IT in Industry, Vol. 9, No.3, 2021 Published Online 04-05-2021

NOVEL DEEP LEARNING-BASED MODEL PIPELINE FOR ANY LUNG DISEASE USING CHEST X-RAY IMAGES- A CASE OF COVID DETECTION

Rudransh Agnihotri [1], Sharmila Agnihotri [2]
[1] *XI, Scholar, Birla Vidya Niketan, Pushp Vihar sector 4 New Delhi 110017*
roshan@iul.ac.in
[2] *Assistant Professor, New Delhi Institute of Management, Tuglaqabad Institutional area New Delhi 110062,*
anuragcvsdu@gmail.com,.

Abstract: The corona virus spread very fast than any other virus in the world including China, Europe, USA and India. AI and deep learning are used in the healthcare sector for different purpose. This paper is written with respect early detection of lung deceases by using chest X ray with the help of Deep learning based pipeline for easy, Cheaper and perfect detection. Further, this paper is aim at developing a deep learning model for detection of lung deceases like TB, Phunumonia fibtrosis lung cancer and Covid 19 Using Chest X-ray. However in this paper this model will be tested specifically on Covid 19. This deep learning pipeline can also be applied for detecting any type of disease whether it is fibrosis, tuberculosis, lung cancer or any other disease by providing it the dataset of the disease which you want to detect. We have made a classifier with sensitivity of 98 percent and specificity of 98 percent and an overall model training accuracy is 98 percent itself. This paper can be a way forward for further research in the field of deep learning and its use in medical field.

1. Introduction

COVID-19, which began with the reporting of unknown causes of pneumonia in Wuhan, Hubei province of China on NOV 19, 2019, has rapidly become a pandemic. From China, this virus spread in the various parts of the world. The United States of America which is the largest economy in the world, here the first seven cases were found in January 2020 and then reached up to 300,000 cases by the 5th of April. The main thing about corona virus is that it spreads very rapidly than any other virus in the world. Proof for this can be seen above in the case of the USA. AI and deep learning are used in the healthcare sector for different purpose. One of the major uses is Lungs disease diagnoses. For deep learning you require a lot of data around millions of images or thousands of images and in addition to this you require a lot of computing power due to which most of the current models for lungs disease diagnoses are not successful or they have to be stopped due to the lack of data. Further no new improvisation is done in the deep learning models which developers are using to diagnose lungs disease. In the present scenario covid is all around the globe and people are coming up with various AI models to detect covid form chest X-ray but none of them are successful because they all are using same concept; some are not getting enough data or they don't have enough computing power. So, our objective is to design a deep learning-based pipeline for any lungs disease detection which could be trained on less data, less computing power and get more higher accuracy also this pipeline-based model could be easily deployed anywhere. There Fore I have proved my pipeline and my work by applying it on covid chest X-ray images along with pneumonia chest X-ray images. COVID-19 presentation, which began with the reporting of unknown causes of pneumonia in Wuhan, Hubei province of China on December 31, 2019, has rapidly become a pandemic. From China, this virus spread in the various parts of the world. The United States of America which is the largest economy in the world, here the first seven cases were found in January 2020 and then reached up to 300,000 cases by the 5th of April. The main thing about coronavirus is that it spreads very rapidly than any other virus in the world. Proof for this can be seen above in the case of the USA. The typical clinical features of COVID-19 include fever, cough, sore throat, headache, fatigue, muscle pain, and shortness of breath. The main challenge is the diagnosis of the coronavirus. In the current situation, there is only one test RT-PCR (Reverse transcription-polymerase chain reaction). There are two major drawbacks to this test. First is it takes a lot of time to diagnose COVID-19 using RT-PCR and it is also very costly. In India, the cost of an RT-PCR test for one person costs around 4000 rupees which is highly costly and everyone cannot afford it. The second major drawback of RT-PCR is that it gives a lot of false-negative results that is its sensitivity is very low around 60% to 70%. Which is why some patient with negative RT-PCR results are found to as corona virus-positive later on. The above situation tells us that we urgently

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731

need a solution which is more accurate, cheaper, and more accessible.

Therefore, the solution is a CT scan and chest X-rays. As we know that the CT scans and chest X-rays are routine diagnosis tools and they are performed in the day to day life of patients and chest X-rays are relatively cheaper. X-rays have been a very promising tool among the doctors and the radiologist to diagnose any type of disease related to chests. In India, a single chest X-ray cost up to rupees 400 only.

Chest X-rays are found to have shown the sign of COVID-19 at an early stage. Therefore, if we use deep learning and artificial intelligence in this process to identify chest X-rays as COVID-19 positive then it would be very helpful as it will lower down the burden on the hospital staff and infrastructure. This AI model is basically a progressive web app which can be used for the diagnosis of the chest X-ray as covid-19 positive or negative. The radiologist or the doctor who is dealing covid-19 patients need to upload the chest X-ray of the patients and within 2 to 3 seconds this app will classify that chest X-ray within 3 categories which are

- Covid positive.
- Pneumonia positive.
- Covid negative and Pneumonia negative case.

