CoVariance Imaging - Final Report

Vlad Kesler, Alyssa Cartwright, Nicholas Vitale
vkesler@stanford.edu, apcart@stanford.edu, nhvitale@stanford.edu

Abstract

We have designed and characterized a convolutional neural network to classify chest X-rays as COVID positive or negative. Our final model, informed by CheXNet, makes use of a DenseNet-121 architecture [1]. We compare the performance of a model that is not pre-trained and a model that applies transfer learning to the CheXNet learned weights. Our transfer learning model achieved a train/test/val accuracy of 99.8/90.0/86.2 while the fully trained model achieved an accuracy of 99.1/86.3/86.2. Further analysis of recall, precision, f1-score, and area under receiver-operator curve metrics for these two models are presented later in this work.

1 Introduction

In the era of SARS-CoV-2 (COVID-19), rapid diagnosis of COVID-19 is critical to maximize patient throughput in hospitals where real-time polymerase chain reaction (RT-PCR) based testing may be limited. This project aims to use chest X-rays, an ubiquitous and inexpensive diagnostic tool, to distinguish SARS-CoV-2 infected lungs from other pneumonias and healthy lungs in order to better inform caretakers. Our model accepts chest X-rays and classifies them as COVID positive or negative.

1.1 Related Work

There is a wealth of previous work on automated classification of chest X-rays or CT images to identify tumors, score segmentation, or even identify different diseases[2]. Of specific note to our project, CheXNet exemplifies the current state-of-the-art image classification for X-ray images[1], which implements an extremely deep convolutional neural network (CNN) to recognize low and high level features of X-ray images and is able to diagnose patients based on the images as well as a human radiologist[3,4]. With the emergence of COVID-19, several researchers have begun applying neural networks, including the Inception transfer learning model [5] and other deep convolutional neural network models such as VGG19 and Dense Convolutional Network (DenseNet) models [6].

2 Dataset and Features

Our datasets were pulled from open source datasets and designed to achieve an 80-10-10 split between training, dev, and test sets, with dev and test sets having a 50-50 split of COVID and non-COVID[7, 8, 9]. The final breakdown of datasets is described in Table 1. Some images are from longitudinal trails depicting the progression of COVID-19, and in many cases, early images are marked as negative which introduces some user bias into the dataset. Additionally, many X-rays are off-centered, rotated slightly, have radiologist markings, or are even in the supline plane which introduces significant image noise which could impact training. Finally, the imbalance of the dataset poses a major issue, as the model will generally be able to reduce cost by labeling all images as negative.

CS230: Deep Learning, Spring 2020, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

Dataset	COVID	Non-COVID
Train	170	871
Dev	65	65
Test	65	65
Total	300	1001

Table 1: Dataset Distribution

3 Methods

We evaluated the performance of several different architectures on this task, including a baseline CNN, ResNet18, and DenseNet-121. Performance was evaluated by generating confusion matrices, calculating metrics (such as recall, precision, specificity, and accuracy), examining intermediate layer activations, and comparing heat maps of specific classified images. Much of our model design was intended to mitigate the prevalence of non-COVID images and the scarcity of positive examples in our training set that led to significant over-fitting in early models. We will go over some our early approaches and end with our final approach which was inspired by CheXNet.

3.1 Baseline CNN

Our first approach was with a baseline model (See Appendix A), which was built from a Keras binary CNN classifier tutorial and modified to classify our dataset [10]. Images were normalized and resized to 640x640 before being passed as input. The CNN was trained with an Adam optimizer where the initial $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999$ with minibatches of 32 images and ran for 15 epochs. Baseline training accuracy was 99% and test accuracy reached 70%. The high accuracy on the training set indicated some type of over-fitting or that the impact of data imbalance was dominating. In order to address this, we next explored employing regularization, manual weighting, and data up-sampling and generated confusion matrices for different conditions to evaluate the success of these tactics.

We primarily trained 4 different models with varying types of modifications made to the data and model itself. Out goal was to identify how these modifications would impact the models' performance and address the imbalance issue present in the data-set. Each model was designed as follows:

- **Model 1:** Original baseline model configured with 83/17 split (Non-COVID to COVID) for the training set while the test set was split 50/50.
- **Model 2:** Configured with the same data split configuration, but dropout layers were integrated after each CONV-POOL pair as well as a batch normalization (BN) layer at the input to reduce image variance.
- **Model 3:** Inherited droupout and BN from Model 2 and integrated two additional features. First, class weighting was manually specified to be 1:6.12 (Non-COVID to COVID), the inverse of the ratio of COVID to Non-COVID examples in the training set.
- **Model 4:** Inherited the structure of Model 3, but used an up-sampled training set with a 50:50 split of COVID:Non-COVID images generated with random image duplication.

