VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

EEE3999 – Technical Answers to Real world Problems

Diagnosis of CoVID using image processing methods on Lung X-Ray

Team Members

Rohan Mathur – 18BEE0132

Advait R. Marathe – 18BEE0164

Under the Guidance

Dr. Dhanamjayulu C

Associate Professor

School of Electrical Engineering

VIT, Vellore

Winter Semester 2020-21

Diagnosis of CoVID using image processing methods on Lung X-Ray

1. Abstract –

The covid-19 pandemic has affected many lives drastically. Whether it be the education sector or economy, it has proven to be one of the most prominent challenges so far. Till December 2020, over 3.1 million people have died due to the pandemic. Only effective testing, social distancing and isolation can help us to curb the spread of the pandemic. A potent and rapid way of testing can help us in this challenge a lot.

Thus, we should develop an effective testing service. Although the tests are available, still the method of physical contact with the test subject is inevitable. Thus, in this project, we aim to develop a contact-less yet efficient way to detect the infection using image processing on the X-Ray lung scan. Such a method helps us harness the existing framework to the fullest for rapid testing of the population. And since the process is contactless, it also assures the safety of the staff. The results are automated, the medical personnel can focus on more challenging aspects of the pandemic, as in the more effective and cautious treatment of an individual.

Image processing is one of the effective ways to detect any anomaly in the human body. The CoVid diagnosis consists of performing an X-Ray on a patient and the fibrosis and shadowing of the lungs, due to insufficient oxygen supply, is observed. The severity of the symptoms can help us to classify the gravity of a health emergency.

2. Keywords –

coronavirus; COVID-19; deep learning; convolution neural network; X-Ray images

Sr. No.	Name of the paper	Author	Description	Outcome (Accuracy)	Year published
1.	Learning to diagnose from scratch by exploiting dependencies among labels	Li Yao, Eric Poblenz, Dmitry Dagunts, Ben Covington, Devon Bernard, Kevin Lyman	Given a limited arrangement of potential labels, the multi-label order issue is to relate each occurrence with a subset of those labels. Being pertinent to applications in numerous spaces, an assortment of models have been proposed in the writing.	79.8%	2018
2.	Large-Scale Screening of COVID-19 from Community-Acquired Pneumonia using Infection Size-Aware Classification	Feng Shi, Liming Xia, Fei Shan, Dijia Wu, Ying Wei, Huan Yuan, Huiting Jiang, Yaozong Gao, He Sui, Dinggang Shen	All images were preprocessed to obtain the segmentation of both infections and lung fields, which were used to extract location-specific features. An infection Size Aware Random Forest method (iSARF) was proposed, in which subjects were	87.9%	2021

			automatically categorized into groups with different ranges of infected lesion sizes, followed by random forests in each group for classification.		
3.	CovidGAN: Data Augmentation Using Auxiliary Classifier GAN for Improved Covid-19 Detection	Abdul Waheed, Muskan Goyal, Deepak Gupta, Ashish Khanna, Fadi Al-turjman, And Plácido Rogério Pinheiro.	GANs (Generative Adversarial Networks) combine two neural networks that compete to generate new virtual data instances that can be distributed as real data. For image generation, GANs are widely used. The Auxiliary Classier GAN variant of GAN was used to perform data augmentation in this study.	85%	2020
4.	Deep Learning-based Detection for COVID-19 from Chest CT using Weak	Chuansheng Zheng, Xianbo Deng, Qiang Fu, Jiapie Feng, Hui Ma	DeCoVNet took a CT volume and its 3D lung mask as input. The 3D lung mask	90.1%	2020

	Label		was generated by a pre-trained Unet. The 3D lung mask of an input chest CT volume helped to reduce background information and detect COVID-19 better.		
5.	ChestX-ray8: Hospital-scale Chest X-ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases	Xiaosong Wang, yifan Peng,Le Lu, Mohammedadi Bageri,Ronald M Summers.	Common thoracic disease detection and localization are done through DCNN, pathology is detected by weakly labelled supervision.	81.41%	2017
6.	Development and evaluation of an artificial intelligence system for COVID-19 diagnosis	Cheng Jin,Wiexiang Cheng,yukun Cao,Zhanwei Xu,Ziemng Tang,Xin Zhang,Lie Dheng,	The Authors took Reports of Chest X-ray of people with COVID 19 and trained a deep learning algorithm to detect it. As COVID 19 affects the lungs the most hence it is a faster way to	92.99%	2020

			detect the disease.		
7.	Deep Learning-Based Quantitative Computed Tomography model in Predicting the Severity of COVID-19: A Retrospective Study in 196 Patients	Weiya Shi, Xueqing Peng, Tiefu Liu, Zenghui Cheng, Hongzhou Lu, Shuyi Yang, Jiulong Zhang, Feng Li, Mei Wang, Xinlei Zhang, Yaozong Gao, Yuxin Shi, Zhiyong Zhang, Fei Shan.	Using the regression via the means of LASAO, the severity of patients is found out, and accordingly effective treatment is provided.	89.0%	2021
8.	Deep learning-based model for detecting 2019 novel coronavirus pneumonia on high-resolution computed tomography	Jun Chen, Lianlian Wu, Jun Zhang, Liang Zhang, Dexin Gong, Yilin Zhao, Qiuxiang Chen, Shulan Huang, Ming Yang, Xiao Yang, Shan Hu, Yonggui Wang, Xiao Hu, Biqing Zheng, Kuo Zhang, Huiling Wu, Zehua Dong, Youming Xu, Yijie Zhu, Xi Chen, Mengjiao Zhang, Lilei Yu, Fan Cheng	The model uses the UNet++ algorithm as its backbone. The 512✕512 images are processed on Keras and the model returns the status of CoVID infection.	95.24%	2020

		& Honggang Yu.			
9.	A deep learning algorithm using CT images to screen for CoronaVirus Disease (COVID-19)	Shuai Wang, Bo Kang, Jinlu Ma, Xianjun Zeng, Mingming Xiao, Jia Guo, Mengjiao Cai, Jingyi Yang, Yaodong Li, Xiangfei Meng, Bo Xu1.	The ROI (Region of Interest) is drawn in a particular X-Ray. The ROI is searched for various patterns depicting infection in an individual's X-Ray.	82.9%	2021
10.	Deep learning COVID-19 detection bias: accuracy through artificial intelligence	Shashank Vaid & Reza Kalantar & Mohit Bhandari	The use of transfer learning reduces the need for a larger labelled dataset and thus making the learning easier.	96.3%	2020
11.	CT radionics can help screen the coronavirus disease 2019 (COVID-19): a preliminary study	Mengjie Fang, Bingxi He, Li Li, Di Dong, Xin Yang, Cong Li, Lingwei Meng, Lianzhen Zhong, Hailin Li, Hongjun Li & Jie Tian	The lung lesions are separated from CT and the radionic features depicting CoVID presence are extracted.	82.6%	2020
12.	COVID-19 Infection Detection from Chest X-Ray Images Using	Asu Kumar Singh, Anupam Kumar, Mufti Mahmud, M	The lung XRay preprocessing and feature extraction is done and the	99.65%	2021

