Udacity Machine Learning Nanodegree 2021

Capstone Project Report

Pneumonia Detection with Deep Learning

Ravee Rachakonda

Nov 2021

I. Definition

Project Overview

This project is about Pneumonia Detection based on Chest X-Ray images. In 2017, 2.5 Million people [10] died of Pneumonia. Pneumonia is lung inflammation which impacts air sacs called alveoli. Alveoli can get filled with fluid and Pus which can be very painful and difficult for breathing. Pneumonia shows up in X-Ray images and can be seen as patchy areas in chest x-ray images. Pneumonia can be caused by bacteria, viruses, other reasons like fungi, parasites, hospital environments. Pneumonia is a deadly lung disease that can be diagnosed based on investigating chest x-ray images. This project is based on a kaggle challenge [chest-xray-pneumonia](). The data provided for us by Kaggle has bacteria, virus based pneumonia infections and normal healthy x-ray images. Pneumonia results in liquid built up in Alveoli reducing the oxygen intake capacity. Pneumonia patients will need to be on ventilators with enriched oxygen for this reason. Doctors use the x-ray images for checking if the patient is suffering from Pneumonia and diagnose accordingly. Using the data set provided I want to build a model that can detect if the provided x-ray image has pneumonia or not.

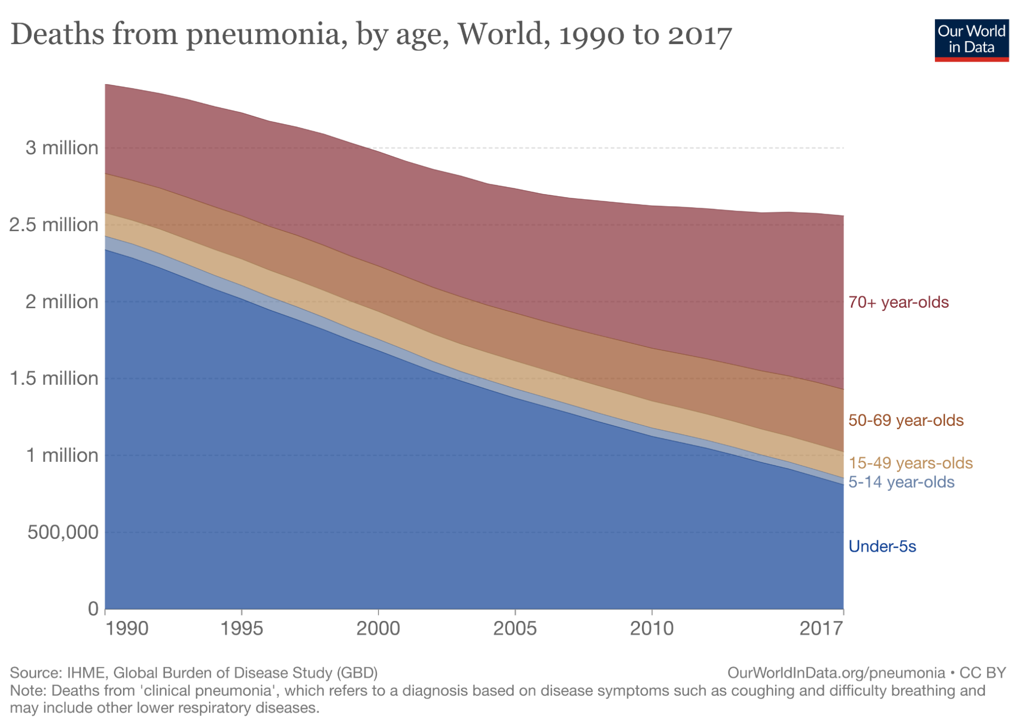


Image Source: https://ourworldindata.org/pneumonia

Problem Statement

The main goal of this capstone project is to utilize Deep Learning algorithms for detecting pneumonia in provided X-Ray. Build classifier which can score the X-Ray for detecting pneumonia and output as detected yes or no.

Metrics

Loss, Accuracy, Precision and Recall are really good metrics for such kinds of problems as the dataset is 81.921% of Pneumonia positive. We will use the confusion matrix from the predictions and check what is the recall and precision of my model.

II. Analysis

Data Exploration

The dataset used for this capstone project is from the Kaggle project. The data is a set of chest X-Ray images in jpeg format. Dataset consists of 5956 x-ray images and is categorized into 3 categories:

- Train (5216)
  - NORMAL (1341)
  - PNEUMONIA (3875)
- Test (624)
  - NORMAL (234)
  - PNEUMONIA (390)
- Val (16)
  - NORMAL (8)
  - PNEUMONIA (8)

Exploratory Visualization

Data Pre-processing

Data provided in the Kaggle project is already split into train, test and validation sets. Each set contains folders for Normal and Pneumonia categories.

1. Image processing steps should be built to read in images
2. Leverage python libraries for convert image data to matrices

Kaggle input data containing the X-Ray images is pre-organized into train, val, test and classes: NORMAL, PNEUMONIA.

I have downloaded the kaggle data set, unzipped and uploaded the data to S3 bucket under data directory. As the data processing will be done with large instances using Sagemaker, it will be handy to have the data in S3 bucket. Images are grayscale and standard size of 224x224 pixels.