3rd category which is covid negative and pneumonia negative case is the category in which tells that the given chest X-ray is neither of covid nor pneumonia but it might have some other disease such as pneumothorax or Tuberculosis. We have used the cunning edge techniques of deep learning which are transfer learning and autoencoders. I have used several pre trained model like vgg16, Resnet-50, InceptionV3 and Xception for the classification process. Then this app is also giving Heatmaps which is gradient class activation map for the visualization and the cross examination of the results given by the model. Further the app is also generating annotations on the chest X-ray images which will annotate the exact location of the disease patches on the chest X-rays. Which will in turn help the doctors to know the exact spots where the disease is also, they will get an idea of the disease severity in the chest.

Objective: This paper is aim at developing a deep learning model for detection of lung deceases like TB, Phunumonia fibtrosis lung cancer and Covid 19 Using Chest X-ray. However in this paper this model will be tested specifically on Covid 19.

Review of literature: The concept of computer-aided diagnosis (CAD) for chest x-rays has been around for the past fifty years, and has made dramatic progress from rule-based survival prediction from lung x-rays to machine learning approaches to, now, deep learning. Ginneken et al. make the argument that CAD in radiology is imperative, as the workload of radiologists is quickly becoming unmanageable. First, computer-aided diagnosis was sought for other types of diagnostic tasks: breast cancer localization by Google Net and skin cancer classification by a network out of Stanford by Esteva et al.. Both of these well-designed networks, along with others, have proven that convolutional neural networks can be used very successfully in not just natural image classification, but also medical image classification and segmentation. In the world of CXR classification tasks, Rajkumar et al. used Google Net along with image augmentation and pre-training on ImageNet to classify CXR images as either front alor lateral with 100 percent accuracy. While this is not directly clinically relevant, it is an important proof on concept of the use of deep learning on CXR images. Anvi et al. sought to create a network that could, given a query image, rank the other CXR images in its database by similarity to the query. They found that a 5-layer convolutional neural network was much more effective than similarity based on image descriptors. Such a network could be used to help clinicians search for past cases easily and help inform their current or future diagnosis. In 2016, Shin et al. used a CNN to detect specific diseases in CXR images and assign disease labels. They then used an RNN to describe the context of the annotated disease based on the features of the CNN and patient metadata. They were only able to achieve validation accuracy of .698 on this ambitious task. This performance may be largely due to their relatively small data set size of 7470 images, the challenges of multi-class classification, and incorporating textual data from patient records. Most recently, in 2017, Wang et al. successfully designed a CNN to diagnose specific diseases by detecting and classifying lung nodules in CXR images with high accuracy.

2. Dataset:

Due to the Non-disclosure policy of the government, I have taken the open-source datasets which are available on the internet. Images of covid-19 chest X-rays has been used to train the model; these images were collected from Mendel datasets. Images of pneumonia were collected from Kaggle pneumonia detection challenge. Normal chest X-rays and other disease chest X-ray were also collected from the Kaggle pneumonia challenge itself.

- **Dataset division:** All the images were first simply downloaded into their respective folder. Covid images were downloaded into Covid folder, Pneumonia images were downloaded into the Pneumonia folder and normal and other chest X-rays were downloaded into non Covid and no pneumonia folder. Total 978 images of each category were downloaded into their respective folders. These images were further divided into train, test and validation folders, which is as follows: -

Train: -

- Covid – 326 images
- Pneumonia – 326 images
- Non Covid non pneumonia - 326 images

Validation: -

- Covid – 326 images

- Pneumonia – 326 images
- Non Covid non-Pneumonia - 326 images

Test: -

- Covid – 326 images
- Pneumonia – 326 images
- Non Covid non-Pneumonia - 326 images

Size of each original image was 3000x3000 pixels. The ratio was 978:978:978. Hence the dataset was kept very-well balanced. To remove any kind of biasness from the model.