	Hybrid Social Group Optimization and Support Vector Classifier	Shamim Kaiser, Akshat Kishore	possibility of a person getting infected are given.		
13.	A fully automatic deep learning system for COVID-19 diagnostic and prognostic analysis	Shuo Wang, Yunfei Zha, Weimin Li, Qingxia Wu, Xiaohu Li, Meng Niu, Meiyun Wang, Xiaoming Qiu, Hongjun Li, He Yu, Wei Gong, Yan Bai, Li Li, Yongbei Zhu, Liusu Wang and Jie Tian.	The lung area segmentation and suppression of the non-lung area is done and then the diagnostic analysis is carried out using the other neural network.	87.0%	2020
14.	Can AI Help in Screening Viral and COVID-19 Pneumonia?	Muhammad E.H Chowdhury, Tawsifur Rahman, Amith Khandakar, Rashid Mazhar, Muhammad Abdul Kadir, Zaid Bin Mahbub, Khandakar Reajul Islam, Muhammad Salman Khan, Atif Iqbal, Nasser Al Emadi, Mamun Bin Ibne Reaz, And	This paper explains the general flow of a CNN model or any AI Model in general used for CoVid detection.	-	2020

		Mohammad Tariqul Islam.			
15.	Artificial Intelligence Distinguishes COVID-19 from Community-Acquired Pneumonia on Chest CT	Lin Li, Lixin Qin, Zeguo Xu, Youbing Yin, Xin Wang, Bin Kong, Junjie Bai, Yi Lu, Zhenghan Fang, Qi Song, Kunlin Cao, Daliang Liu, Guisheng Wang, Qizhong Xu, Xisheng Fang, Shiqin Zhang, Juan Xia, Jun Xia	The CT slices are passed and through the probability determination and SoftMax function the patients are classified into various categories according to infections.	89%	2020
16.	Automated soil prediction using bag-of-features and chaotic spider monkey optimization algorithm	Sandeep Kumar Basudev Sharma, Vivek Kumar Sharma, Ramesh C. Poonia	The CSO algorithm is used for feature extraction in the soil. The same can be implemented for feature extraction in CT Lung X-Ray scan.	79%	2018
17.	Deep learning Enables Accurate Diagnosis of Novel Coronavirus (COVID-19) with CT images.	Song Ying, Shuangjia Zheng, Liang Li, Xiang Zhang, Xiaodong Zhang, Ziwang Huang, Jianwen Chen, Huiying Zhao,	The preprocessing of lung XRay images is done using OpenCV and then processed through deep learning.	86%	2020

		Ruixuan Wang, Yutian Chong, Jun Shen, Yunfei Zha, Yuedong Yang.			
18.	4S-DT: Self Supervised Super Sample Decomposition for Transfer learning with application to COVID-19 detection	Asmaa Abbas, Mohammed M. Abdelsamea, and Mohamed Medhat Gaber	The dataset is self-segregate d by using transfer learning to provide a minimal and proper labelled dataset for training.	97.4%	2020
19.	CovXNet: A multi-dilation convolutional neural network for automatic COVID-19 and other pneumonia detection from chest X-ray images with transferable multi-receptive feature optimization	Tanvir Mahmud, Md Awsafur Rahman, Shaikh Anowarul Fattah	The already proposed CovXNet is further multi-diluted, i.e. Transfer Learning for improving the accuracy.	94.8%	2020
20.	Econet: An Ensemble Of Deep Convolutional Neural Networks Based On Efficientnet To Detect Covid-19 From	NK Chowdhury, M Rahman, N Rezoana	The paper uses EfficientNet along with a deep learning algorithm for enhanced acc.	95.51%	2020

	Chest X-rays				
21.	Towards an Effective and Efficient Deep Learning Model for COVID-19 Patterns Detection in X-ray Images	Eduardo Luza, Pedro Silva, Rodrigo Silvaa, Ludmila Silva, Gladston Moreiraa, David Menottid	The paper uses just this family of NN with random forest classification for detection.	93.9%	2020
22.	Learning to recognize Abnormalities in Chest X-Rays with Location-Awar e Dense Networks	Sebastian Gundel, Sasa Grbic, Bogdan Georgescu, S. Kevin Zhou, Ludwig Ritschl, Andreas Maier and Dorin Comaniciu	The paper proposes a method to segment the various parts of CT scan and then proper labelling of those.	AUC = 87%	2018
23.	A Deep Learning System to Screen Novel Coronavirus Disease 2019 Pneumonia	Xiaowei Xu, Xiangao Jiang, Chunlian Mac, Peng Dud, Xukun Li, Shuangzhi Lv, Liang Yu ,Qin Ni, Yanfei Chen, Junwei Su, Guanjing Lang, Yongtao Li , Hong Zhao, Jun Liu, Kaijin Xu, Lingxiang Ruan, Jifang Sheng, Yunqing Qiu, Wei Wua, Tingbo Liang, Lanjuan Li	The candidate region is segregated and the probability of infection is determined.	86.7%	2020

24.	Evolutionary algorithms for automatic lung disease detection	Naman Gupta, Deepak Gupta, Ashish Khanna, Pedro P. Rebouças Filho Victor Hugo C. de Albuquerque	The improvised evolutionary algorithms are applied to preprocessed images.	99%	2019
25.	Deep learning covid-19 features on cxr using limited training data sets	Oh, Yujin, Sangjoon Park, and Jong Chul Ye.	The first neural network is used for image segmentation and then detection via other neural networks.	88.9%	2020

3. Introduction –

A pandemic is a disease globally affecting several populations. The globe has witnessed many pandemics within the twentieth century. The stream of contagious disease viruses is the main reason for pandemics. These viruses show ever-changing behaviour with the ever-changing seasons. Thus, predicting the action of such viruses is a formidable step to prevention. Health professionals typically create correct predictions concerning most viruses. But some viruses have such unusual behaviour. Thus, the square measure is troublesome to predict. Such viruses cause pandemics as humans do not have the immunity to resist such viruses. The latest coronavirus malady called COVID-19 has appeared and unfolds very quickly. Since its discovery in Dec 2019 in a metropolis, China, the infection has already met 199 countries and territories. The Severe Acute metabolism Syndrome Coronavirus two (SARS-CoV-2) causes COVID-19 [1]. The virus could be an RNA (RNA) virus from the Coronavirus family, the family from which many viruses cause respiratory illness. The added severe variety of coronaviruses is Severe Acute metabolism Syndrome Coronavirus (SARS-CoV) and Middle EastRespiratory Syndrome Coronavirus (MERS-CoV). COVID-19 can cause metabolism ailments starting from the common cold to fatal diseases like respiratory disorder. The number of cases worldwide has reached 5,817,385 resulting in the deaths of 362,705 people as of May 30 2020 as per a report revealed by the World Health Organization (WHO) [2]. The correct information concerning the emergence of COVID-19 is still unknown. However, the initial cases have established links with the Huanan (Southern China) food Wholesale Market [3, 4]. The infection is contagious, and the virus unfolds amongst humans via metabolism droplets, physical contact, and additionally through faecal-oral transmission [5]. Various cases of respiratory disorder of unknown cause were rumoured in a metropolis, China in December 2019. The patients showed similar clinical characteristics with infectious agent pneumonia [6]. The patients suffering from COVID-19 infection square measure were discovered to have severe respiratory disorders with abnormal observations on chest computerized axial tomography (CT) examination [7]. The inconvenience of medication for this infection needs efficient identification strategies to dominate the malady. Common cold to respiratory disorder is caused by a gaggle of viruses called CoV. These diseases embody respiratory, enteric, renal, and medicine diseases. These viruses are classified into four genres particularly alpha-CoV, beta-CoV, gamma-CoV, and delta-CoV [8]. The virus affects people of all age teams and genders. A research study reveals that two teams of people square measure specifically full of this infection. The first group of people squares measures people who square measure higher than sixty years old. The