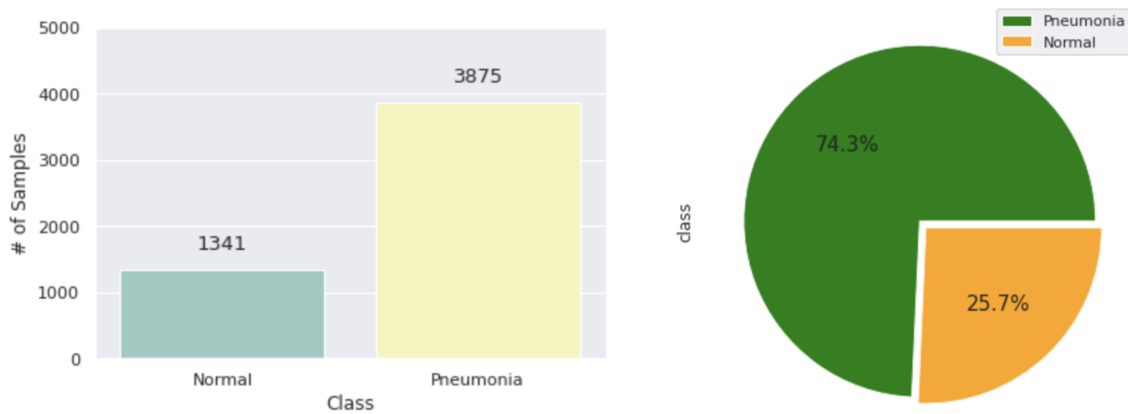


Fig 1: Training data distribution with ~3x of Pneumonia images over Normal

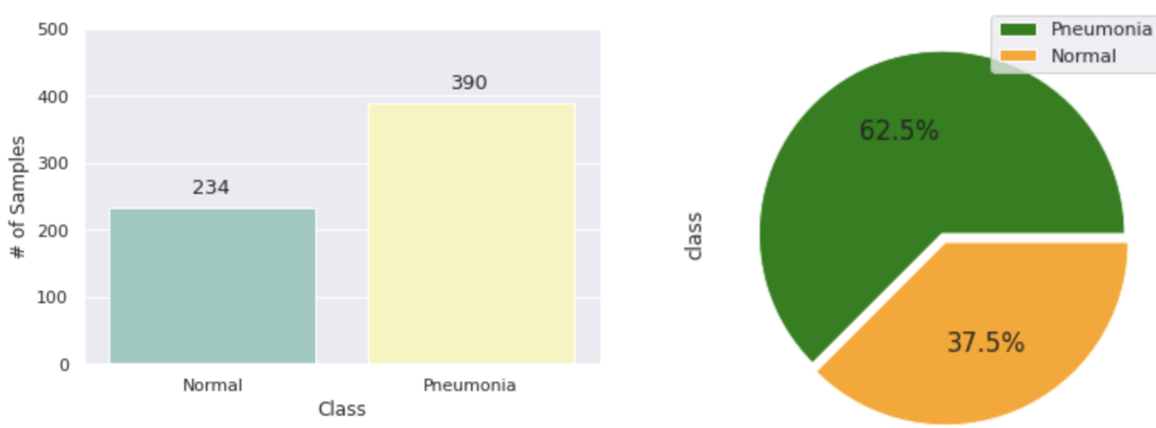
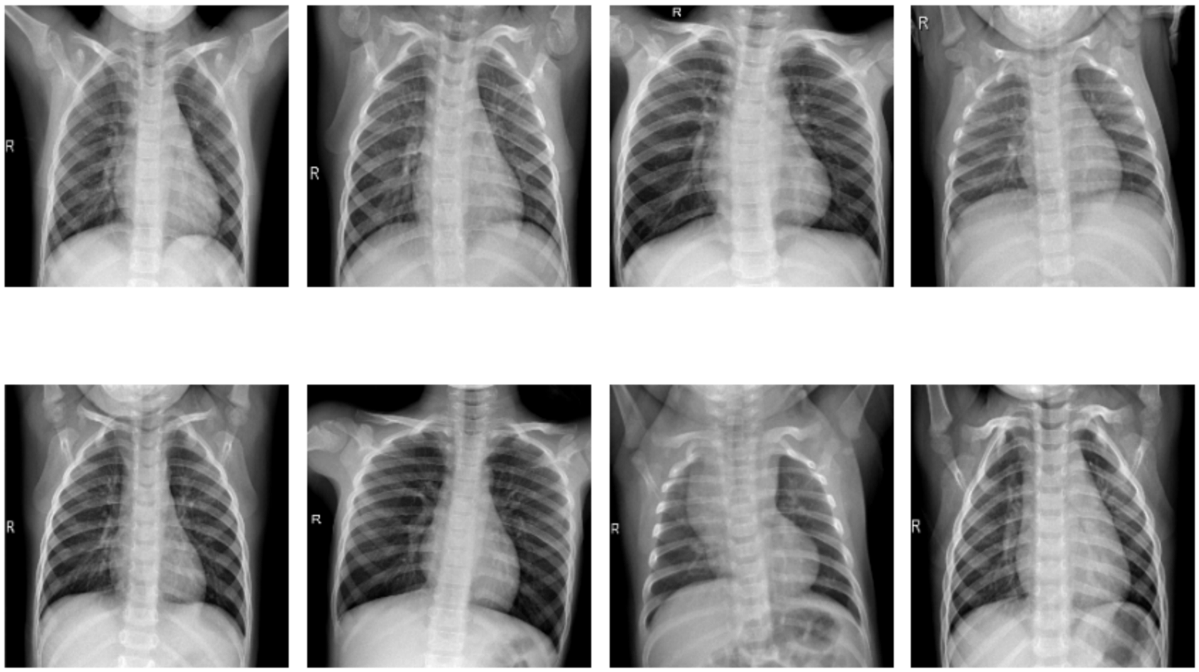


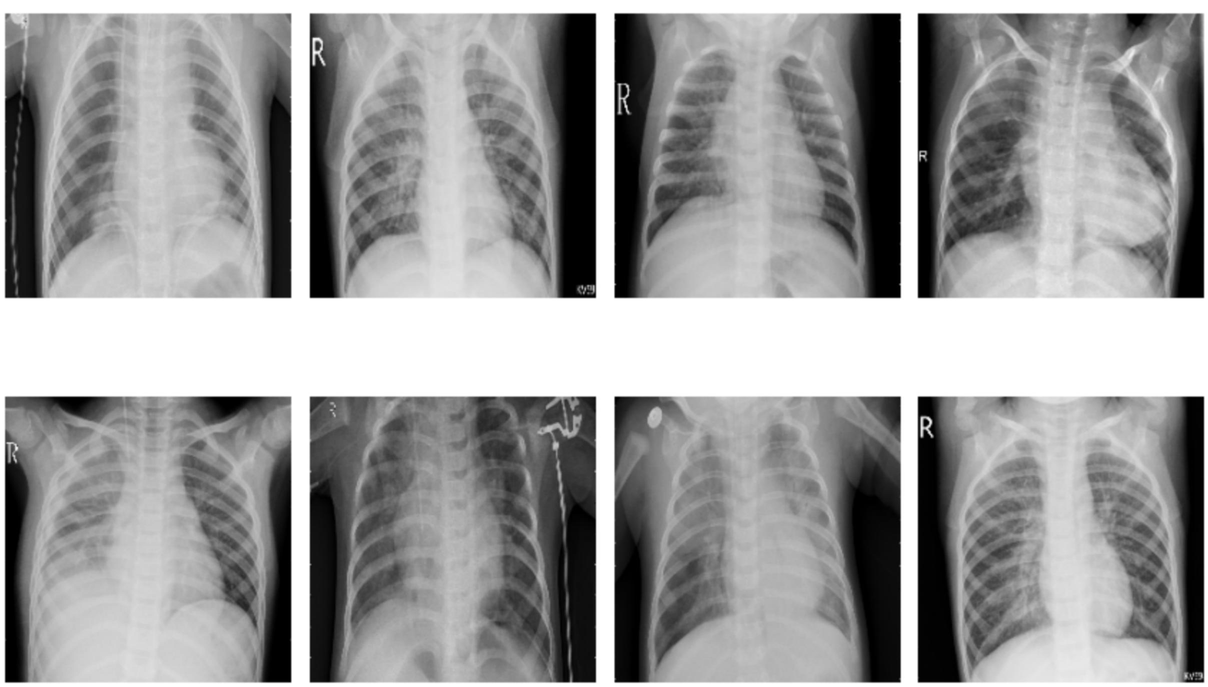
Fig 2: Test data distribution with ~1.7x of Pneumonia images over Normal