Over the four models we computed the precision, recall, confusion matrix results, and accuracy. The exact results can be found in **Appendix B**. However, the main takeaway was that Models 3 and 4 demonstrated reduction in over-fitting and increased generalization from the training set to the test/val datasets. However, the in the best case scenario our models had an accuracy performance ceiling of 70% and recall below 50%. This is not desirable in clinical settings and thus we set off in efforts of exploring a more sophisticated model.

3.2 ResNet-18

Our first attempt to increase network complexity was to make use of the ResNet-18 architecture [11]. As in the initial CNN, images were normalized and sized to 640x640 and mini-batch size was 32. SGD optimization was used with a learning rate of 0.001 and momentum of 0.9. After 25 training epochs, the validation accuracy reached 65%. We decided not to pursue this approach as we believed that we would achieve better success using DenseNet-121, a more complex architecture that has been demonstrated to successfully classify chest X-rays in CheXNet[1].

3.3 DenseNet-121

Our final model was settled on by manipulating the CheXNet DenseNet121 structure which is shown in Figure 1 [12, 1].

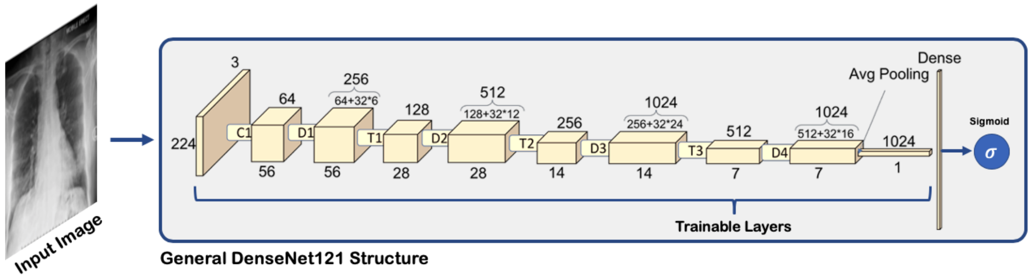


Figure 1: General structure of a the DenseNet121 with a sigmoid binary classifier layer attached at the end after the last Dense fully connected layer.

The weights from CheXNet were imported from a saved 14-class model state and then loaded into TorchVision's DenseNet121 template[13]. The 14-label classifier layer was stripped and replaced with a Sigmoid layer to convert the multi-classifier structure into a binary classifier. We tested 11 different models where the numbers of frozen layers (ie. un-trainable layers) were varied to see what effects it had on performance and training time [1]. Refer to **Appendix C** for a general work-flow overview. On each iteration, the DenseNet model would be recompiled with a new frozen layer domain. The model was then trained with mini-batches of 16 320x320 size images for 25 epochs. Additionally, a class weighting scheme of 6.12:1 (positive:negative) was applied to curtail to influence of an imbalanced dataset.

4 Results

The DenseNet121's performance for each model was analyzed extensively. Metrics were computed for each class using the sklearn open-source library for the train, test, and val data sets (See **Appendix D** for all reported metrics for each of the 11 models). Figure 2 depicts the computed precision, recall, f1-score, area under receiver-operator curve (AUROC), and accuracy assuming the COVID label as the positive class for each of the data sets. It is important to note that the frozen layers start at the beginning of the neural network and work forward into deeper layers.

After computing performance benchmarks for each model, we looked at some select activation heatmaps from the final fully connected layer. The model we chose for heatmap generation had around 311 trainable layers while the first 51 layers were frozen in the transfer learning model adopted from CheXNet. This model version, as well as the completely retrained model (362 trainable layers), showed the highest performance while demonstrating good generalization and immunity to over-fitting. In Figures 3 and 4 depict different heatmaps for two correctly labeled COVID cases and two correctly labeled Non-COVID cases. **Appendix E** shows some select activation outputs from the Dense Layers in the DenseNet for Models 10 and 11 on a COVID positive image.

