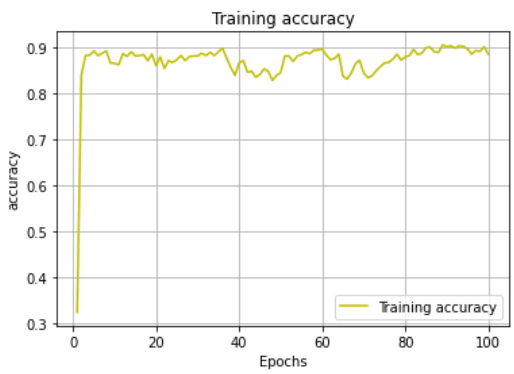
We proceeded in a similar manner for each of the other 13 binary classifiers, the results of which can be seen in the appendix. Notably, we did not use a default learning rate for each classifier; we noticed that our models seemed to perform better with lower learning rates, and so we attempted to push the learning rate as low as possible on each model. Similar approaches were taken with tuning other hyperparameters; we did not use one default value, but instead tuned separately for each classifier. The results of this are indicated in the appendix. We include a table with the learning rate used for each classifier, the number of epochs used to train the classifier, the optimal prediction threshold for each classifier, the number of Convolutional layers in the classifier, and the number of Dropout layers in the classifier for the reader's edification.

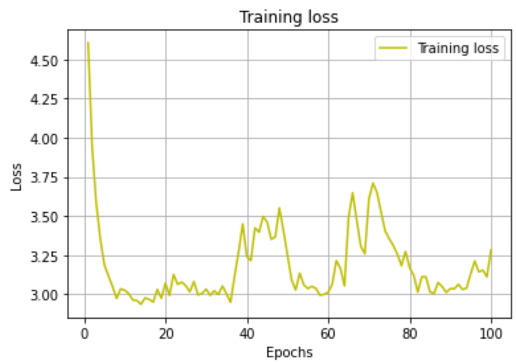
We noticed a few other important takeaways from binary classification. For instance, our classification models achieved higher F1 scores (on average) for more localized abnormalities. Localized abnormalities are ones concentrated in a specific (usually small) region; we were given a bounding box for each abnormality in the training data, and so were able to compare the average sizes and locations of different abnormalities. In contrast, our classification models did not do as well on less localized abnormalities, which could be located in many different places throughout the image and often have a large bounding box. One such example is Lung Opacity, which was perhaps our worst binary classifie.

Another important takeaway is that upsampling greatly improved performance in every classifier. Our great imbalance between positive and negative examples in the training set prevented learning gains at the beginning of our research; once we decided to upsample, though, our gains were markedly better.

5.3 Transfer learning

Since our image type (xray) was very different from the classifications of Inception V3 (pencil, keyboard, etc) we decided to make 38 of the layers trainable, all except for the first 10 layers which will capture basic structures like edges. After flattening the pre-trained model, I added 6 dense layers followed by a softmax output. I found that the best results came from cascading the number of units used in each dense layer, specifically cutting the units in half each time. The number of units started at 1024, followed by 512, 256, 128, 64, 32. When I tried to increase the number of dense layers by doubling up some of them, the results got worse so I concluded that more layers does not necessarily mean better results. Between each of the dense layers I applied batch normalization which achieved better results than dropout and other combinations of regularization. However, on the very last layer I used dropout which showed an increase in accuracy when applied here specifically. The inception V3 model actually does a technique similar to this where it only uses dropout at the end. Decreasing and increasing the dropout rate from .2 did not show to have any benefit. The Adam optimizer was found to achieve better accuracy results than SGD and rmsprop. Having 8 steps per epoch got the best results when training. By doing all of this fine tuning the hyper-parameters (layers frozen, dense layers, normalization, optimizers) I was able to get an accuracy of up to 90%, but mostly in the 80s range. My choices of hyperparameters resulted from experimentation and running many different models and comparing the results. Other hyperameters such as learning rate were kept at the default value for the adam optimizer.





6 Conclusion/Future Work

We explored 3 different strategies for tackling the problem of diagnosing chest x-ray abnormalities: binary classifiers from scratch, a multi-class classifier from scratch, and a multi-class classifier using transfer learning.

Our binary classifiers experienced the most success in a couple of different areas. First, our classifiers with highest F1 score were for Pleural Thickening, Cardiomegaly, and Pulminary Fibrosis; we hypothesize that this is due in part to the characteristic locality of these abnormalities. We also noticed success in binary classifiers when upsampling. We envision these binary classifiers as a "sanity check" for doctors. If a doctor suspects that one abnormality is present when glancing at an xray, they can use the binary classifier for that specific abnormality to check their initial thoughts.

It is clear that our multi-class classifier from scratch needs work. We thought this would be an interesting undertaking since most pre-trained models are typically trained with normal images, not X-rays, CT scans, etc. However, we realized that if we could approach Bayes' error via transfer learning, it might not be justified to pursue this further unless we had a longer development cycle.