Fig 3: Visualizing the a sample of 12 x-ray images for Normal and Pneumonia

Train Set - Normal



Train Set - Pneumonia

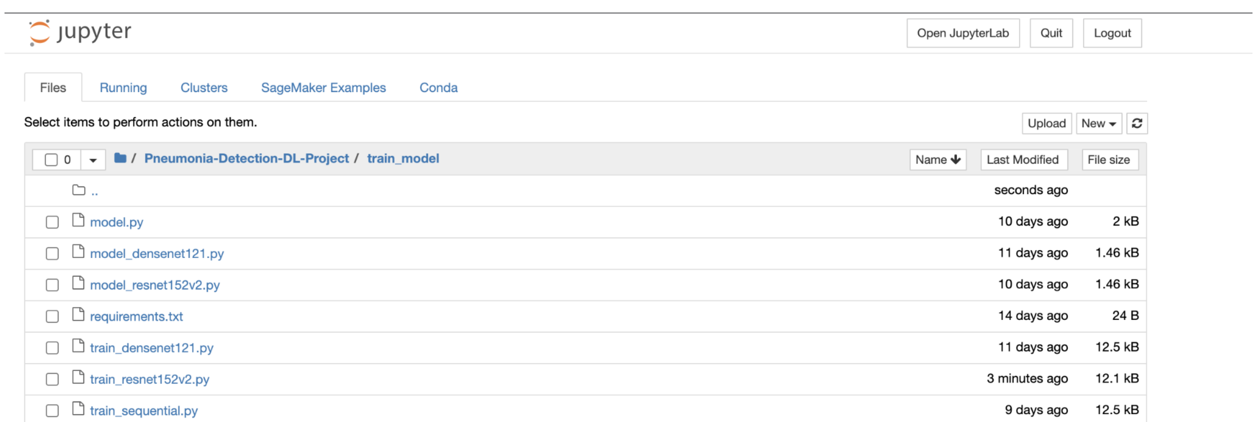


Algorithms and Techniques

We will use Deep Learning algorithms using Convoluted Neural Network (CNN) to train and build a model. I have used Tensorflow with Sagemaker for building the model. Kernel conda_tensorflow2_36 (Tensorflow 2 with Python 3.6) is used for the execution. Along with the Sequential model, I have tried transfer learning models (ResNet152V2, DenseNet121) to compare performance. Based on the performance of these algorithms, I want to choose one and tune it for better performance. ResNet152V2 & DenseNet121

A Sequential Model with 6 layers total. Image size of 224 x 224 with a depth of 3 is passed to the inputs_input layer.

Notebook initialized the Tensorflow function with calling entry_point of train_sequential.py, train_resnet152v2.py, train_densenet121.py which calls respective models for building the model.



Benchmark Model

The benchmark is used from a research paper published in cell.com (https://www.cell.com/cell/fulltext/S0092-8674(18)30154-5). In the comparison of chest X-rays presenting as pneumonia versus normal, the research team achieved an accuracy of 92.8%, with a sensitivity of 93.2% and a specificity of 90.1% operating on the same data set. Research team used a transfer learning technique with CNN that was trained on various kinds of image data (Retinal, Chest X-Rays).

III. Methodology

Data Preprocessing

Data is already split for train, test and validation sets. We will need to randomly load the corresponding matrix data for normal and pneumonia images and create a training data set. Similarly, for test and Validation sets.

DataFrame object is created by loading all the normal and pneumonia images for training into df_train. Total of 5216 rows with class [Normal, Pneumonia] and Image path.

Training (df_train): 5126 x 2 columns
Test (df_test): 624 x 2 columns

	class	image
0	Normal	./chest-xray-pneumonia/chest_xray/train/NORMAL/NORMAL2-IM-0571-0001.jpeg
1	Normal	./chest-xray-pneumonia/chest_xray/train/NORMAL/NORMAL2-IM-0799-0001.jpeg
2	Normal	./chest-xray-pneumonia/chest_xray/train/NORMAL/NORMAL2-IM-0995-0001-0002.jpeg
3	Normal	./chest-xray-pneumonia/chest_xray/train/NORMAL/IM-0704-0001.jpeg
4	Normal	./chest-xray-pneumonia/chest_xray/train/NORMAL/NORMAL2-IM-0986-0001.jpeg
...	...	...
5211	Pneumonia	./chest-xray-pneumonia/chest_xray/train/PNEUMONIA/person967_bacteria_2892.jpeg
5212	Pneumonia	./chest-xray-pneumonia/chest_xray/train/PNEUMONIA/person1340_virus_2312.jpeg
5213	Pneumonia	./chest-xray-pneumonia/chest_xray/train/PNEUMONIA/person1679_bacteria_4450.jpeg
5214	Pneumonia	./chest-xray-pneumonia/chest_xray/train/PNEUMONIA/person585_bacteria_2414.jpeg
5215	Pneumonia	./chest-xray-pneumonia/chest_xray/train/PNEUMONIA/person1343_bacteria_3414.jpeg

5216 rows × 2 columns

Using the train_test_split, I have split the training data into 80% and validation 20%. Keras ImageDataGenerator is used for processing the image data and creating the data set. Had issues with directly reading the files from S3 using flow_from_directory so leveraged the flow_from_dataframe by reading the images from the local directory on the sagemaker server.