Out of the models we explored, the modified pretrained CheXNet model demonstrated superior performance. Based on the performance metrics, the final two models of the modified CheXNet models performed the best. Model 10 has 311 trainable layers while Model 11 is essentially just the DenseNet121 template completely retrained. Table 2 compares the metrics for both of these models.

The results show that there is greater agreement/generalization between datasets and a better balance of importance to precision and recall for Model 11. Model 10 demonstrates excellent performance, but there is some deviation between accuracy and a bit of bias towards precision among the datasets. Ideally, the favorable model is one that gives equal balance to both precision and recall as it is important to identify COVID positive images and not label negative cases as positive. Although Table 2 reports fantastic results, the heatmaps pose an interesting problem.

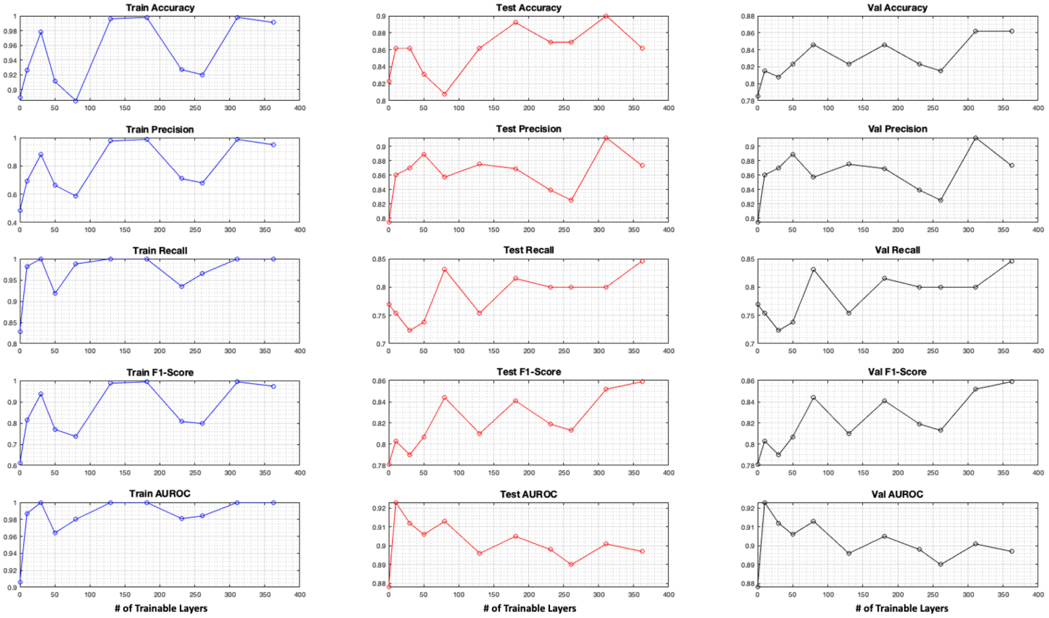


Figure 2: Metric results for Train (blue), Test (red), and Validation (black) sets against the number of trainable layers.

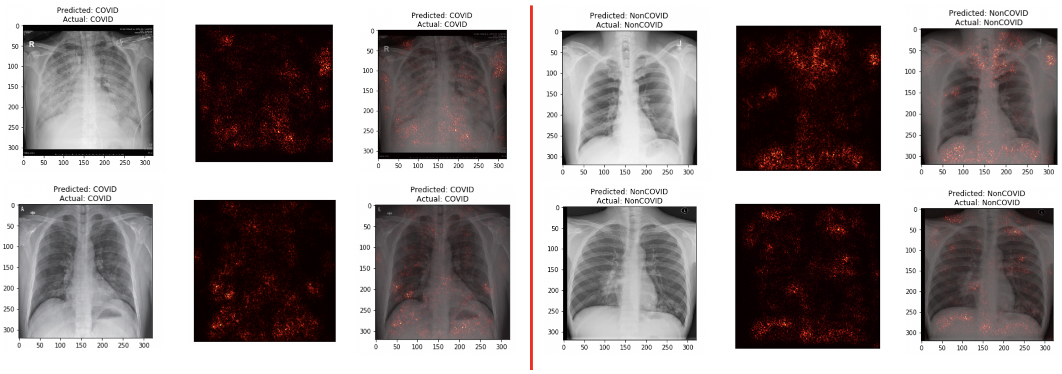


Figure 3: Example heatmap overlays for two COVID positive and two Non-COVID examples. The original image and its heatmap are shown individually and then combined to depict areas of interest.