second cluster is of these people. The United Nations agency has some underlying medical conditions like polygenic disease, cardiovascular infection, and cardiovascular disease. The common symptoms of COVID-19 embody fever, dry cough, and respiratory issues like shortness of breath, muscular soreness, and fatigue. In some patients, symptoms and vomiting also are rumoured. The severity of the infection ranges from delicate contagious disease to respiratory disorder inflicting metabolism ailments. The advanced stage of the infection can even cause organ failures and Acute metabolism Distress Syndrome (ARDS) resulting in the deaths of the patients. The fast human to human transmission of the disease could be a matter of subtle concern for the regulative authorities globally. The management of COVID-19 mostly depends on the identification at the proper time. The offered methods for identification comprise of laboratory tests like Reverse-Transcription enzyme Chain Reaction (RT-PCR), a period RT-PCR (RRT-PCR), and Reverse Transcription Loop-mediated equal Amplification (RT-LAMP) test [9, 10]. The laboratory tests have some limitations. Firstly, the check needs testing kits that have restricted handiness within the provided chain. Secondly, the test needs time overwhelming thanks to the laboratory processes concerned. The X-Ray facilities are accessible easily worldwide and the results are square measure also made at a quick pace. Therefore, the chest XRay pictures are also used for police work in the presence of COVID-19. An automatic method supporting chest X-Ray pictures for support in clinical deciding is vital for disease management. In line with the United Nations agency, the infection can be controlled by stopping the chain of transmission. The officials have rumoured that testing and isolation square measure the two key actions that square measure help break the chain of transmission. Therefore, the correct identification is significant in dominant COVID-19. The detection of COVID-19 may be done at the AN earlier stage with chest pictures compared to the PCR testing. The chest X-Ray pictures may be analyzed by victimization artificial intelligence techniques [11]. Numerous techniques for identification of COVID-19 using machine learning techniques on imaging square measure are offered within the literature. A transfer learning model for the identification of coronavirus from chest X-Ray images is given in referee [12]. Another technique with improved accuracy given a segmentation-based approach. The strategy classified the input pictures as simple, virus infection, and COVID-19 [13]. A deep learning-based model is applied to CT pictures for the detection of COVID-19. Some researchers have additionally developed public datasets consisting of chest X-Ray images of COVID-19 patients [14, 15]. a technique named COVID-Net is developed and applied on these public datasets for identification of COVID-19 [14]. The employment of deep learning for identification from the chest X-Ray pictures provides rapid results. Deep learning models' square

measures are being used widely for medical image processing. In [16], the detection of the respiratory disorder is concluded in victimization convolution neural networks. During this paper, an automatic technique for the identification of COVID-19 from a deep network is projected. The projected network utilizes the feature generated by multiresolution analysis. The mix of wave transforms alongside the deep network brings multiple benefits. The wave decomposition is fed into the network. The network used isn't the traditional Convolutional Neural Network (CNN). A depthwise divisible network is employed during this work.

4. Background -

In this section, we will discuss some basics about the convolutional neural network and some basic concepts to know before the implementation of the same via software code.

I The Convolutional Neural Network (ConvNet) -

In the field of deep learning, the convolutional neural network is used for visual imaging processes. This neural network algorithm is inspired by the animal cortex. The animal cortex captures the image via the eye and then the brain interprets the image. For example, the eye is a convex lens that captures the image as an inverted virtual image and then the brain inverts it again to make it upright and then we interpret it based on past experiences.

CNN uses the same sort of algorithm. The images are pre-processed and then interpreted based on past data or labelled data. CNN’s are regularized versions of multilayer perceptrons. Multilayer perceptrons usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "full connectivity" of these networks makes them prone to overfitting data. Typical ways of regularization, or preventing overfitting, include: penalizing parameters during training (such as weight decay) or trimming connectivity (skipped connections, dropout, etc.) CNN’s take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble patterns of increasing complexity using smaller and simpler patterns embossed in their filters. Therefore, on a scale of connectivity and complexity, CNNs are on the lower extremity.

II The building blocks of CNN -

A convolutional neural organization comprises an information layer, covered up layers and a yield layer. In any feed-forward neural organization, any center layers are called covered up in light of the fact that their information sources and yields are concealed by the initiation capacity and last convolution. In a convolutional neural organization, the secret layers incorporate layers that perform convolutions. Normally this incorporates a layer that plays out a speck result of the convolution portion with the layer's info framework. This item is generally the Frobenius internal item, and its initiation work is usually ReLU. As the convolution bit slides along the information grid for the layer, the convolution activity produces an element map, which thus adds to the contribution of the following layer. This is trailed by different layers, for example, pooling layers, completely associated layers, and standardization layers.

a) Convolutional layers

In a CNN, the information is a tensor with a shape: (number of data sources) x (input stature) x (input width) x (input channels). In the wake of going through a convolutional layer, the picture gets disconnected to a component map, likewise called an initiation map, with shape: (number of information sources) x (highlight map height) x (include map width) x (highlight map channels). A convolutional layer inside a CNN by and large has the accompanying ascribes:

- Convolutional channels/bits characterized by a width and tallness (hyper-boundaries).

- The quantity of info channels and yield channels (hyper-boundaries). One layer's info channels should rise to the quantity of yield channels (likewise called profundity) of its information.

- Extra hyperparameters of the convolution activity, like cushioning, step, and enlargement.

Convolutional layers convolve the information and pass its outcome to the following layer. This is like the reaction of a neuron in the visual cortex to a particular stimulus.[17]. Each convolutional neuron measures information just for its open field. Albeit completely associated feedforward neural organizations can be utilized to learn includes and characterize information, this engineering is for the most part unrealistic for bigger data sources like high-goal pictures. It would require an extremely high number of neurons, even in shallow engineering, because of the enormous information size of pictures, where every pixel is an applicable information inclusion. For example, a completely associated layer for a (little) picture of size 100 x 100 has 10,000 loads for every neuron in the subsequent layer. All things being equal, convolution lessens the quantity of free boundaries, permitting the organization to be more profound [18]. For instance, paying little mind to picture size, utilizing a 5 x 5 tiling area, each with similar shared loads, requires just 25 learnable boundaries.
Using regularized weights over fewer parameters avoids the vanishing gradients and exploding gradients problems seen during backpropagation in traditional neural networks [19,20] Furthermore, convolutional neural networks are ideal for data with a grid-like topology (such as images) as spatial relations between separate features are taken into account during convolution and/or pooling.

b) Pooling layers

Convolutional organizations may incorporate nearby or potentially worldwide pooling layers alongside conventional convolutional layers. Pooling layers decrease the components of information by consolidating the yields of neuron bunches at one layer into a solitary neuron in the following layer. Nearby pooling joins little bunches, tiling sizes, for example, 2 x 2 are usually utilized. Worldwide pooling follows up on every one of the neurons of the element map.[21,22] There are two normal sorts of pooling in well known use: max and normal. Max pooling utilizes the greatest worth of every neighborhood bunch of neurons in the component map, [23,24] while normal pooling takes the normal worth.