Solution Statement

The development of convolutional neural network layers has allowed for significant gains in the ability to classify images and detect objects in a picture. These are multiple processing layers to which image analysis filters, or convolutions, are applied. The abstracted representation of images within each layer is constructed by systematically convolving multiple filters across the image, producing a feature map that is used as input to the following layer. This architecture makes it possible to process images in the form of pixels as input and to give the desired classification as output. The image-to-classification approach in one classifier replaces the multiple steps of previous image analysis methods

IV. Results

Model Evaluation and Validation

Testing the machine learning models with 100 epochs:

1. Sequential
2. Resnet152v2 (pretrained model)
3. DenseNet121 (pretrained model)
4. InceptionResNetV2 (pretrained model)

Working through each of the algorithms to solve the pneumonia detection problem, the goal is to identify which model is effective and do fine tuning on the model to check if we can improve performance.

I have chosen to implement a simple CNN using a Sequential algorithm with few layers. Since I am working with complex image data, I wanted to try transfer learning with models that have been trained on similar data sets like image classification. So I have chosen Resnet152V2, DenseNet121 and InceptionResnetV2 from the Keras documentation https://keras.io/api/applications/#usage-examples-for-image-classification-models. With pre-trained weights, I will run the evaluation for 100 epochs each and determine which model will perform better based on the test loss, test accuracy, sensitivity and Specificity.

Performance Metrics

Model: Sequential

This model is a Keras based CNN with 6 layers. Used Sigmoid function for the final layer to determine Normal or Pneumonia. The model is run for 100 epochs and below are the results observed.

Metrics:

	precision	recall	f1-score	support
Pneumonia (Class 0)	0.99	0.47	0.63	234
Normal (Class 1)	0.76	1	0.86	390
accuracy			0.8	624
macro avg	0.87	0.73	0.75	624
weighted avg	0.84	0.8	0.78	624

Test loss	**2.007941**
Test accuracy	**0.7980769277**

ROC_AUC	0.9173296077

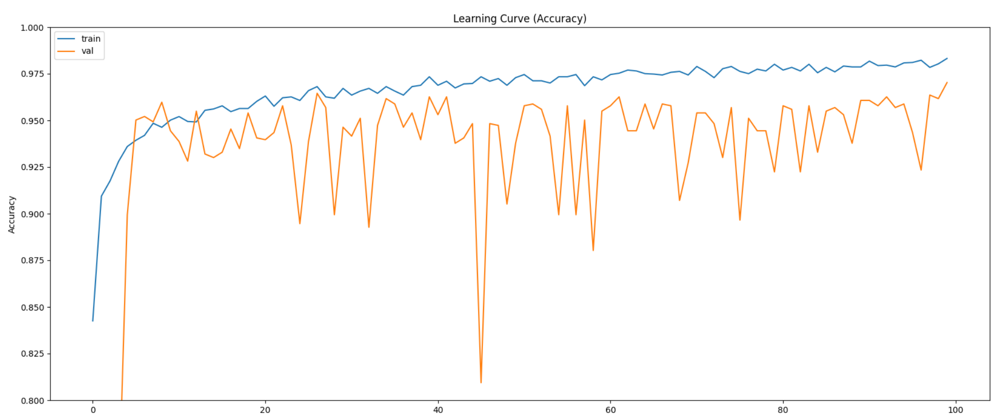


Fig 4: Learning Curve (Accuracy) - Sequential Model

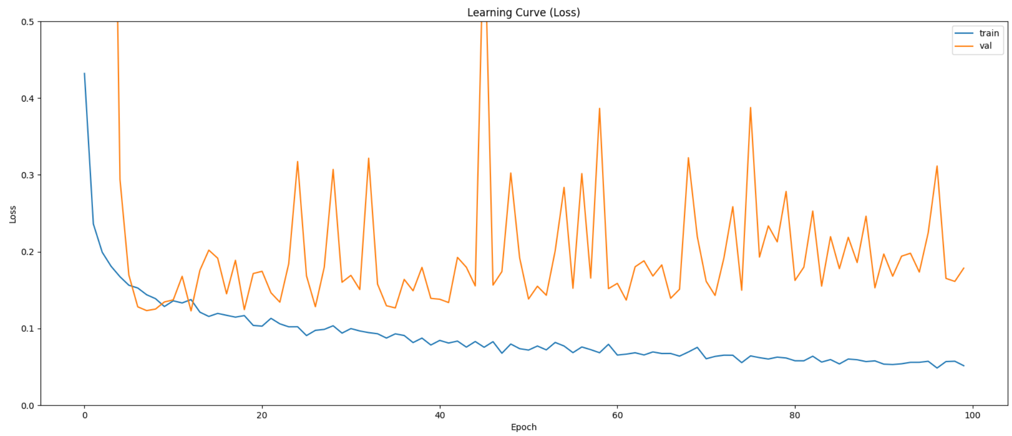


Fig 5: Learning Curve (Loss) - Sequential Model

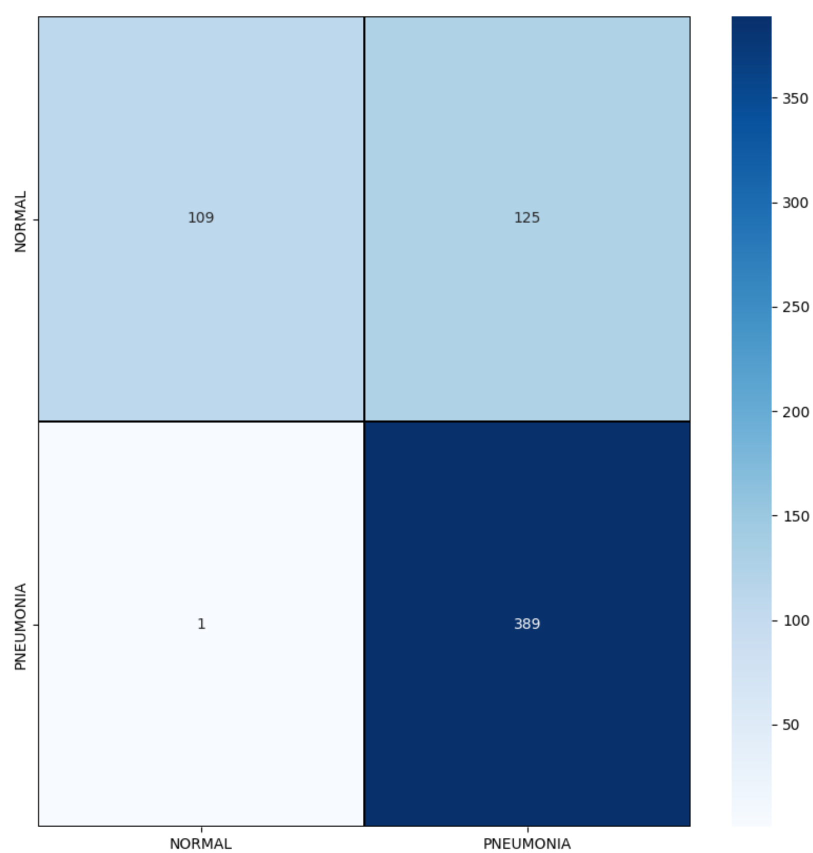


Fig 6: Confusion Matrix - Sequential Model

Sensitivity of this model is very high at 99.74% while the specificity is low at 46.5%. So most of the time model is predicting Pneumonia causing a high level of false positives.