5 Discussion

As observed in the Results section, Models 10 and 11 perform the best with slight difference between them. One notable characteristic of Model 10 is that it took 5 minutes less to train than Model 11. This might seem insignificant, but when scaling up to larger datasets, which should be available as COVID progresses, applying a transfer learning model where layers do not have to be trained does cut down on the train time. Decreased train time increases the potential for more model prototypes in a shorter amount of time thus increasing the community's ability to identity the absolute superior method of detecting COVID in X-ray images.

On the other hand, Model 11 has extremely tight deviations between test sets when considering recall, precision, and accuracy. This indicates that retraining the entire DenseNet does provide some leverage. One explanation might be attributed to the space of examples that the pretrained CheXNet is optimized for. Although it seems that COVID images would be inside the set space that CheXNet operates in, the space of discerning COVID positive lung infection versus other lung infection could be a set space that is just slightly outside of CheXNet's. This could explain the slight increase in generalization and overall performance when retraining the entire DenseNet. It does raise the

		Model 10	**Model 11**
Accuracy	**Train**	99.8	99.1
	Test	90.0	86.3
	Val	86.2	86.2
Precision	**Train**	98.8	95.0
	Test	93.3	87.3
	Val	91.2	86.3
Recall	**Train**	100	100
	Test	86.2	84.6
	Val	80.0	85.6
F1-Score	**Train**	99.4	97.4
	Test	89.6	85.6
	Val	85.2	85.9
AUROC	**Train**	99.9	99.9
	Test	94.5	93.9
	Val	90.1	89.7

Table 1: Table 2: Comparison of results between the final two modified CheXNet models.

question as to what set space does the COVID image dataset reside in when comparing it to other lung infection datasets.

Another discussion worth having is: what do the heatmaps show? Many researchers not only display the numerical results of their neural network, but also present heatmaps that reflect what the neural network is "looking at". Looking back at our heatmaps, there does not seem to be any patterns of focus in the heatmaps when comparing negative vs positive examples. This observation was consistent over all datasets. While there is some activation near the opaque tracts where the lung infection resides, the network seems to heavily activate on the high intensity areas of the image like fatty tissue. This raises the question as to what unique features the network is exactly identifying when it labels a positive example correctly. The odd nature of the heatmaps could indicate that maybe there are very subtle features present the network is picking up on and visualising them requires more data processing of the activation maps in general.

This work shows promise for the use of neural nets in the identification of COVID positive patients. One of the greatest challenges we faced in this work was the scarcity of positive examples. Despite our limited amount of data, we were pleased to see our classifier could perform quite well. As additional data is made available, this model can be better evaluated in regard to its clinical usability and model architecture and hyperparameters can be better designed.

6 Contributions

All group members contributed to discussions about project direction and project reports. Action items in implementation were discussed and split between group members as follows.

All of our code can be found at our github repo [14]. All images used and CheXNet model code can be found at their respective github repos [8,13].

- **Vlad:** Action item manager, metric architecture implementation, model building/modification
- **Alyssa:** Document organizer, heatmap and activation map creation, model building/modification
- **Nick:** Figure development, AWS management, CheXNet importing, model building/modification

  A special thanks to our project TA, **Shubhang Desai**, as well as **Pranav Rajpurkar** for all of their help and insightful suggestions as we tackled this project.

References

[1] Deep learning for chest radiograph diagnosis: A retrospective comparison of the CheXNeXt algorithm to practicing radiologists Rajpurkar, Pranav, Irvin, Jeremy, Ball, Robyn L, Zhu, Kaylie, Yang, Brandon, Mehta,

Hershel, Duan, Tony, Ding, Daisy, Bagul, Aarti, Langlotz, Curtis P, and others. PLOS Medicine, Public Library of Science, 15, (11), pages e1002686, 2018.

[2] Learning to read chest X-ray images from 16000+ examples using CNN: Dong, Yuxi, Yuchao Pan, Jun Zhang, and Wei Xu. In 2017 IEEE/ACM International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE), pp. 51-57. IEEE, 2017.