c) Fully connected layers

Completely associated layers interface each neuron in one layer to each neuron in another layer. It is equivalent to a conventional multi-facet perceptron neural organization (MLP). The leveled grid goes through a completely associated layer to order the pictures.

d) Receptive field

In neural organizations, every neuron gets a contribution from some number of areas in the past layer. In a convolutional layer, every neuron gets contribution from just a limited space of the past layer called the neuron's responsive field. Regularly the territory is a square (for example 5 by 5 neurons). Though, in a completely associated layer, the open field is the whole past layer. In this manner, in each convolutional layer, every neuron takes contribution from a bigger region in the contribution than past layers. This is expected to apply the convolution again and again, which considers the worth of a pixel, just as its encompassing pixels. When utilizing expanded layers, the quantity of pixels in the responsive field stays consistent, yet the field is all the more meagerly populated as its measurements develop when joining the impact of a few layers.

e) Weights

Every neuron in a neural organization figures a yield esteem by applying a particular capacity to the info esteems obtained from the open field in the past layer. The capacity that is applied to the info esteems is controlled by a vector of loads and a predisposition (commonly genuine numbers). Learning consists of iteratively changing these inclinations and loads.

The vector of loads and the predisposition are called channels and address specific highlights of the info (e.g., a specific shape). A distinctive component of CNNs is that numerous neurons can have a similar channel. This diminishes the memory impression in light of the fact that a solitary inclination and a solitary vector of loads are utilized across all responsive fields that share that channel, instead of each open field having its own predisposition and vector weighting. [25]

III Basic mathematical formulae governing the neural network

Portion convolution isn't just utilized in CNN's but on the other hand is a vital component of numerous other Computer Vision calculations. It is an interaction where we take a little network of numbers (called bit or channel), ignore our picture and change it depending on the qualities from the channel. Resulting highlight map esteems are determined by the accompanying equation, where the information picture is meant by f and our portion by h. The files of lines and segments of the outcome framework are set apart with m and n individually.

$$G[m, n] = (f * h)[m, n] = \sum_{j} \sum_{k} h[j, k]\, f[m - j, n - k] \qquad 1)$$

Subsequent to putting our channel over a chosen pixel, we take each worth from the piece and increase them two by two with relating values from the picture. At last, we summarize everything and put the outcome in the correct spot in the yield highlight map. Above we can perceive how such an activity looks on a miniature size, yet the thing is significantly really fascinating, is the thing that we can accomplish by performing it on a full picture.

Since our picture recoils each time we perform convolution, we can do it just a predetermined number of times, before our picture vanishes totally. Also, in the event that we take a gander at how our piece travels through the picture we see that the effect of the pixels situated on the edges is a lot more modest than those in the focal point of the picture.

To take care of both of these issues we can cushion our picture with an extra line. For instance, on the off chance that we utilize 1px cushioning, we increment the size of our photograph to 8x8, so the yield of the convolution with the 3x3 channel will be 6x6. Generally, by and by, we fill in extra cushioning with zeros. Contingent upon if we use cushioning, we are managing two sorts of convolution — Valid and Same. Naming is very tragic, so for clarity: Valid — implies that we utilize the first picture, Same — we utilize the line around it so the pictures at the information and yield are a similar size. In the subsequent case, the cushioning width should meet the accompanying condition, where p is cushioning and f is the channel measurement (typically odd).

$$\mathbf{p} = \frac{f-1}{2} \qquad \mathbf{2)}$$

These days, we don't have to waste time with backpropagation — profound learning systems do it for us, yet I feel it merits knowing what's happening in the engine. Very much like in thickly associated neural organizations, we will probably compute subordinates and later use them to refresh the upsides of our boundaries in a cycle called inclination drop.

In our estimations, we will utilize a chain rule - which I referenced in past articles. We need to survey the impact of the adjustment of the boundaries on the subsequent highlights map, and consequently on the eventual outcome. Before we begin to delve into the subtleties, let us concede to the numerical documentation that we will utilize - to make my life simpler, I will relinquish the full documentation of the fractional subordinate for the abbreviated one noticeable underneath. In any case, recollect that when I utilize this documentation, I will consistently mean the halfway subordinate of the expense work.

$$\mathbf{dA^{[l]}} = \frac{\partial L}{\partial A^{[l]}} \quad \mathbf{dZ^{[l]}} = \frac{\partial L}{\partial Z^{[l]}} \quad \mathbf{dW^{[l]}} = \frac{\partial L}{\partial W^{[l]}} \quad \mathbf{db^{[l]}} = \frac{\partial L}{\partial b^{[l]}} \qquad \mathbf{3)}$$

3. Proposed method -

The proposed method will have the following steps -

I Metadata visualization -

First of all, we will dive into the data present in the form of metadata. A script can be exclusively created for visualization of the type of data present. This script shows the various fields in the metadata, such as patient name, country of origin, findings via X-Ray, the axis of X-Ray, etc. These data can be visually presented in the form of pie charts or bar graphs.

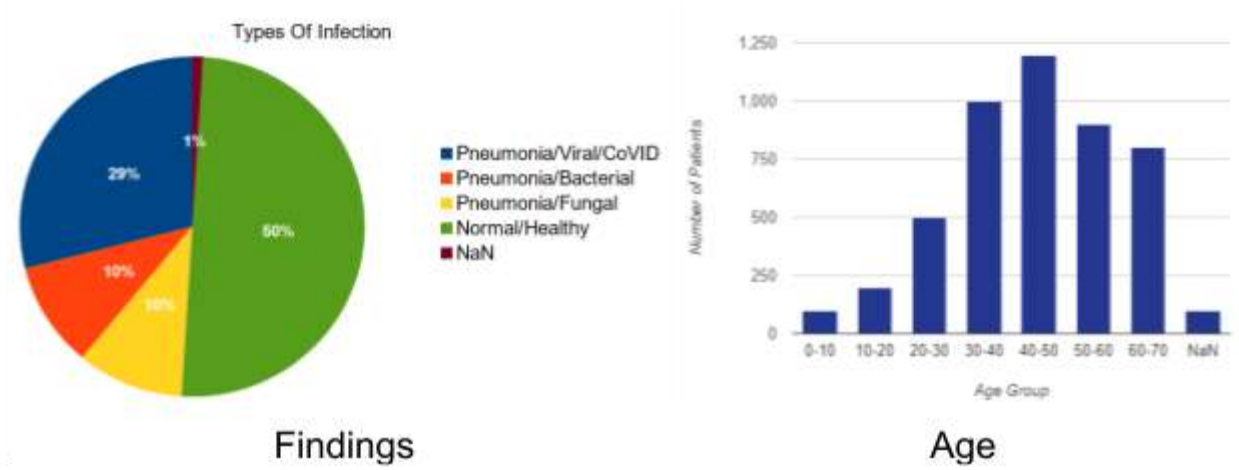


Figure 1 - Sample Distribution Charts

II Data Cleaning -

The next step is to remove the images that have no findings. Such images can hinder the processing capabilities of the neural network. These images can be removed with the help of a script that can go through the entries with 'no findings' and can remove the same cols and the images from the dataset. Also, the same can be repeated for the cols with other missing data entries, but since our model just predicts the infection based on the image itself, we can ignore the rest of the other missing values.