Model: Resnet152v2 (Pre-trained)
Resnet152v2 pretrained model is used as part of transfer learning with layers locked and run for 100 epochs. The execution was very fast and the algorithm performed better than the sequential model. This is a pretrained model on image classification and used for image classification problems so leveraged for classifying the X-ray images.

Metrics:

Test loss	**0.3072773177**
Test accuracy	**0.9070512652**

	precision	recall	f1-score	support
Normal (Class 1)	0.96	0.78	0.86	234
Pneumonia (Class 0)	0.88	0.98	0.93	390
accuracy			0.54	624
macro avg	0.92	0.88	0.9	624
weighted avg	0.91	0.91	0.9	624

ROC_AUC	0.9704142012

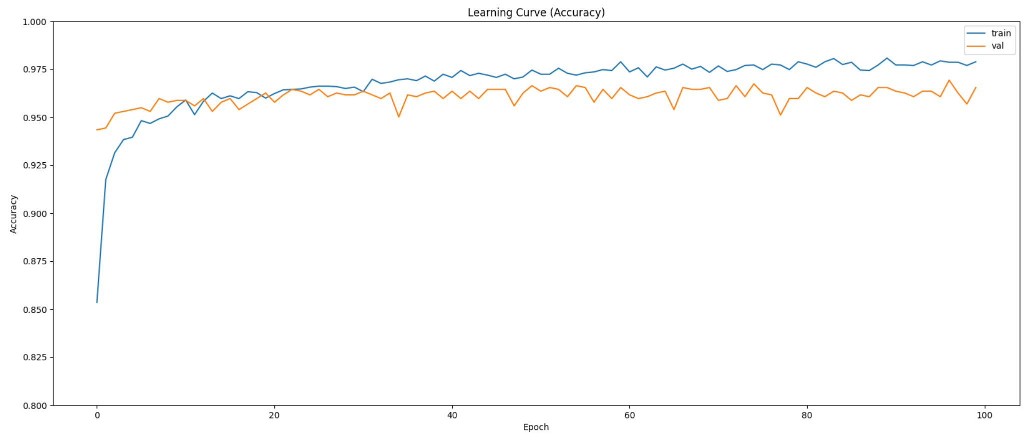


Fig 7: Learning Curve (Accuracy) - Resnet152v2 Model

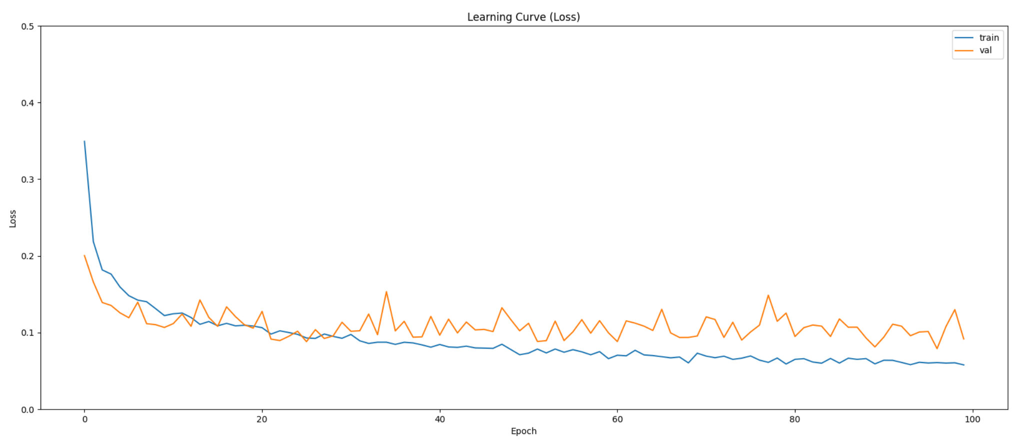


Fig 8: Learning Curve (Loss) - Resnet152v2 Model

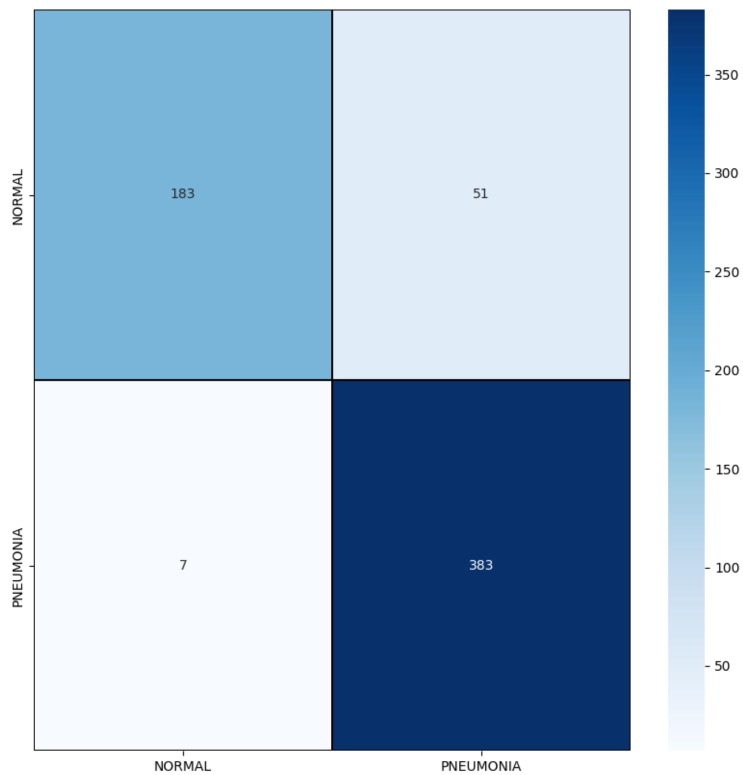


Fig 9: Confusion Matrix - Resnet152v2 Model

Model: DenseNet121 (Pre-trained)
Densenet121 pretrained model is used as part of transfer learning with layers locked and run for 100 epochs. The execution was very fast and the algorithm performed has been on par with Resnet152v2. Performance wise they are similar.