[3] CheXpert: A Large Chest Radiograph Dataset with Uncertainty Labels and Expert Comparison Irvin, Jeremy, Rajpurkar, Pranav, Ko, Michael, Yu, Yifan, Ciurea-Ilcus, Silviana, Chute, Chris, Marklund, Henrik, Haghgoo, Behzad, Ball, Robyn, Shpanskaya, Katie, and others. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, 2019.

[4] CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning Rajpurkar, Pranav, Irvin, Jeremy, Zhu, Kaylie, Yang, Brandon, Mehta, Hershel, Duan, Tony, Ding, Daisy, Bagul, Aarti, Langlotz, Curtis, Shpanskaya, Katie, and others. arXiv preprint arXiv:1711.05225, 2017.

[5] A deep learning algorithm using CT images to screen for Corona Virus Disease (COVID-19): Wang, Shuai, Bo Kang, Jinlu Ma, Xianjun Zeng, Mingming Xiao, Jia Guo, Mengjiao Cai et al. medRxiv (2020).

[6] Covidx-net: A framework of deep learning classifiers to diagnose covid-19 in x-ray images: Hemdan, Ezz El-Din, Marwa A. Shouman, and Mohamed Esmail Karar. arXiv preprint arXiv:2003.11055 (2020).

[7] Joseph Paul Cohen and Paul Morrison and Lan Dao COVID-19 image data collection, arXiv:2003.11597, 2020 https://github.com/ieee8023/covid-chestxray-dataset

[8] https://github.com/aildnont/covid-cxr

[9] Kermany, Daniel; Zhang, Kang; Goldbaum, Michael (2018), "Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images for Classification", Mendeley Data, v2

[10] https://colab.research.google.com/github/google/eng-edu/blob/master/ml/pc/exercises/image-classification-part1.ipynb

[11] Deep Residual Learning for Image Recognition, Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. arXiv: 1512.03385 (2015).

[12] Densely Connected Convolutional Networks, Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. arXiv: 1608.06993 (2016).

[13] https://github.com/arnoweng/CheXNet

[14] https://github.com/nhv3/Master-Control-230

7 Appendix

7.1 Appendix A

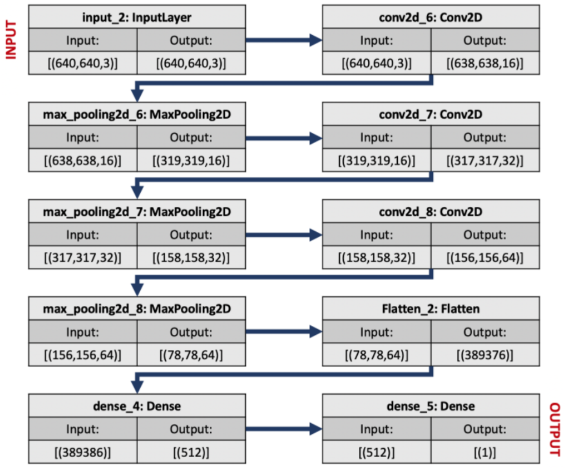
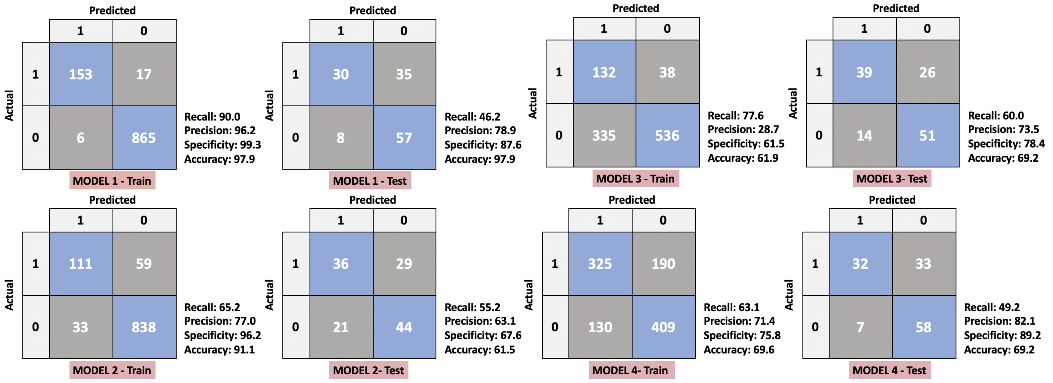


Figure 4: Baseline CNN Architecture: This architecture employs 3 2D convolution-max pooling blocks, a flattening layer, and two dense layers before the output.