Transfer learning also experienced high levels of success and was able to get results quickly with little data. By strategically freezing layers of the pre-trained model and adding additional dense layers, we were able to train the model to detect abnormalities with high accuracy. This multi classifier can help doctors make informed decisions quickly about xrays.

7 Contributions

Each team member contributed an equal amount of work. The team collaborated on the proposal, milestone, and final report. Kyle tackled the multi-class classifier from scratch, the individual healthy/unhealthy classifier, and worked on the data pipeline from Kaggle to Google Drive. Phil designed and iterated on each of the binary classifiers for individual abnormalities. Ryan focused on the multi-class classifier with Transfer Learning.

References

[1] Rajpukar, P., Irvin, J., Zhu, K., et al. (2017). CheXNet: Radiologist-level pneumonia detection on chest x-rays with deep learning.

[2] Bhandary, A., Prabhu, G., Rajnikanth, V., et al. (2019). Deep-learning framework to detect lung abnormality - a study with chest x-ray and lung ct-scan images.

[3] Ibrahim, A., Ozsoz, M., Serte, S., Al-Turjman, F., Yakoi, P. (2021). Pneumonia classification using deep learning from chest x-ray images during COVID-19.

[4] Baltruschat, I., Nickisch, H., Grass, M., Knopp, T., Saalbach, A. (2019). Comparison of deep-learning approaches for multi-label chest x-ray classification.

[5] Szegedy, Christian, et al. Rethinking the Inception Architecture for Computer Vision. 11 Dec. 2015.

[6] Chollet, François, et al. Keras. 2015. https://keras.io

[7] Abadi, Martín; Barham, Paul; Chen, Jianmin; Chen, Zhifeng; Davis, Andy; Dean, Jeffrey; Devin, Matthieu; Ghemawat, Sanjay; Irving, Geoffrey; Isard, Michael; Kudlur, Manjunath; Levenberg, Josh; Monga, Rajat; Moore, Sherry; Murray, Derek G.; Steiner, Benoit; Tucker, Paul; Vasudevan, Vijay; Warden, Pete; Wicke, Martin; Yu, Yuan; Zheng, Xiaoqiang (2016). "TensorFlow: A System for Large-Scale Machine Learning"

[8] Bhattiprolu, Sreenivas. "Bnsreenu/python_for_microscopists." GitHub, 21 July 2020, github.com/bnsreenu/python_for_microscopists/blob/master/143-multiclass_cifar.py.

[9] Team, Keras. "Keras Documentation: Image Classification from Scratch." Keras, keras.io/examples/vision/image_classification_from_scratch/.

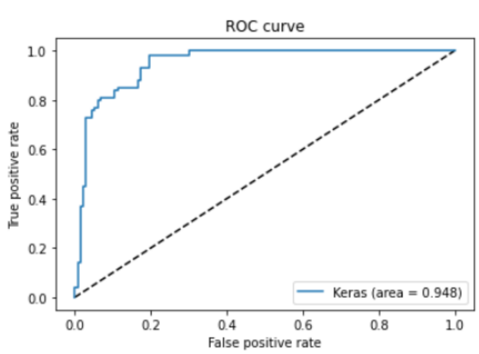
8 Appendix

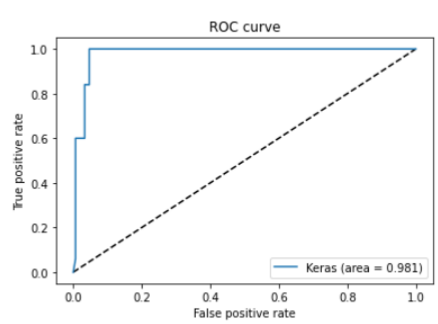
8.1 Binary Classifiers: Hyperparameter Choices

Table	Learning Rate	# Epochs	Threshold	# Convolutions	# Dropouts
Aortic Enlargement	0.00001	150	0.6	13	1
Cardiomegaly	0.00001	250	0.92	17	2
ILD	0.000015	175	0.5	13	1
Infiltration	0.00001	150	0.5	13	1
Lung Opacity	0.00001	225	0.5	19	2
Nodule or Mass	0.00001	150	0.5	13	1
Other Lesions	0.00001	300	0.85	19	2
Pleural Effusion	0.00001	250	0.9	19	2
Pleural Thickening	0.00001	150	0.5	13	1
Pulminary Fibrosis	0.00001	150	0.3	13	1