3. Processing of Data

- **Pre-processing: -** Since this dataset was collected from various sites on internet, therefore this dataset contains a lot of noise which will degrade the performance of our deep learning model. Hence, I have done a lot of pre-processing on this dataset so that our deep learning model can learn good features from these chest X-ray images and give us appropriate results. Major source of variance and noise was in terms of some written text on the images, contrast view, positional variance. Like all the images were not clicked in the same angle. First pre-processing step which was done is that, a Gaussian filter was applied over all the images in their respective folders to remove any kind of noise in terms of text on the images. This was done using python library called **OPEN-CV.** Second pre-processing which was done in the terms of enhancing the contrast and make the chest X-ray images clearer. The images were enhanced using the **PILLOW** library in python. Third and the final pre-processing was done in terms of enhancing the edges and sharpness of the image as some of the images were quite dull due to which backbone, chest and heart were not very much clear. This was again done with the help of **PILLOW** library in python.(**This all was done by taking the advice of the doctors and the radiologists.)**
- **Data Augmentation: -** Since we all know that Convolutional neural networks are bad at classifying images at different angles and also Convolutional neural networks are very prone to Over fitting which is not good for any model. Therefore, in order to resolve these problems each image before being given to the neural network to be trained upon was first augmented at different angles like image were filliped up to 90-degree, 180-degree, 120 degree and etc. Then the images were further augmented by shearing them, zooming and cropping them. This also gave me an advantage by increasing the dataset diversity.
- **Lungs Segmentation: -** Since complete image of a chest X-ray could have various information written on it such as name of the hospital. In which lobe or part, the lungs disease is or patients name etc. This all information's could be considered as noise for a deep learning model though it is necessary for the hospital to keep records and noise is one of the reasons which could reduce our model's accuracy. Further we just want our model to look only at the lungs specifically and normal CNN model would iterate over the whole chest X-ray which are not required.

In order to overcome all these problems, we will perform **Semantic Segmentation** using the **U-NET** architecture to segment out the lungs from the whole X-ray and train our main CNN model particularly on the segmented lungs only. U-NET consists of two convolutional neural networks, one is the contractional CNN also called as the encoder which has convolutional layers and max pooling layers and is used to capture the context of the image. The second CNN is symmetric expanding CNN also called as decoder network which is used to enable precise localization using the transposed convolutional neural network. Hence it a fully connected neural network. It does not contain any dense layer so that it can be trained on the images of any size.

OUR U-NET ACHITECTURE: We have used the same U-NET model as described above but with some changes. Instead of simply using an encoder made with normal CNN layers I have used the concept of transfer learning. In which we used pre trained model called VGG16 model along with its pre trained IMAGE-NET in the encoder part of the U-NET network. This made the model to learn the features of the image more properly and in a more efficient way. The decoder network was kept same.

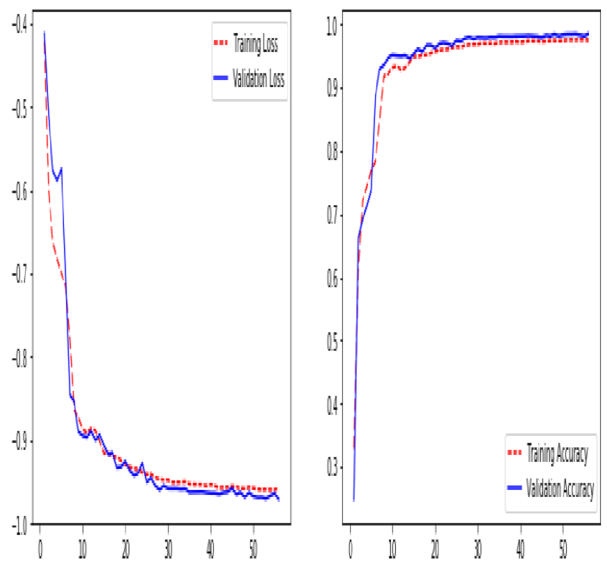
4. Descritption of the dataset used for our u-net segmentation model:

The dataset was taken from kaggle. Name of the dataset is Chest X-ray masks and labels uploaded earlier with size of 5.04 GB. There were 800 chest X-ray images and masks directory contains corresponding 800 images of segmented chest X-rays. Model was given chest x-ray images as the training data and the segmented chest X-ray images as the target data. Hence this is the case of binary classification.

- **DATA SPLITTING: -** Data was divided into training and validation part 750 images were kept for training purpose from both the folders. 50 images were kept for validation purpose from both the folders.
- **DATA AUGMENTATION: -** To protect the model from overfitting and providing It more image samples at different angle data augmentation was performed prior to the training phase.
- **SEGMENTATION MODEL EVALUATION: -** This model was only evaluated on the basis of accuracy graph and loss graph.

Both graphs of the model are as follows:

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731



SOURCE: - Our App output image

Above graphs are showing that the training accuracy and validation accuracy has increased per epoch. Training loss and validation loss has also gone down per epoch. Model achieved 98 percent accuracy.