7.2 Appendix B



MODEL 1 - Train

Actual \ Predicted	1	0
1	153	17
0	6	865

Recall: 90.0
Precision: 96.2
Specificity: 99.3
Accuracy: 97.9

MODEL 1 - Test

Actual \ Predicted	1	0
1	30	35
0	8	57

Recall: 46.2
Precision: 78.9
Specificity: 87.6
Accuracy: 97.9

MODEL 3 - Train

Actual \ Predicted	1	0
1	132	38
0	335	536

Recall: 77.6
Precision: 28.7
Specificity: 61.5
Accuracy: 61.9

MODEL 3- Test

Actual \ Predicted	1	0
1	39	26
0	14	51

Recall: 60.0
Precision: 73.5
Specificity: 78.4
Accuracy: 69.2

MODEL 2 - Train

Actual \ Predicted	1	0
1	111	59
0	33	838

Recall: 65.2
Precision: 77.0
Specificity: 96.2
Accuracy: 91.1

MODEL 2- Test

Actual \ Predicted	1	0
1	36	29
0	21	44

Recall: 55.2
Precision: 63.1
Specificity: 67.6
Accuracy: 61.5

MODEL 4- Train

Actual \ Predicted	1	0
1	325	190
0	130	409

Recall: 63.1
Precision: 71.4
Specificity: 75.8
Accuracy: 69.6

MODEL 4 - Test

Actual \ Predicted	1	0
1	32	33
0	7	58

Recall: 49.2
Precision: 82.1
Specificity: 89.2
Accuracy: 69.2

Figure 5: Generated confusion matrices for Models 1-4: While regularization, up-weighting, and up-sampling reduced overfitting, performance limits suggested that a more complex architecture would be appropriate.

7.3 Appendix C

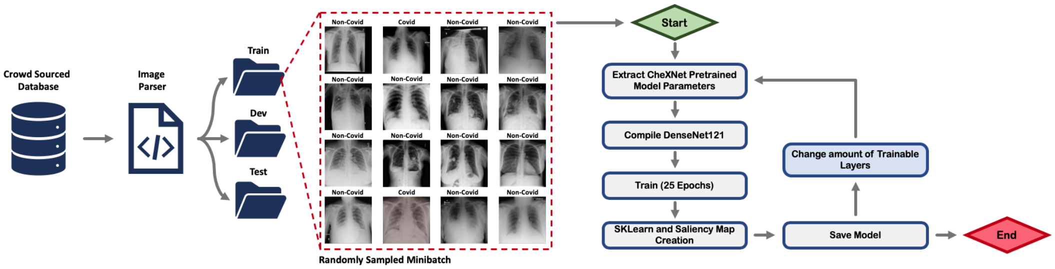
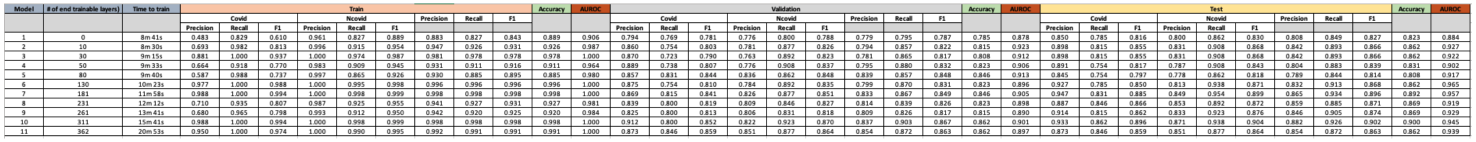


Figure 6: General work flow when training various DenseNet121 model flavors.