Metrics

	precision	recall	f1-score	support
Normal (Class 0)	0.96	0.78	0.86	234
Pneumonia (Class 1)	0.88	0.98	0.93	390

accuracy			0.91	624
macro avg	0.92	0.88	0.89	624
weighted avg	0.91	0.91	0.9	624

Test loss	**0.3100538471**
Test accuracy	**0.9054487348**

ROC_AUC	0.966874863

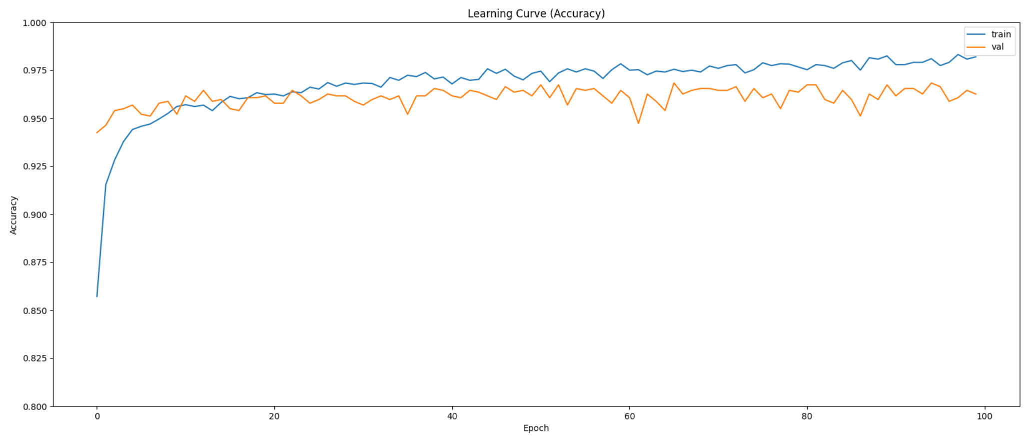


Fig 10: Learning Curve (Accuracy) - Densenet121 Model

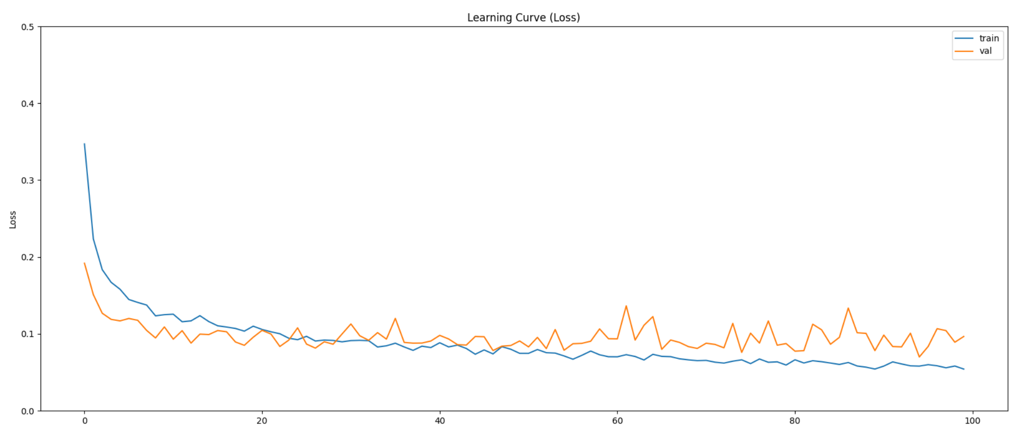


Fig 11: Learning Curve (Loss) - Densenet121 Model

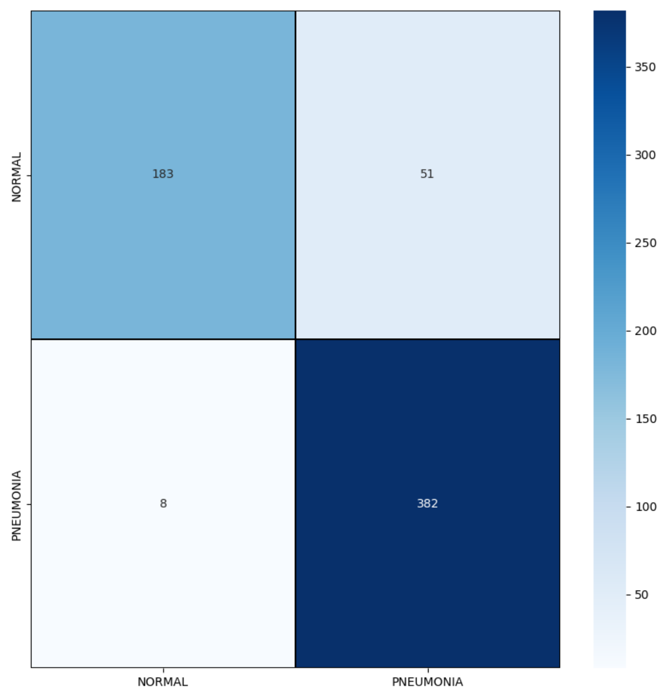


Fig 12: Confusion Matrix - Densenet121 Model

Model: InceptionResNetV2 (Pre-trained)
InceptionResNetV2 pretrained model is used as part of transfer learning with layers locked and run for 100 epochs. The execution was slower than other models used but the algorithm performed better than all other models.

Metrics:

	precision	recall	f1-score	support
Normal (Class 0)	0.93	0.79	0.85	234
Pneumonia (Class 1)	0.88	0.96	0.92	390
accuracy			0.9	624
macro avg	0.91	0.88	0.89	624
weighted avg	0.9	0.9	0.9	624