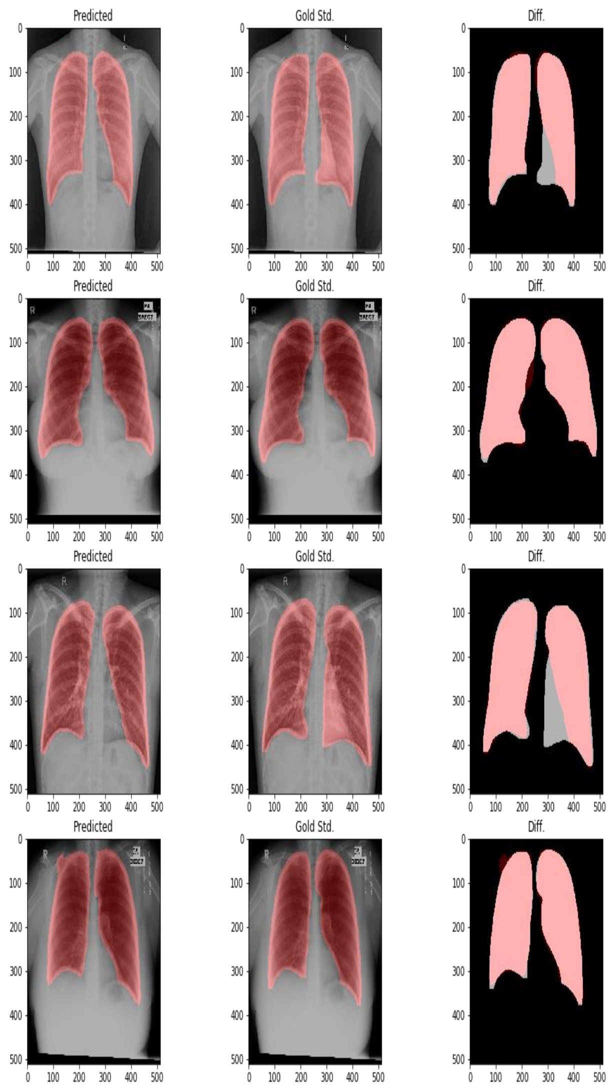
5. Results given by the model:

Over here we should not very about the difference cause the model has just skipped marking the heart area in the X-ray and we also don't require the mask in the heart region. Further the difference is very much minute.

Lungs final segmentation: - In the last we will crop the lungs and save them in a directory so that we can use these segmented lungs for further analysis in our pre-processing pipeline.

Feature Extraction: - This is the second last step which is very necessary to be performed before giving the images for training to our classifier. Over here I have used the latest technique in the field of deep learning that is convolutional autoencoders.

Since I had a very small dataset for training our classifier and in general CNN require a very large dataset so that they can be properly trained. To overcome this problem, I have performed a process called feature extraction along with dimensionality reduction process.



SOURCE: - Our App output image

To implement feature extraction along with dimensionality reduction I have used a convolutional autoencoder. First of all, Autoencoders are mainly a dimensionality reduction (or compression) algorithm. Autoencoders consist of 3 components which are mainly **ENCODER, CODE AND DECODER.** The function of each part is as follows:

1. **ENCODER**: - This is the part of the network that compresses the input into a latent-space representation. It can be represented by an encoding function *h=f(x).*
2. **DECODER**: - This part aims to reconstruct the input from the latent space representation. It can be represented by a decoding function *r=g(h).*
3. **CODE**: - This part of the network lies between the encoder and the decoder. This part consists all the extracted features from a given set of data.

There is no use if a neural network is just building the copy of the given dataset. But the reason why autoencoder is used because of the following reasons: -

DIMENSTONALITY REDUCTION: - When the encoder part of the network compresses the input into a latent-space representation. This latent space representation is basically a

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731

compressed set of extracted features from the given dataset. Since the extracted features are in a compressed form, therefore the size of these feature reduces and our model learns more efficiently.

SELECTING IMPORTANT FEATURES: - This part is also a branch under the dimensionality reduction process. To make model more generalize and more efficient with less quantity of dataset we need to extract only **important features** from the dataset. If we are to perform this process without auto encoders then it is a very tedious task because in this case, we have to first extract all the features of the dataset at once, after that we visualize these feature on various types of graphs and see their distribution. Then we use unsupervised machine learning techniques such as k- means clustering algorithm to cluster the important features together in group and then in the last we perform principal component analysis. Principal component analysis (PCA) is a technique for reducing the dimensionality of such datasets, increasing interpretability but at the same time minimizing information loss. It does so by creating new uncorrelated variables that successively maximize variance. Therefore, the PCA is just used to identify and reduce the weights of more important feature from that cluster, then in the last these features are given to the model for training. There is a problem with such approach as it is very time consuming and we cannot reduce and extract all the important features from the given dataset. Since this is a manual process therefore sometimes, those features are also included which are not necessary.