III Preliminary data segregation -

In this step, we will be extracting the required images by the means of an automated script. The script will select images based on the type of findings and the type of X-Ray view mentioned in the metadata file. This script will look for all those images in the main repository data and then store them into a custom output directory.

IV Model construction -

Our model consists of four convolution 2D layers, two pooling layers and one flattening layer. The convolution 2D layers use 'ReLU' as an activation function, whereas the flattening layer uses the sigmoid function for the same. Now, we will look into each building block of the model and discuss their significance -

a) The Convolution 2D layer -

The convolution neural network layer is usually represented by (size_of_feature_map, number_of_channels_in_feature_map, size_of_image). Here, in this model, we will be using the image size of 150 x 150 and the number of feature map channels as 3.

$$\boldsymbol{conv}\,(\mathbf{I}, \mathbf{K})_{\mathbf{x,y}} = \sum_{i=1}^{n_H} \sum_{j=1}^{n_W} \sum_{k=1}^{n_C} \mathbf{K}_{\mathbf{i,j,k}}\, \mathbf{I}_{\mathbf{x+i-1,\, y+j-1,\, k}} \qquad \mathbf{4)}$$

Where I denotes the image and K denotes the kernel.
i,j,k denote the coordinates of the image
n_H, n_W, n_C denote the height, width and channels of the feature map respectively.

$$\boldsymbol{dim}\,(\boldsymbol{conv}\,(\mathbf{I}, \mathbf{K})) = \left(\left[\frac{n_H + 2p - f}{s} + 1\right], \left[\frac{n_W + 2p - f}{s} + 1\right]\right);\ \mathbf{s > 0}$$
$$= (\mathbf{n_H + 2p - f,\ n_W + 2p - f});\ \mathbf{s = 0} \qquad \mathbf{5)}$$

where [x] is the floor function of x
There are some special types of convolution:

- **Valid convolution:** p = 0
- **Same convolution:** p = $\frac{f-1}{2}$
- **1 ×1 convolution:** f = 1

Where p, f denotes the padding and the filter size respectively.
s is the stride index, which is the number of iterations taken to convolute the image

b) The activation function -

The activation function in any layer of the model is used to determine the weights of the feature map so that it can determine the correct output. Here, we will be using two types of activation functions -

i) ReLU Function -

The ReLU function is a simple function which purely returns the maximum value from a given input and zero (for a = 0). It is generally used to eliminate all the negative weights from the feature map and get desirable predictions.

$$\mathbf{f(x) = max\ (x,\ ax)} \qquad \mathbf{6)}$$

ii) Sigmoid Function -

This function is rather a complex one and it is given by the equation -

$$\mathbf{S(x) = 1 - S(-x)} = \frac{1}{1 + e^{-x}} = \frac{e^{x}}{1 + e^{x}} \qquad \mathbf{7)}$$

This function is used to diminish all the unnecessary features from the feature map.

c) Pooling Layer -

Pooling layers are used to downsample the image features for faster processing of the image. The weights are pooled into a lower order matrix.

$$\mathbf{\textit{dim}\ (\textit{pooling}\ (image))} = \left(\left[\frac{n_H + 2p - f}{s} + 1\right], \left[\frac{n_W + 2p - f}{s} + 1\right], \mathbf{n_C}\right);$$

$$= (\mathbf{n_H + 2p - f,\ n_W + 2p - f,\ n_C\ );\ s = 0} \qquad \mathbf{8)}$$

d) Flattening Layer -

The flattening layer as the name suggests converts higher dimensional feature map output given by the convolution network into a single value. The single value can be boolean (true or false), binary (0 or 1), or a set of integers depicting the final result interpreted from the image.

e) Densing Layer -

The densing layer suggests the densely connected neurons. It means that all the neurons are given as input to the layer and the output is made fully connected to each neuron. Every densing layer is governed by an activation function.
In general, consider j^{th} node of the i^{th} layer we have the following equations -

$$\mathbf{z_j^{[i]}} = \sum_{l=1}^{n_{i-1}} \mathbf{w_{j,l}^{[i]}}\ \mathbf{a_l^{[i-1]}} + \mathbf{b_j^{[i]}}$$

$$\mathbf{a_j^{[i]}} = \mathbf{\Psi^{[i]}}\ (\ \mathbf{z_j^{[i]}}\ ) \qquad \textbf{9)}$$

The input $a^{[i-1]}$ might be the result of a convolution or a pooling layer with the dimensions ($n_H^{[i-1]}$, $n_W^{[i-1]}$, $n_C^{[i-1]}$). In order to be able to plug it into fully connected layer we flatten the tensor to a 1D vector having the dimension
($n_H^{[i-1]} \times n_W^{[i-1]} \times n_C^{[i-1]}$, 1), thus :

$$\mathbf{n_{i-1}} = \mathbf{n_H^{[i-1]}} \times \mathbf{n_W^{[i-1]}} \times \mathbf{n_C^{[i-1]}} \qquad \textbf{10)}$$

The learned parameters at the l^{th} layer are -

- **Weights ($\mathbf{w_{j,l}}$)** with $n_{l-1} \times n_l$ parameters.
- **Bias** with n_l parameters.

f) Dropout Layer -

As the name suggests, it drops out some samples from the model. This step ensures that the model is not overfitted (any model with an accuracy greater than 97% is termed as an overfitted model). The overfitted model can result in hazy predictions.