Test loss	**0.2987392356**
Test accuracy	**0.8974359035**

ROC_AUC	0.9564760026

Fig 13: Learning Curve (Accuracy) - InceptionResnetV2 Model

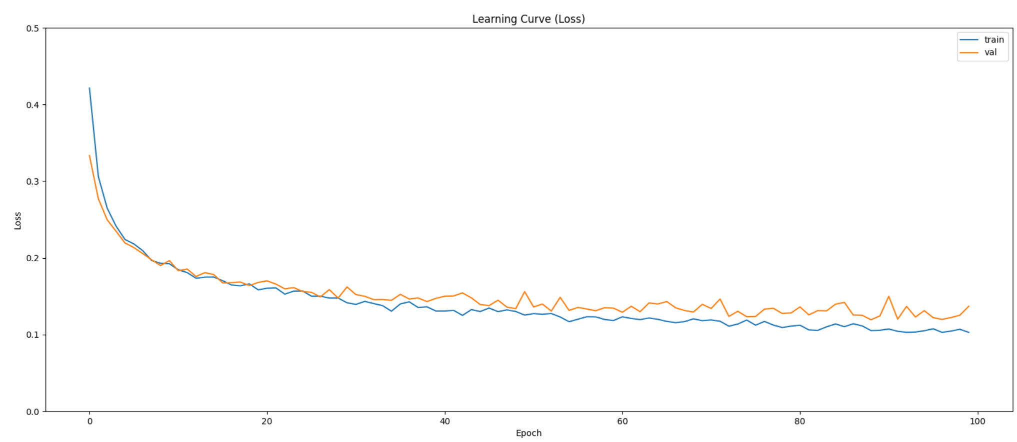


Fig 14: Learning Curve (Loss) - InceptionResnetV2 Model

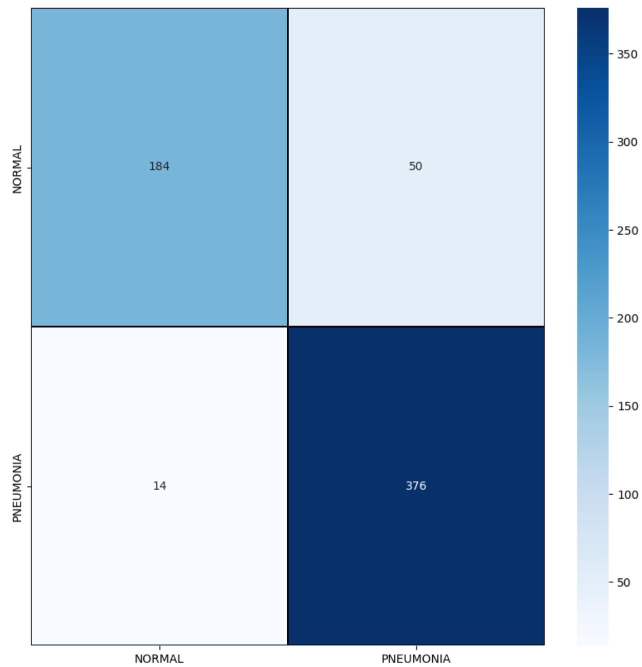


Fig 15: Confusion Matrix - InceptionResnetV2 Model

A comparison chart showing the performance of 4 algorithms.

Model	Precision (%)	Accuracy	Sensitivity (%) / Recall	Specificity (%)
Sequential	99.74	77.57	75.68	99.09
Resnet152v2	98.2	90.7	88.24	96.32
Densenet121	97.94	90.54	88.21	95.81
InceptionResnetV2	96.4	89.74	88.26	92.93

InceptionResnetV2 has better accuracy on the test set with lowest test loss. In multiple runs the test accuracy was around 88%. Sensitivity is high but unlike other models, the specificity is much higher (Couple of runs avg). Resnet152v2 also performed similar to InceptionResnetV2.

Fine Tuning:

Choosing the InceptionResnetV2, I have tuned the model by unfreezing the last 5 layers to allow learning and adjusting the weights for the problem I am trying to solve.

	precision	recall	f1-score	support
Pneumonia (Class 0)	0.97	0.84	0.9	234
Normal (Class 1)	0.91	0.98	0.95	390
accuracy			0.93	624
macro avg	0.94	0.91	0.92	624
weighted avg	0.93	0.93	0.93	624