To overcome this problem, we use autoencoders. Autoencoder in a single step, automatically without any manual intervention and at once extract all the important features from the given dataset, compresses them and reduces them. Most important of all is that it does not leave any single feature which is important for the model training and automatically throws the unwanted features. Also, we can reconstruct the same dataset again from the decoder part with the extracted feature and we can construct a new dataset also of the similar type once again

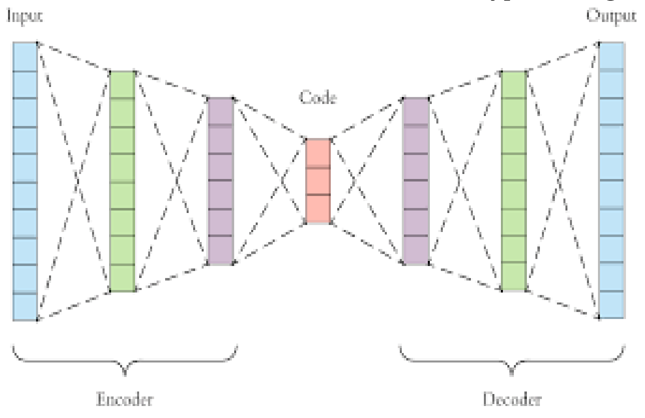


Figure - 1: This is how the an autoencoder look

Source: - towardsdatascience.com

Since in the Covid detection dataset we are mainly dealing with images therefore I have used a convolutional autoencoder. In this the type of autoencoder we replace the encoder with a pretrained model and uses its last pooling layer as a layer to store all the extracted features, then these features are given to the code layer of the network where the latent representation of these features are generated and all the important processes take place. And since we are only concerned with these important features therefore, we do not include the decoder part in the network

Feature extraction model vgg16: - For the encoder network of our feature extraction model I have used famous pretrained model called as VGG16 along with its ImageNet weights. By including the ImageNet weights the model will extract more important features by itself before giving it to the code layer. Architecture of VGG16 is as follows: -

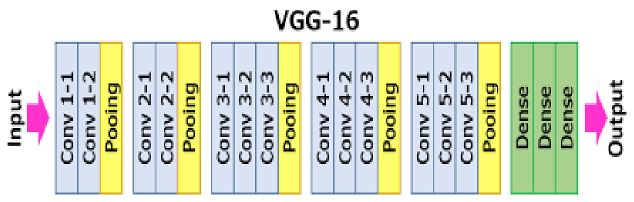


Figure - 2

Source – neurohive.io

The only change we have made in this model before including it into the encoder part is that we have cut down the last 3 dense layers which are fully connected layers and have kept only the pooling layer in the last. This pooling layer basically contains the map of the features extracted on the basis of the ImageNet weights. After this it is given to the separate code layer.

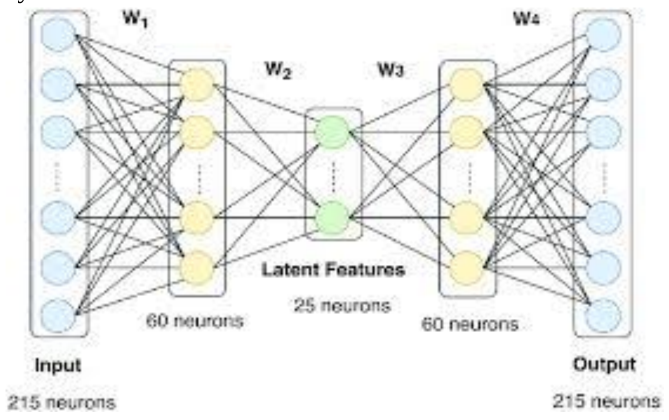


Figure - 3

Source: - towardsdatascience.com

Details of the training phase: - All the models were trained in google Collab where you get NVidia tesla K80 cloud GPU and a ram of 12.0 MB for free. Also, before giving the extracted feature for the training to our classifier the features were divided in the ratio of 80:20 where the 80 percent of the data was kept for the training and rest 20 percent of the dataset is kept for testing purpose.

The classifier VGG16: - This is the model which will be used for the classification of the images as covid positive, pneumonia positive and non covid and non-pneumonia

Copyright © Authors
ISSN (Print): 2204-0595
ISSN (Online): 2203-1731

positive. I again used the vgg16 pretrained model architecture as a classifier, but this time the ImageNet weights were not included and the model was trained from scratch on the basis of the features extracted from the convolutional autoencoder. Some changes were done in the model architecture this time to train it as classifier according to our custom dataset needs. Since in the original structure of the classifier the dense layer in the last has 1000 nodes which shows that this model has the capability to classify 1000 objects into different category. But our dataset has only three labels therefore I replaced the last 3 dense layers of the networks with our own custom layers and the output layer of our new architecture has 3 nodes each represents their respective classes. Last layer also has an activation function of **SoftMax**

In model the last block5_pool layer there are new layers which has average_pooling2d_2, flatten, dense, dropout and then in the last again dense with 3 nodes which represents 3 different classes. Hence to make this model work according to my requirement I added 5 more layers extra. Model was trained on 33,027 parameters.