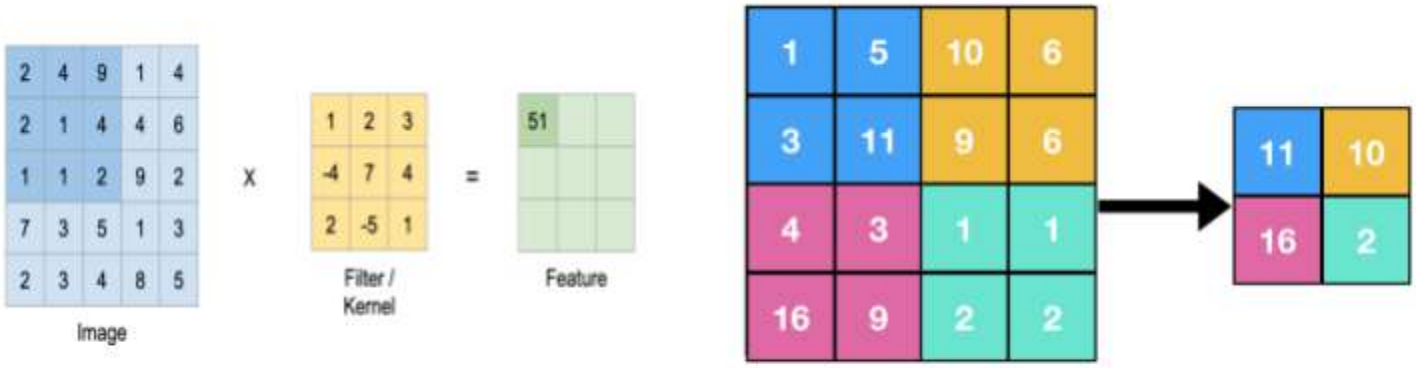


Figure 2 - The visual representation of various operations

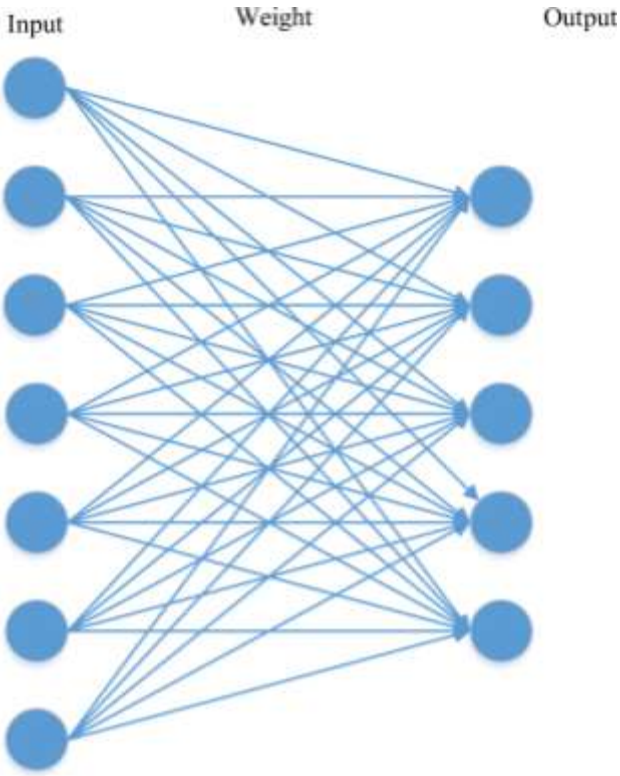


Figure 3 - The densing layer or fully connected layer

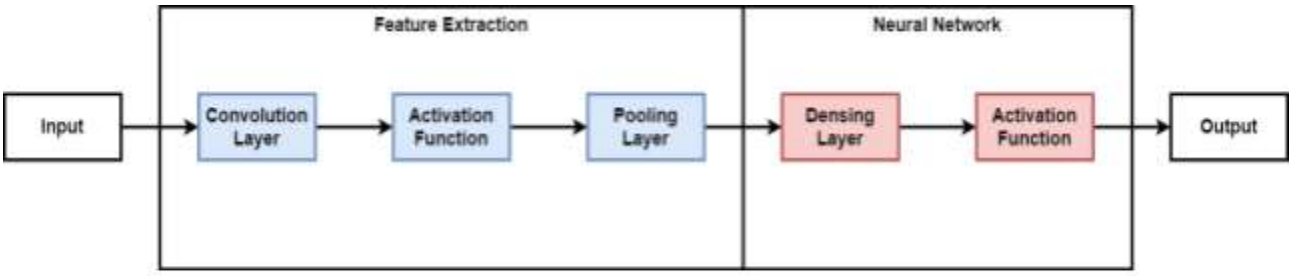


Figure 4 - Basic flowchart of CNN

4. Dataset -

The selected dataset has been gathered from sources [26]-[32]. The metadata along with the dataset was pre-processed with the script. The dataset was further truncated manually. The dataset contains 198 X-Ray images of CoVID patients and 636 images of normal X-Ray. A total of 834 images have been fully verified before training the model. The 59 images from the CoVID dataset and 190 images from the normal dataset are used for testing purposes and the rest is used for training purposes.

Table 1 - Summary of Dataset

	Training (70%)	**Testing (30%)**
CoVID	139	59
Normal	446	190
Total	585	249

5. Results and Discussion -

The model was executed using the Keras framework and with the Tensorflow backend. The model summary, its general flow and all the experimental results are presented in this section. First of all, the model appends the images into two lists - covid_images and normal_images. The images are then displayed using the matplotlib function -

Next, the model was deployed using the keras framework. Here, is the model summary:

Normal X-Ray CoVID Patient X-Ray

Figure 5 - Images used for training

Table 2 - Summary of Model				
Layer	**Activation Function Or Dropout**	**Shape**	**Number of Channels**	**Parameters**
Conv2D_1	ReLU	(148,148,72)	3	896
Pooling Layer	-	(74,74,32)	-	0
Conv2D_2	ReLU	(72,72,64)	3	18496
Pooling Layer	-	(36,36,64)	-	0
Conv2D_3	ReLU	(36,36,128)	3	73856
Conv2D_4	ReLU	(36,36,256)	3	295168
Flatten	-	262144	-	0
Dense_1	25%	32	-	8388640
Dense_2	Sigmoid	1	-	33
Total Parameters - 8,777,089				
Trainable Parameters - 8,777,089				

The dataset was split into 70% for training and 30% for testing. The model was then trained for 50 epochs for a batch size of 6. Overall, it took over an average of 35 seconds per epoch for training. The model accuracy and value accuracy, along with model loss along with value loss was plotted using the script.