7.4 Appendix D



Model	# of end trainable layers)	Time to train	Train									Accuracy	AUROC
			Covid			Ncovid			Precision	Recall	F1		
			Precision	Recall	F1	Precision	Recall	F1					
1	0	8m 41s	0.483	0.829	0.610	0.961	0.827	0.889	0.883	0.827	0.843	0.889	0.906
2	10	8m 30s	0.693	0.982	0.813	0.996	0.915	0.954	0.947	0.926	0.931	0.926	0.987
3	30	9m 15s	0.881	1.000	0.937	1.000	0.974	0.987	0.981	0.978	0.978	0.978	1.000
4	50	9m 33s	0.664	0.918	0.770	0.983	0.909	0.945	0.931	0.911	0.916	0.911	0.964
5	80	9m 40s	0.587	0.988	0.737	0.997	0.865	0.926	0.930	0.885	0.895	0.885	0.980
6	130	10m 23s	0.977	1.000	0.988	1.000	0.995	0.998	0.996	0.996	0.996	0.996	1.000
7	181	11m 58s	0.988	1.000	0.994	1.000	0.998	0.999	0.998	0.998	0.998	0.998	1.000
8	231	12m 12s	0.710	0.935	0.807	0.987	0.925	0.955	0.941	0.927	0.931	0.927	0.981
9	261	13m 41s	0.680	0.965	0.798	0.993	0.912	0.950	0.942	0.920	0.925	0.920	0.984
10	311	15m 41s	0.988	1.000	0.994	1.000	0.998	0.999	0.998	0.998	0.998	0.998	1.000
11	362	20m 53s	0.950	1.000	0.974	1.000	0.990	0.995	0.992	0.991	0.991	0.991	1.000

Model	# of end trainable layers)	Time to train	Validation									Accuracy	AUROC
			Covid			Ncovid			Precision	Recall	F1		
			Precision	Recall	F1	Precision	Recall	F1					
1	0	8m 41s	0.794	0.769	0.781	0.776	0.800	0.788	0.779	0.795	0.787	0.785	0.878
2	10	8m 30s	0.860	0.754	0.803	0.781	0.877	0.826	0.794	0.857	0.822	0.815	0.923
3	30	9m 15s	0.870	0.723	0.790	0.763	0.892	0.823	0.781	0.865	0.817	0.808	0.912
4	50	9m 33s	0.889	0.738	0.807	0.776	0.908	0.837	0.795	0.880	0.832	0.823	0.906
5	80	9m 40s	0.857	0.831	0.844	0.836	0.862	0.848	0.839	0.857	0.848	0.846	0.913
6	130	10m 23s	0.875	0.754	0.810	0.784	0.892	0.835	0.799	0.870	0.831	0.823	0.896
7	181	11m 58s	0.869	0.815	0.841	0.826	0.877	0.851	0.833	0.867	0.849	0.846	0.905
8	231	12m 12s	0.839	0.800	0.819	0.809	0.846	0.827	0.814	0.839	0.826	0.823	0.898
9	261	13m 41s	0.825	0.800	0.813	0.806	0.831	0.818	0.809	0.826	0.817	0.815	0.890
10	311	15m 41s	0.912	0.800	0.852	0.822	0.923	0.870	0.837	0.903	0.867	0.862	0.901
11	362	20m 53s	0.873	0.846	0.859	0.851	0.877	0.864	0.854	0.872	0.863	0.862	0.897

Model	# of end trainable layers)	Time to train	Test									Accuracy	AUROC
			Covid			Ncovid			Precision	Recall	F1		
			Precision	Recall	F1	Precision	Recall	F1					
1	0	8m 41s	0.850	0.785	0.816	0.800	0.862	0.830	0.808	0.849	0.827	0.823	0.884
2	10	8m 30s	0.898	0.815	0.855	0.831	0.908	0.868	0.842	0.893	0.866	0.862	0.927
3	30	9m 15s	0.898	0.815	0.855	0.831	0.908	0.868	0.842	0.893	0.866	0.862	0.922
4	50	9m 33s	0.891	0.754	0.817	0.787	0.908	0.843	0.804	0.883	0.839	0.831	0.902
5	80	9m 40s	0.845	0.754	0.797	0.778	0.862	0.818	0.789	0.844	0.814	0.808	0.917
6	130	10m 23s	0.927	0.785	0.850	0.813	0.938	0.871	0.832	0.913	0.868	0.862	0.965
7	181	11m 58s	0.947	0.831	0.885	0.849	0.954	0.899	0.865	0.934	0.896	0.892	0.957
8	231	12m 12s	0.887	0.846	0.866	0.853	0.892	0.872	0.859	0.885	0.871	0.869	0.919
9	261	13m 41s	0.914	0.815	0.862	0.833	0.923	0.876	0.846	0.905	0.874	0.869	0.929
10	311	15m 41s	0.933	0.862	0.896	0.871	0.938	0.904	0.882	0.926	0.902	0.900	0.945
11	362	20m 53s	0.873	0.846	0.859	0.851	0.877	0.864	0.854	0.872	0.863	0.862	0.939

Figure 7: Complete table of computed metrics for the CheXNet model modification experiment. Weighted average recall, precision, and f1-scores are provided at the end of each dataset for an average metric result.