Classifier Performance: -

Hyperparameters of the classifier: - Since I have trained a vgg16 model as a classifier I have set the following parameters.

1. Learning-rate - 0.0001
2. Regularization – L2
3. Batch-size – 128
4. Drop Out – 0.5

Model loss comparison and training comparison: - Following graph represents the training accuracy and the validation accuracy

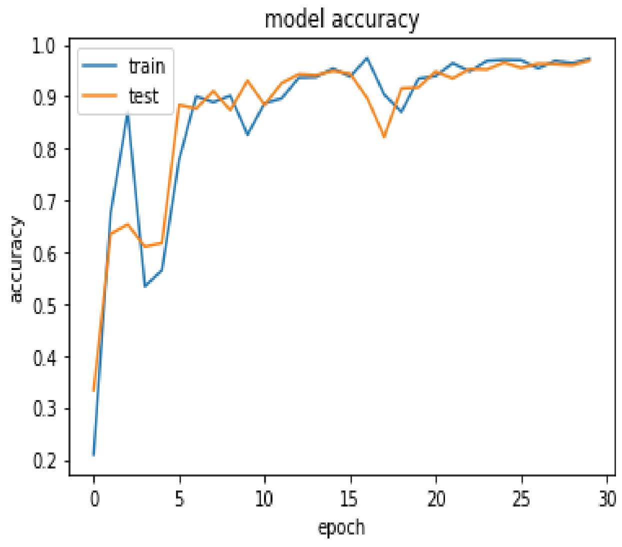


Figure - 5

Source: Author model image

The orange line represents the validation accuracy in real though it is written as test and the blue line represents the training accuracy. From the above graph as it can be seen that the training and the validation accuracies keep increasing with the no of epochs and at the 30th epoch the validation and the training accuracy are highest and both are almost same. So, in training and validation graph model has performed quite well.

Following graph represent the loss rate of the model: -

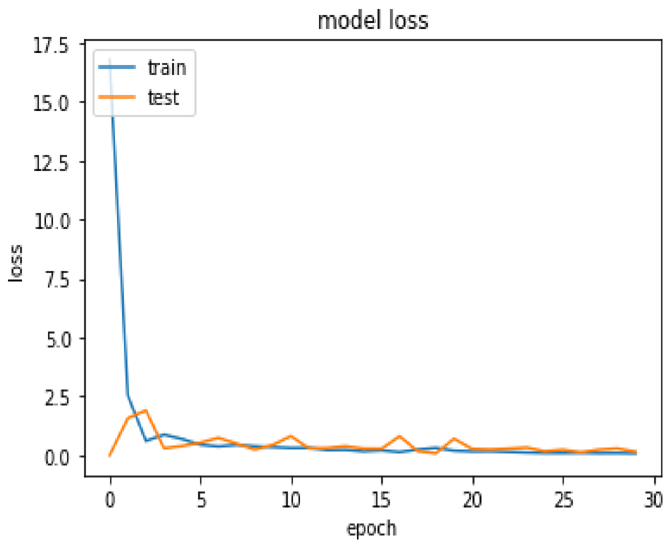
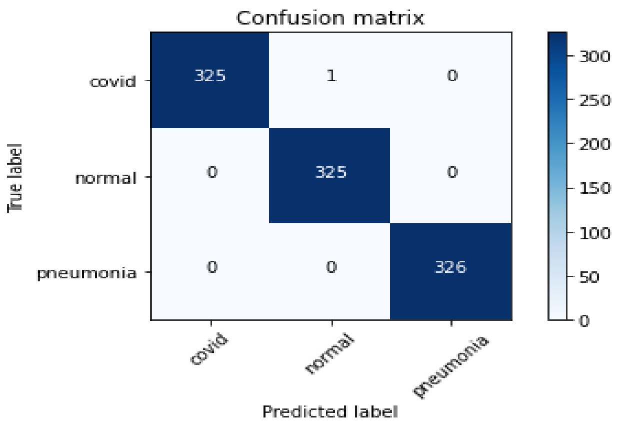


Figure - 6

Source: Author model image

Orange line represents the validation loss though it is written as test loss and the blue line represents the training loss. The above graph shows that the model loss has continuously decreased which shows that whatever model has learned it has learned it pretty well. As the no of epochs are increasing the more training and validation losses are decreasing continuously and at the 30th epoch both training and validation loss are lowest and almost same. Therefore, the model has again performed well in the loss comparison.

Model evaluation on the basis of Confusion Matrix: - Confusion matrix plays an important role while you are evaluation any deep learning model. Confusion Matrix will tell you exactly how many images your model classified incorrectly or correctly. Confusion matrix will also tell you the exact no of true negative, True Positive, False negative and False positive. Following graph will tell you the exact no of images which were wrongly classified and the no of images which were correctly classified.