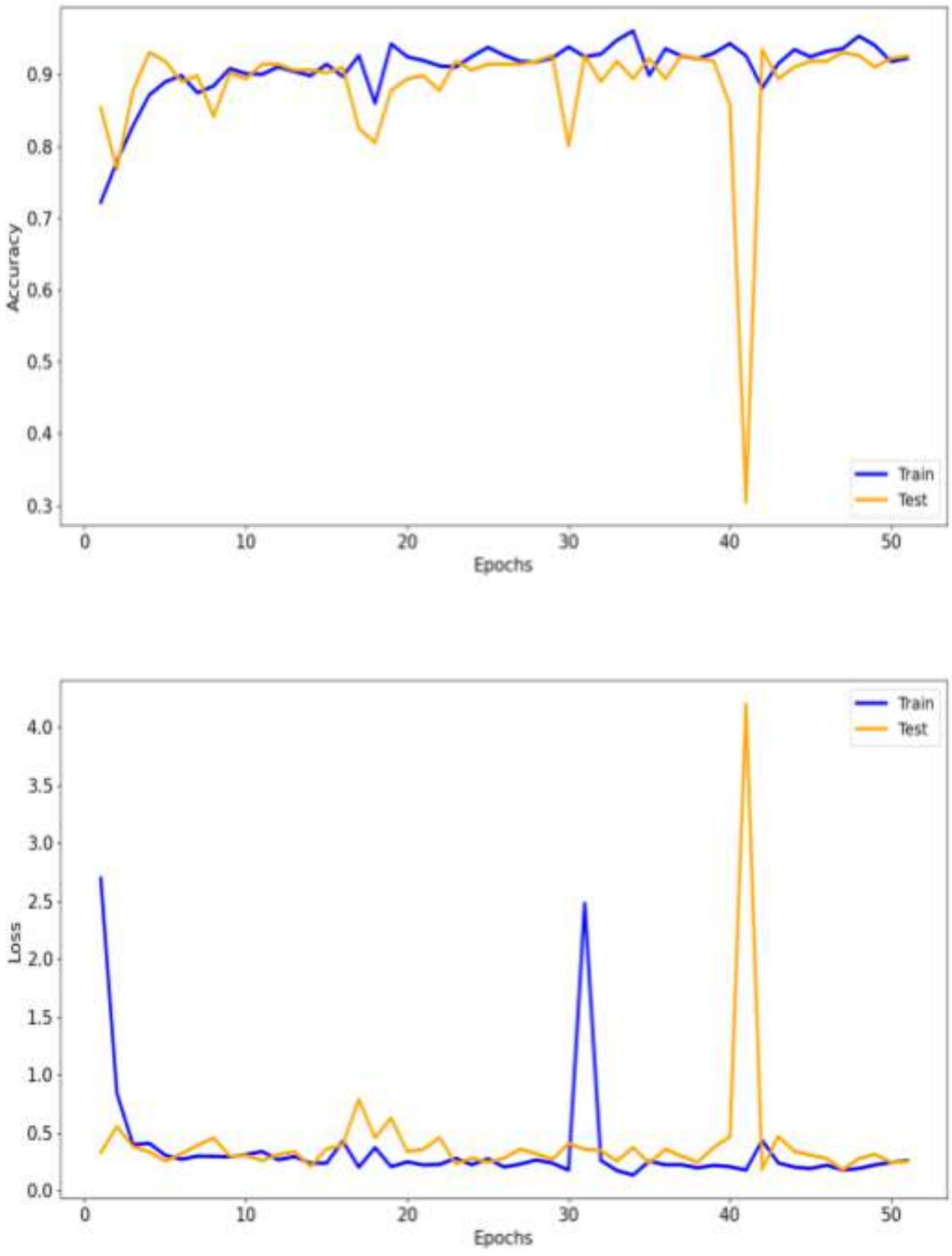


Figure 6 - Accuracy and loss plotted over epochs

The confusion matrix of the model was obtained as follows -

Table 3 - Confusion Matrix of the Model		
Condition	**Predicted**	**Actual**
CoVID	59	59
Normal	190	190
Total	249	249

The model performance is shown as follows -

Table 4 - Performance of the Model				
Condition	**Accuracy**	**Precision**	**Sensitivity**	**F1 Score**
CoVID	92.9 %	96	95	94
Normal	92.9 %	96	95	94

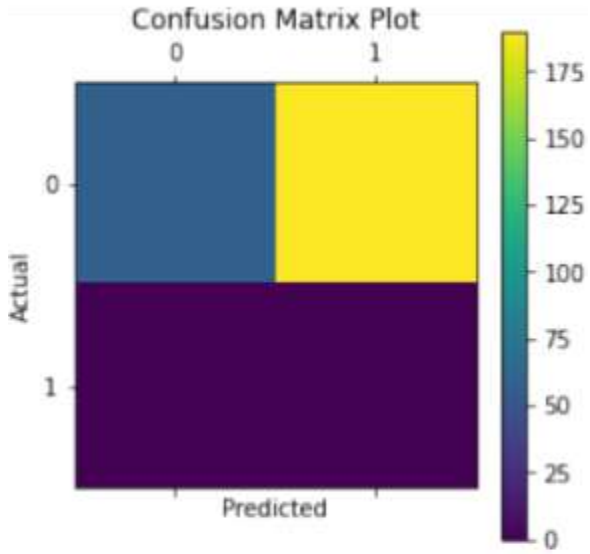


Figure 7 - Confusion Matrix plotted by script

As it can be seen from the plots and the other parameters, the model is pretty accurate and also has a lesser number of parameters to be learnt. The testing and training accuracy and loss tends to coincide as the model training is coming to conclusion.

6. Conclusion -

This project has developed a model to accurately differentiate between the CoVID X-Ray image and normal X-Ray. This model can be pretty useful as the training period of the model is pretty less and it can still give accurate results. The model is proven effective against most of the previous projects.
Here, is the comparison of all previous models with our proposed model -

Table 5 - Comparative Analysis of the Model				
Algorithm	**Accuracy**	**Precision**	**Sensitivity**	**F1 Score**
[33]	83	88	87	86
[34]	82%	87	86	85
[35]	87%	91	90	89
[36]	89%	93	92	91
Proposed	**92.9 %**	**96**	**95**	**94**