7.5 Appendix E

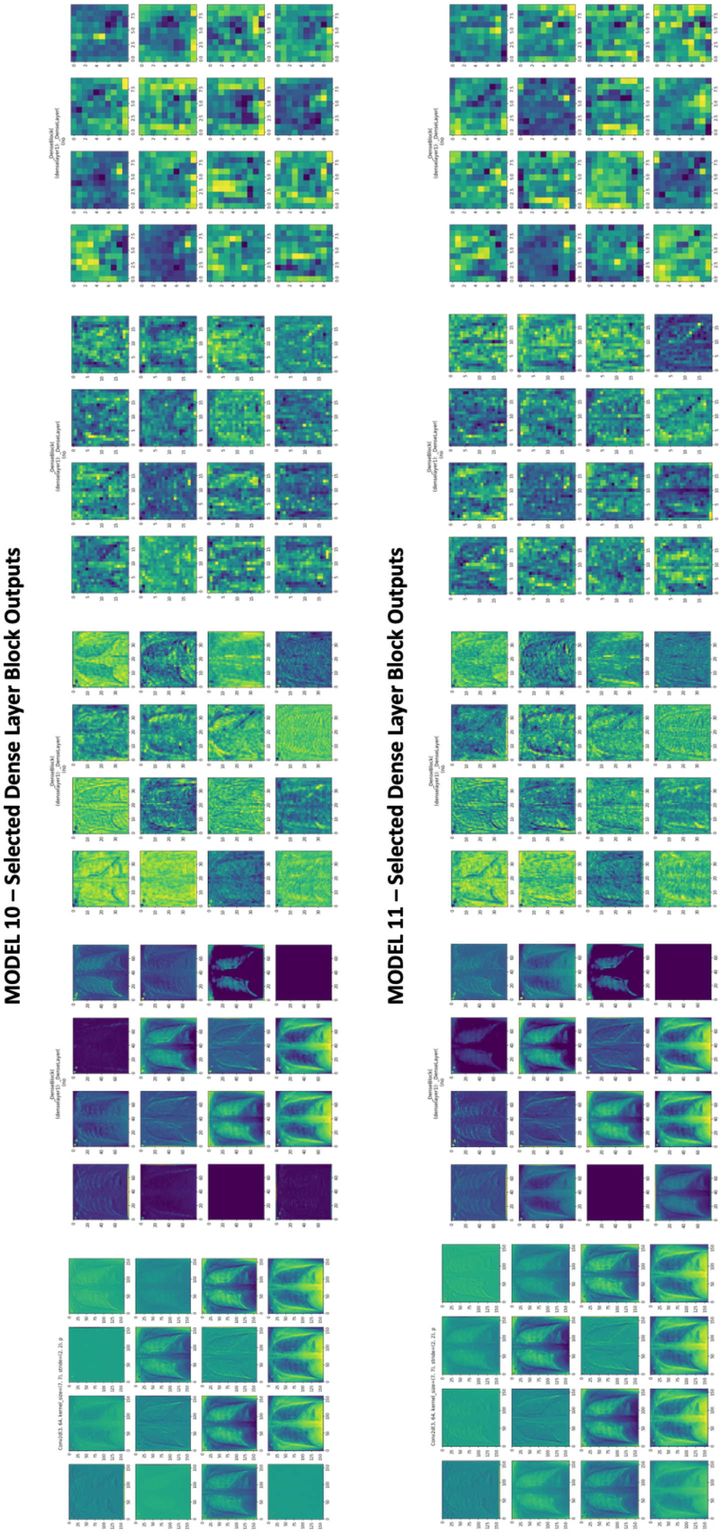

MODEL 10 – Selected Dense Layer Block Outputs
MODEL 11 – Selected Dense Layer Block Outputs
Conv2d(3, 64, kernel_size=(7, 7), stride=(2, 2), p
Conv2d(3, 64, kernel_size=(7, 7), stride=(2, 2), p

Figure 8: Selected activation maps from a few Dense Layer output blocks from Models 10 and 11 starting from the initial layers and moving up (left to right).