Confusion matrix

True label \ Predicted label	covid	normal	pneumonia
covid	325	1	0
normal	0	325	0
pneumonia	0	0	326

Figure - 7 **Source:** Author model image

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731

This graph has been generated on the basis of classifying the testing dataset. The model was never ever trained on this dataset. Over here the testing dataset comprised of 326 images of covid,325 images of normal (**normal represents the non covid and non-pneumonia category)** and 326 images of pneumonia. So total no of images which testing dataset compromised of is 977. Model has predicted only 1 image incorrectly out 977 images. This one image was covid image but the model predicted it as normal.

Following graph represents the percentage wise error of in predicting the class of each image: -

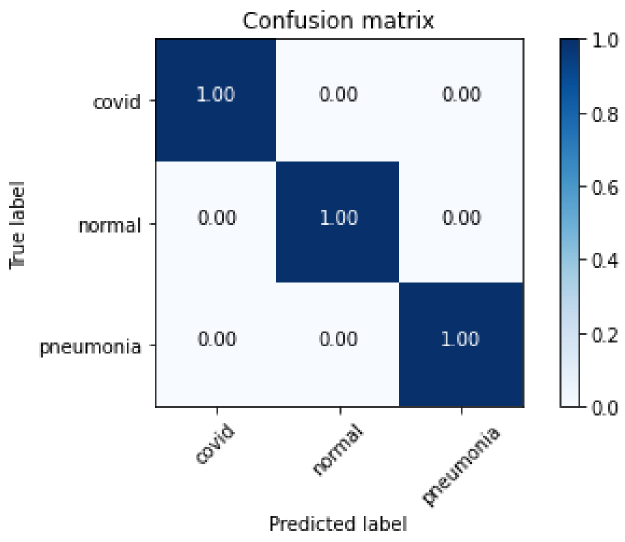


Figure - 8

Source: Author model image

These graphs have not been generated synthetically instead the they have been generated by the model itself. Scikit- learn library was used to generate these confusion matrices.

Sensitivity and specificity: - Sensitivity and specificity describe how well the test discriminates between patients with and without disease. **Sensitivity** is the proportion of patients *with* disease who test positive. In probability notation: $P(T^+|D^+) = TP / (TP+FN)$. [gcim.unmc.edu]. **Specificity** is the proportion of patients *without* disease who test negative. In probability notation: $P(T^-|D^-) = TN / (TN + FP)$. [gcim.unmc.edu]

- TP is True Positive
- FN is False Negative
- TN is True Negative
- FP is False Positive

TP, FN, TN and FP are calculated on the basis of Confusion matrix shown in Fig – 7

- Number of TP (True Positive) - 652
- Number of TN (True Negative) - 325
- Number of FP (False Positive) - 0
- Number of FN (False Negative) - 1

Therefore, calculated sensitivity and specificity using the formula $P(T^+|D^+) = TP / (TP+FN) *100$ and $P(T^-|D^-) = TN / (TN + FP) *100$ respectively are as follows: -

Sensitivity (%) = 99%

Specificity (%) = 100%

Visualizing the results of classifier: - In order to gain insight into the relevant features learned by the above-mentioned deep learning model I sought to create images which will let us know easily that which part of the image was looked by the model and on the basis of it the model is giving the classifications. Therefore, the type of the image which I generated is called as heatmap. Technical term for them is class activation maps.

Class Activation Maps are a simple technique to get the discriminative image regions used by a CNN to identify a specific class in the image. In other words, a class activation map (CAM) lets us see which regions in the image were relevant to this class. Uses the class-specific gradient information flowing into the final convolutional layer of a CNN to produce a coarse localization map of the important regions in the image. Grad-CAM is a strict generalization of the Class Activation Mapping. Unlike CAM, Grad-CAM requires no re-training and is broadly applicable to any CNN-based architectures. We have also generated the similar grad cam for my model also. In my heatmaps all the disease part in the chest image is covered with only a single colour which is blue. This will also help doctors to know the over view analysis about the spread and progression of the disease in the chest X-ray.