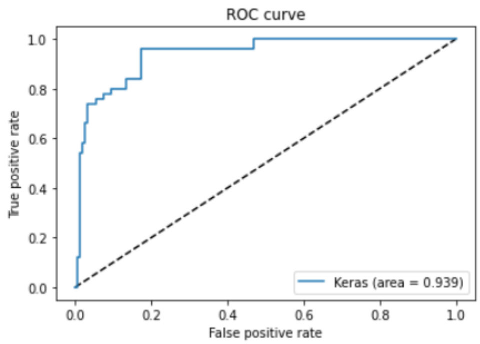
8.2 Binary Classifiers: ROC Curves

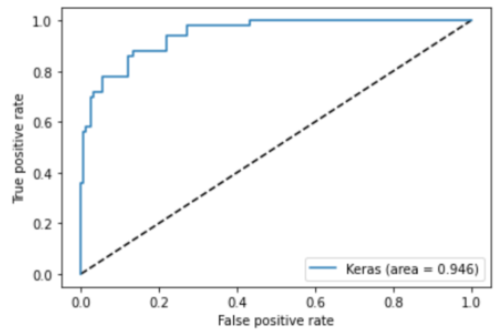
Aortic Enlargement and Cardiomegaly:



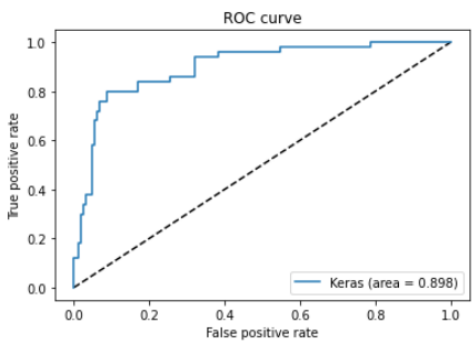


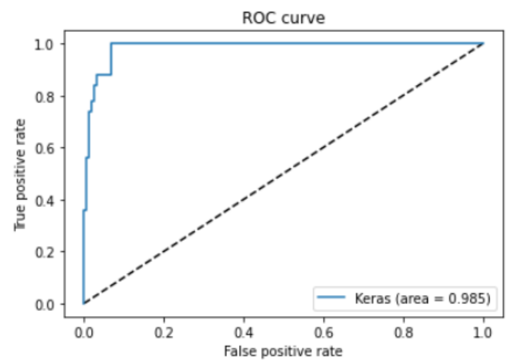
ILD and Infiltration:



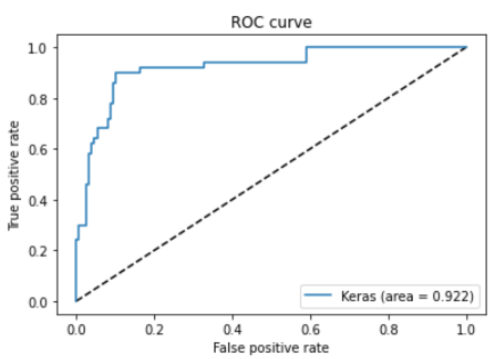


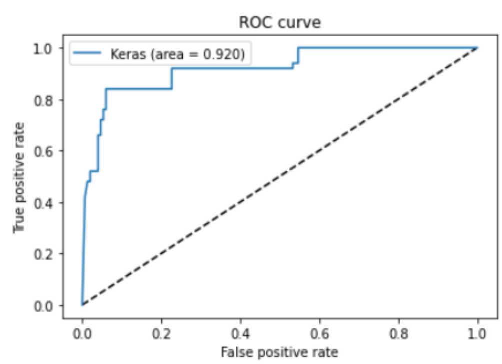
Lung Opacity and Nodule/Mass:



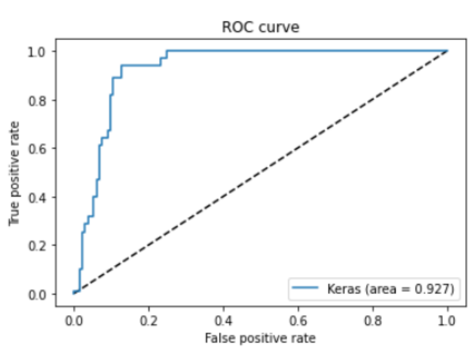


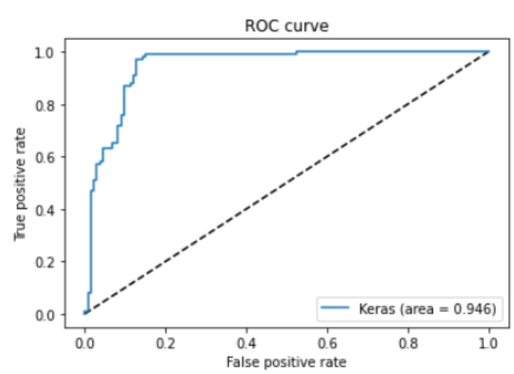
Other Lesions and Pleural Effusion:





Pleural Thickening and Pulminary Fibrosis:

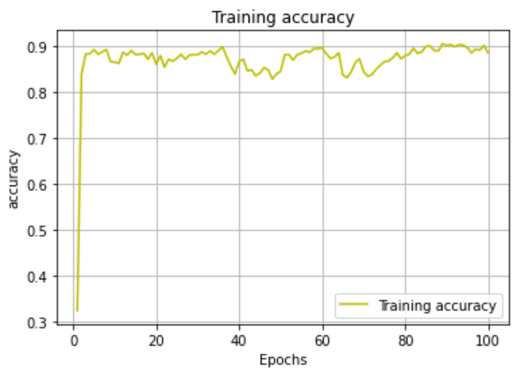


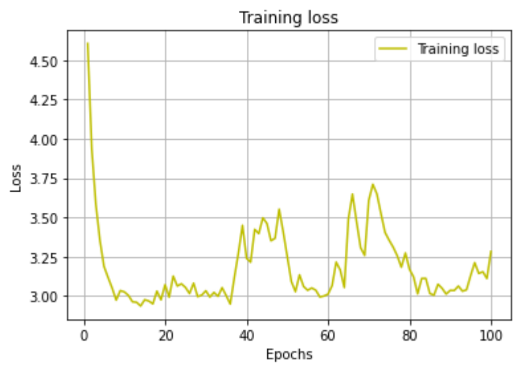


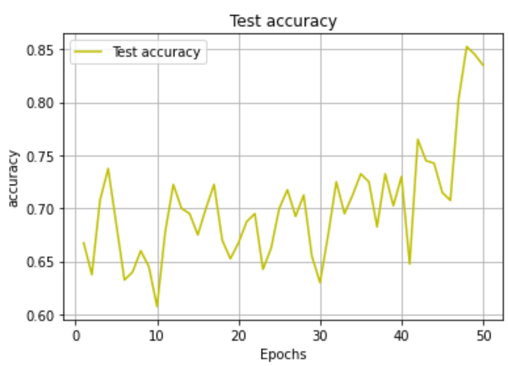
8.3 Binary Classifiers: AUROC, F1, and Accuracy Metrics

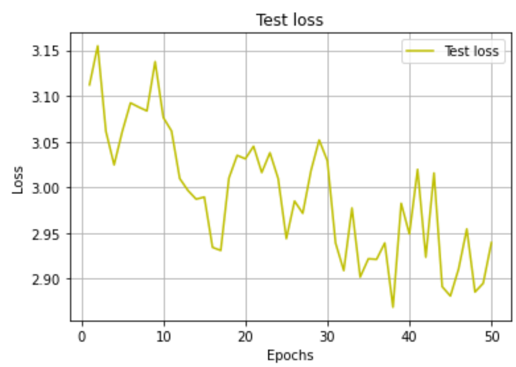
Table	AUROC	F1 Score	Accuracy
Aortic Enlargement	0.948	0.667	0.82
Cardiomegaly	0.981	0.750	0.940
ILD	0.939	0.645	0.812
Infiltration	0.946	0.624	0.906
Lung Opacity	0.898	0.551	0.870
Nodule or Mass	0.985	0.699	0.914
Other Lesions	0.922	0.637	0.902
Pleural Effusion	0.920	0.627	0.900
Pleural Thickening	0.927	0.767	0.886
Pulminary Fibrosis	0.946	0.708	0.850

8.4 Transfer learning: Accuracy and Loss Metrics



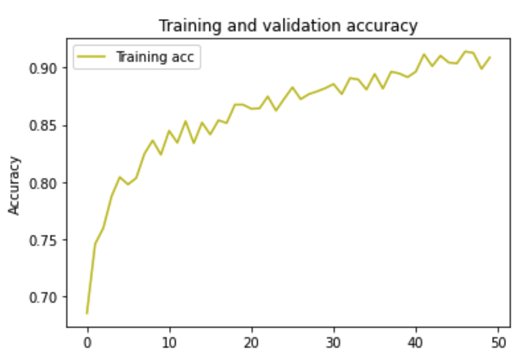


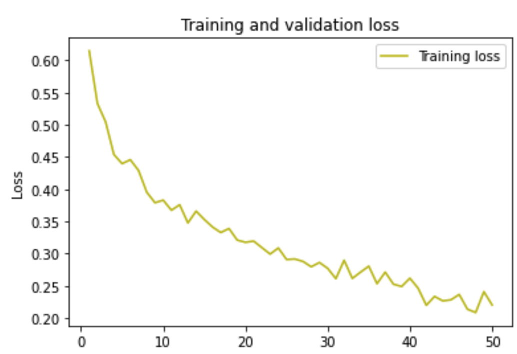




8.5 Gateway Classifier: Accuracy and Loss Metrics

Model architecture adopted from [9]





8.6 Multiclass Classifier from Scratch: Accuracy and Loss Metrics

Model architecture adopted from [8]