Test loss	**0.300268496**
Test accuracy	**0.9294871688**

ROC_AUC	0.9773833005

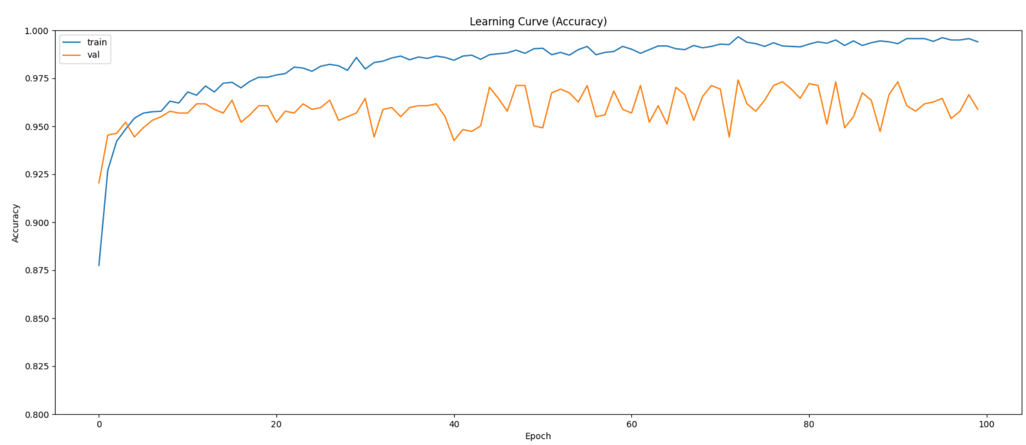


Fig 13: Learning Curve (Accuracy) - InceptionResnetV2 Tuned Model

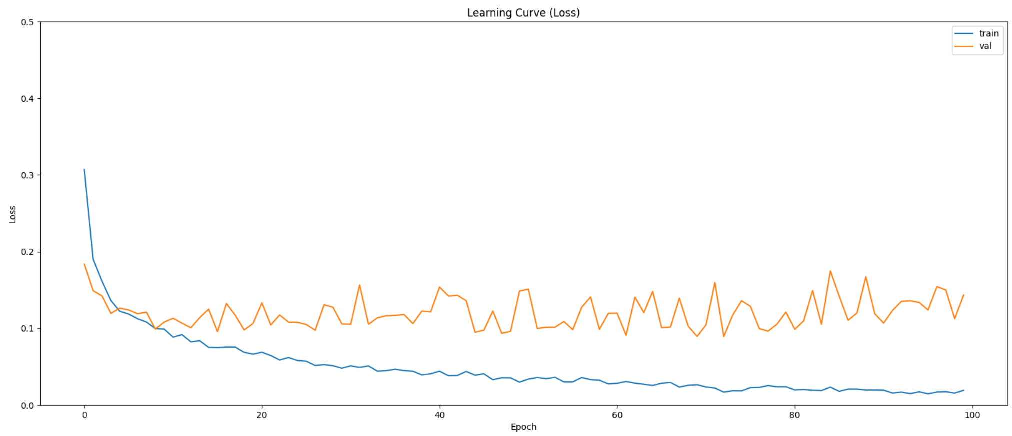


Fig 14: Learning Curve (Loss) - InceptionResnetV2 Tuned Model

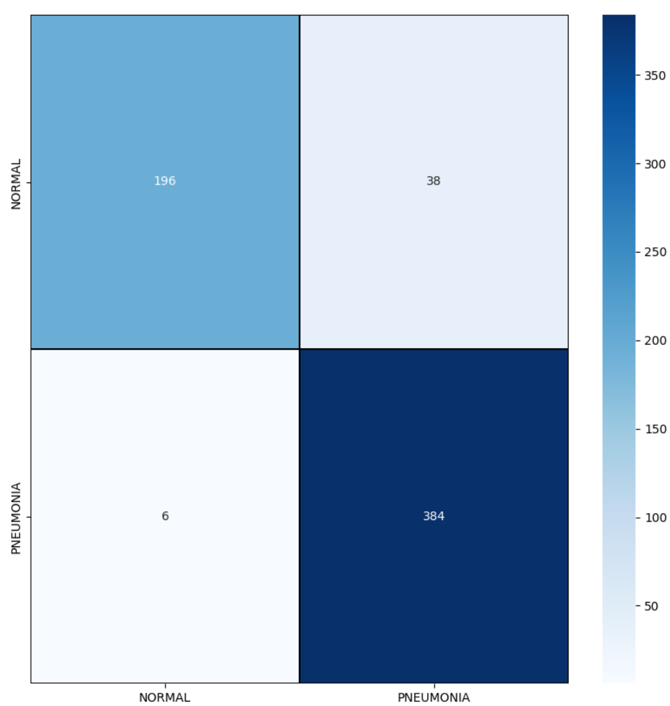


Fig 15: Confusion Matrix - InceptionResnetV2 Tuned Model

Model	Precision (%)	Accuracy	Sensitivity (%) / Recall	Specificity (%)
Sequential	99.74	77.57	75.68	99.09
Resnet152v2	98.2	90.7	88.24	96.32
Densenet121	97.94	90.54	88.21	95.81
InceptionResnetV2	96.4	89.74	88.26	92.93
InceptionResnetV2-Tuned	98.46	92.95	90.99	97.03

InceptionResnetV2 tuned model's test accuracy is higher than the pre-trained model with a sensitivity of 91% and specificity of 97.03%.

Justification

The InceptionResNetv2 model has a high accuracy rate when tested on test images. Sensitivity and Specificity are on par with the benchmark chosen. Further tuning the model by unfreezing the last layers for training on the pretrained model (Resnet152v2) helped achieve better sensitivity and specificity. The InceptionResnetV2 has 572 layers which I think helped with better learning and capturing the complex patterns of images. Without tuning the InceptionResnetV2 was performing better than other algorithms but with tuning and probably longer training we should be able to achieve higher.

Model	Test loss	Test accuracy	ROC_AUC	Precision (%)	Accuracy	Sensitivity (%) / Recall	Specificity (%)
Sequential	2.008	0.7981	0.9173	99.74	77.57	75.68	99.09
Resnet152v2	0.3073	0.9071	0.9704	98.2	90.7	88.24	96.32
Densenet121	0.3100	0.9054	0.9669	97.94	90.54	88.21	95.81
InceptionResnetV2	0.2987	0.8974	0.9564	96.4	89.74	88.26	92.93
InceptionResnetV2-Tuned	0.2771	0.9151	0.9638	98.46	92.95	90.99	97.03

V. Conclusion

Reflection

With few layers unfreezed the performance of the model has gone beyond the benchmark. One of the problems I have observed is with the provided dataset which is heavy on Pneumonia x-ray images over the normal ones. The data sets for many of the pre-trained models have been on closer sets like 5000 Normal images vs 5000 Pneumonia or Covid images. That helped with better training and predictions in my opinion. With given data where over 70% of the images is Pneumonia, prediction Pneumonia already has 70% success changes. I think further improvement could be done with a balanced data set. The pre-trained models I have chosen like ResNet152v2, InceptionResNetV2 have been trained on complex image data and that has significantly improved the prediction accuracy.

Improvement

The pre-trained models like InceptionResnetV2, ResNet152v2, we can check around unfreezing few layers (I tested with 5 layers) and see the performance. Enhancing the data set with a balance of Normal and Pneumonia images can help with training and accuracy. The data sets can be supplemented with additional Normal, Pneumonia x-rays. For the given images, we can do more transformations like rotating, mirror image zoom in, zoom out settings to help with training.

References

[1] Daniel S. Kermany, Michael Goldbaum, Wenjia Cai, Carolina C.S. Valentim, Huiying Liang, Sally L. Baxter, Alex McKeown, Ge Yang, Xiaokang Wu, Fangbing Yan, Justin Dong, Made K. Prasadha, Jacqueline Pei, Magdalene Y.L. Ting, Jie Zhu, Christina Li, Sierra Hewett, Jason Dong, Ian Ziyar, Alexander Shi, Runze Zhang, Lianghong Zheng, Rui Hou, William Shi, Xin Fu, Yaou Duan, Viet A.N. Huu, Cindy Wen, Edward D. Zhang, Charlotte L. Zhang, Oulan Li, Xiaobo Wang, Michael A. Singer, Xiaodong Sun, Jie Xu, Ali Tafreshi, M. Anthony Lewis, Huimin Xia, Kang Zhang
"Identifying Medical Diagnoses and Treatable Diseases by Image-Based Deep Learning" https://www.cell.com/cell/fulltext/S0092-8674(18)30154-5

[2] Paul Mooney https://www.kaggle.com/paultimothymooney/chest-xraypneumonia
[3] https://en.wikipedia.org/wiki/Pneumonia
[4]https://www.hopkinsmedicine.org/health/conditions-and-diseases/pneumonia
[5]
https://www.kaggle.com/jonaspalucibarbosa/chest-x-ray-pneumonia-cnn-transfer-learning/notebook
[6] https://www.kaggle.com/saran27/pneumonia-detection-using-cnn-91-3-accuracy
[7]
https://medium.com/@robparkin_38642/running-keras-models-on-amazon-sagemaker-b4059843f767
[8]
https://towardsdatascience.com/using-tensorboard-in-an-amazon-sagemaker-pytorch-training-job-a-step-by-step-tutorial-19b2b9eb4d1c
[9] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7345724/
[10] https://ourworldindata.org/pneumonia