Following are the some heatmaps generated by the model: -

Original covid chest X-ray Generated Heat map on covid chest X-ray

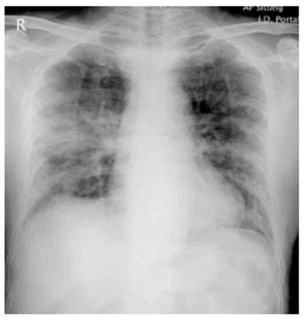
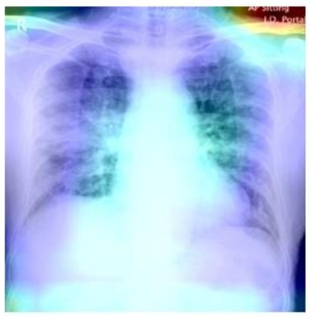



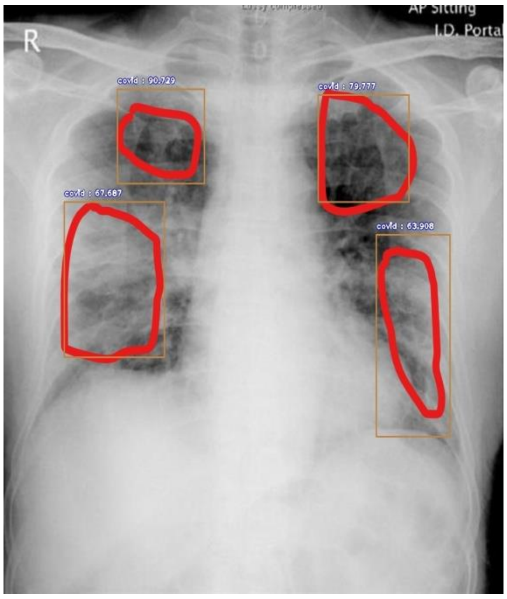
Figure – a Figure – b

Source - Authors image

Figure A is the original chest X-ray of a Covid positive chest X-ray and the figure b shows the heatmap of the X-ray. The blue part in the figure b shows the area of the chest covered by the disease. Although this doesn't give the exact quantity of the disease in the chest X-ray but it gives an overview look to the doctor.

Object Detection Technique: - System also uses the technique of object detection to localize the disease patch in chest X-ray image. Object detection methods will classify on the images on the basis of the detected disease patch in chest X-ray image. Another advantage of this system is that it will let doctors know the exact place where the disease is and it

will also tell the exact area of the chest covered by the disease and it will also tell you the severity of the disease. When you will give this model an image it will simply annotate the area covered by the disease. For object detection I'm using an open source pretrained model called yolo (you only look once). This image shows you that how this model is detecting the disease.
Image patches detected by YOLO Model



Source – Authors image
Figure - 7:

The brown rectangular boxes are the marking which is done by the model itself and the red polygon type marking has been done by a doctor to cross examine the results produced by the object detection model.

6. Findings :

We can conclude that by using CNN and using existing deep learning models with little modification in their architectures we have successfully trained a deep learning pipeline model on very less amount of data and fair computing power. This deep learning pipeline can also be applied for detecting any type of disease whether it is fibrosis, tuberculosis, lung cancer or any other disease by providing it the dataset of the disease which you want to detect. We have made a classifier with sensitivity of 98 percent and specificity of 98 percent and an overall model training accuracy is 98 percent itself. Model was evaluated on the basis of heatmaps, confusion matrix and ROC curve. Further, This model can be used for other lung diseases by just training the model on that data using our model

References

[1] B. van Ginneken, C. Schaefer-Prokop, and M. Prokop. Compter-aided diagnosis: How to move from the laboratory to the clinic. Radiology, 261(3), 2011.

[2] B. van Ginneken, A. Setio, C. Jacobs, and F. Ciompi. Off-the-shelf convolutional neural network features for pulmonary nodule detection in computed tomography scans. Biomedical Imaging (ISBI), pages 286–289, 2015.

[3] Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, Dhruv Batra

[4] Dermatologist-level classification of skin cancer with deep learning neural networks. Andre Esteva, Brett Kuprel, Roberto A Novoa, Justin Ko, Susan M Swetter, Helen M Blau, Sebastian Thrun

[5] U-Net: Convolutional Networks for Biomedical Image Segmentation. Olaf Ronneberger, Philipp Fischer, Thomas Brox

[6] Autoencoders, Unsupervised Learning, and Deep Architectures. I. Guyon, G. Dror, V. Lemaire, G. Taylor and D. Silver

[7] You Only Look Once: Unified, Real-Time Object Detection. Joseph Redmon University of Washington, Santosh Divvala Allen Institute for Artificial Intelligence, Ross Girshick Facebook AI Research, Ali Farhadi University of Washington.

[8] Hinton G. LeCun Y, Bengio Y. Deep learning. Nature., 521(7553), 2015.

[9] Alvin Rajkomar, Sneha Lingam, Andrew G. Taylor, Michael Blum, and John Mongan. High-Throughput Classification of Radiographs Using Deep Convolutional Neural Networks. Journal of Digital Imaging, 30:95–101, Feb 2017.

[10] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gael Varoquaux. API de- ¨ sign for machine learning software: experiences from the scikit-learn project. In ECML PKDD Workshop: Languages for Data Mining and Machine Learning, pages 108–122, 2013.

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731