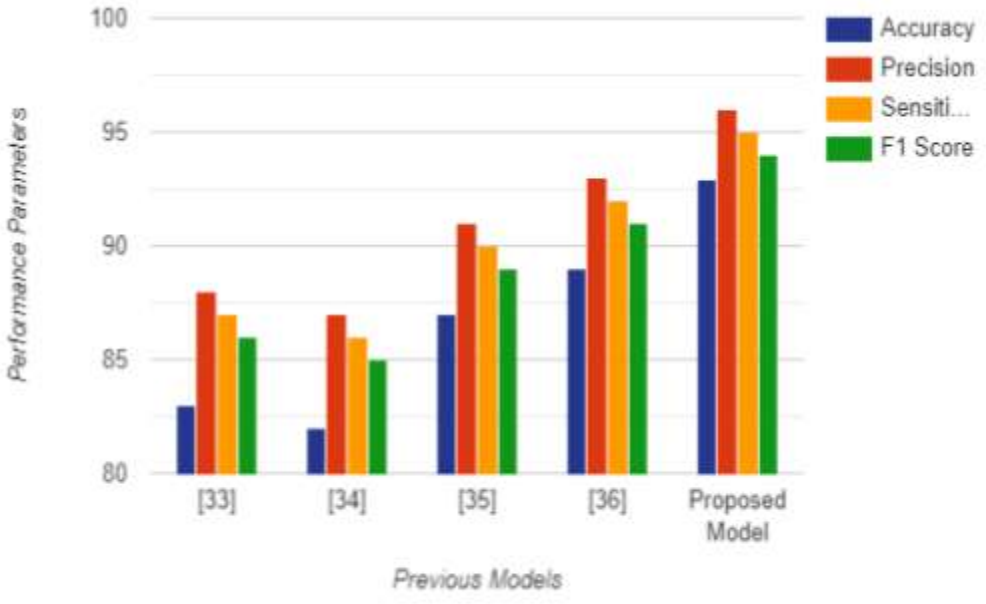


Figure 8 - Comparative Analysis of Model

Here is the flowchart summarising our model -

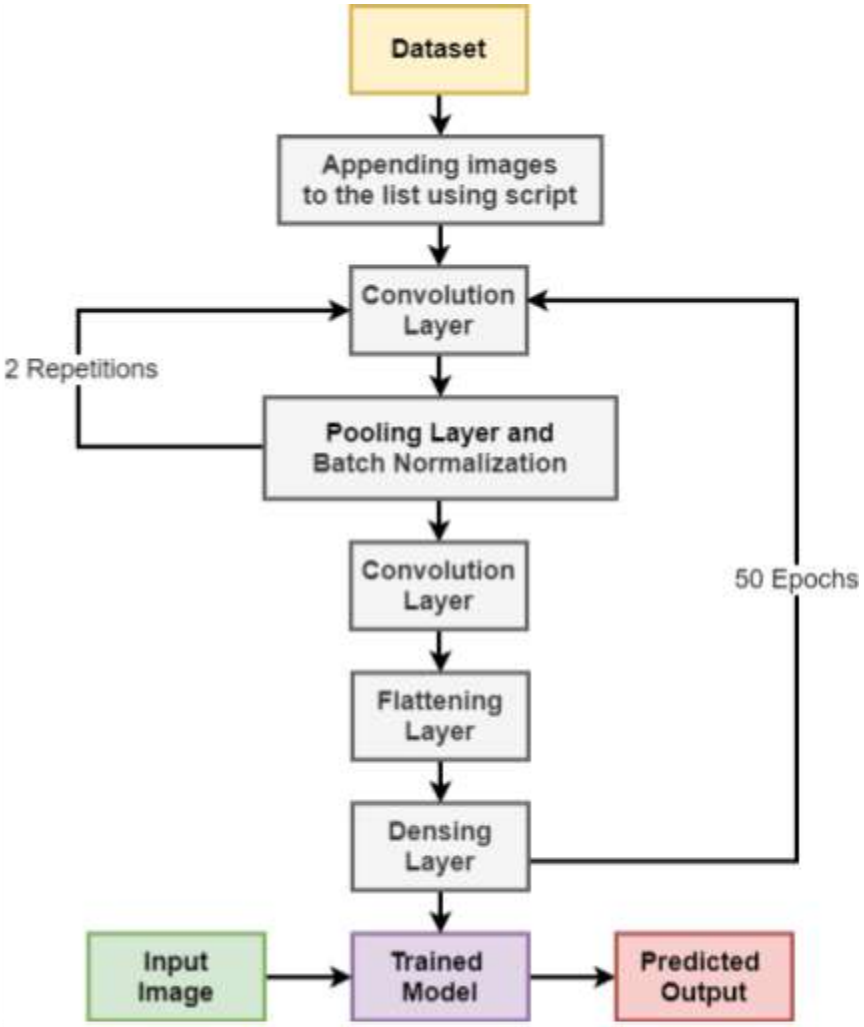


Figure 9 - The flowchart of proposed model

References –

[1] A. C. Walls, Y. J. Park, M. A. Tortorici, A. Wall, A.T. McGuire, and D. Veesler, Structure, function, and antigenicity of the SARS-CoV-2 spike glycoprotein, Cell, vol. 181, no. 2, pp. 281–292, 2020.

[2] World Health Organization, Coronavirus Disease 2019 (COVID-19), Situation Report–81,https://www.who.int/docs/default-source/coronaviruse/situationreports/20200410-sitrep-81-covid-19.pdf, 2020.

[3] N. S. Chen, M. Zhou, X. Dong, J. M. Qu, F. Y. Gong, Y. Han, Y. Qiu, J. L. Wang, Y. Liu, Y. Wei, et al., Epidemiological and clinical characteristics of 99 cases of 2019 novel coronavirus pneumonia in Wuhan, China: A descriptive study, Lancet, vol. 395, no. 10 223, pp. 507–513, 2020.

[4] Q. Li, X. H. Guan, P.Wu, X. Y.Wang, L. Zhou, Y. Q. Tong, R. Q. Ren, K. S. M. Leung, E. H. Y. Lau, J. Y.Wong, et al., Early transmission dynamics in Wuhan, China, of novel coronavirus–infected pneumonia, New England Journal of Medicine, vol. 382, no. 13, pp. 1199–1207, 2020.

[5] J. F. W. Chan, S. F. Yuan, K. H. Kok, K. K. W. To, H. Chu, J. Yang, F. F. Xing, J. L. Liu, C. C. Y. Yip, R. W. S. Poon, et al., A familial cluster of pneumonia associated with the 2019 novel coronavirus indicating person-to-person transmission: A study of a family cluster, Lancet, vol. 395, no. 10 223, pp. 514–523, 2020.

[6] D. W. Wang, B. Hu, C. Hu, F. F. Zhu, X. Liu, J. Zhang, B.B. Wang, H. Xiang, Z. S. Cheng, Y. Xiong, et al., Clinical characteristics of 138 hospitalized patients with 2019 novel coronavirus–infected pneumonia in Wuhan, China, JAMA, vol. 323, no. 11, pp. 1061–1069, 2020.

[7] C. L. Huang, Y. M. Wang, X. W. Li, L. L. Ren, J. P.Zhao, Y. Hu, L. Zhang, G. H. Fan, J. Y. Xu, X. Y. Gu, et al., Clinical features of patients infected with 2019 novel coronavirus in Wuhan, China, Lancet, vol. 395, no. 10 223, pp. 497–506, 2020.

[8] X. Y. Ou, Y. Liu, X. B. Lei, P. Li, D. Mi, L. L. Ren, L. Guo, R. X. Guo, T. Chen, J. X. Hu, et al., Characterization of spike glycoprotein of SARS-CoV-2 on virus entry and its immune cross-reactivity with SARS-CoV, Nature Communications, vol. 11, p. 1620, 2020.

[9] P. Zhai, Y. B. Ding, X. Wu, J. K. Long, Y. J. Zhong, and Y. M. Li, The epidemiology, diagnosis and treatment of COVID-19, International Journal of Antimicrobial Agents, vol. 55, no. 5, p. 105 955, 2020.

[10] X. Z. Xie, Z. Zhong, W. Zhao, C. Zheng, F. Wang, and J. Liu, Chest CT for typical 2019-nCoV pneumonia: Relationship to negative RT-PCR testing, Radiology, vol. 296, no. 2, p. 200 343, 2020.

[11] D. S. Kermany, M. Goldbaum,W. J. Cai, C. C. S. Valentim, H. Y. Liang, S. L. Baxter, A. McKeown, G. Yang, X. K. Wu, F. B. Yan, et al., Identifying medical diagnoses and treatable diseases by image-based deep learning, Cell, vol. 172, no. 5, pp. 1122–1131, 2018.

[12] S. Wang, B. Kang, J. L. Ma, X. J. Zeng, M. M. Xiao, J. Guo, M. J. Cai, J. Y. Yang, Y. D. Li, X. F. Meng, et al., A deep learning algorithm using CT images to screen for CoronaVirus Disease (COVID-19), https://doi.org/10.1101/2020.02.14.20023028, 2020.

[13] X. W. Xu, X. G. Jiang, C. L. Ma, P. Du, X. K. Li, S. Z. Lv, L. Yu, Y. F. Chen, J. W. Su, G. J. Lang, et al., Deep learning system to screen coronavirus disease 2019 pneumonia, arXiv preprint arXiv: 2002.09334, 2020.

[14] J. P. Cohen, P. Morrison, and L. Dao, COVID-19 image data collection, arXiv preprint arXiv: 2003.11597, 2020.

[15] O. Gozes, M. Frid-Adar, H. Greenspan, P. D. Browning, H. Q. Zhang, W. B. Ji, A. Bernheim, and E. Siegel, Rapid AI development cycle for the coronavirus (COVID-19) pandemic: Initial results for automated detection & patient monitoring using deep learning CT image analysis, arXiv preprint arXiv: 2003.05037, 2020.

[16] D. Varshni, K. Thakral, L. Agarwal, R. Nijhawan, and A. Mittal, Pneumonia detection using CNN based feature extraction, in Proc. 2019 IEEE Int. Conf. Electrical, Computer and Communication Technologies (ICECCT), Coimbatore, India, 2019, pp. 1–7.

[17] "Convolutional Neural Networks (LeNet) – DeepLearning 0.1 documentation". *DeepLearning 0.1*. LISA Lab. Retrieved 31 August 2013.

[18] Habibi, Aghdam, Hamed (2017-05-30). *Guide to convolutional neural networks: a practical application to traffic-sign detection and classification*. Heravi, Elnaz Jahani. Cham, Switzerland.

[19] Venkatesan, Ragav; Li, Baoxin (2017-10-23). Convolutional Neural Networks in Visual Computing: A Concise Guide. CRC Press

[20] Balas, Valentina E.; Kumar, Raghvendra; Srivastava, Rajshree (2019-11-19). *Recent Trends and Advances in Artificial Intelligence and the Internet of Things*. Springer Nature

[21] Ciresan, Dan; Ueli Meier; Jonathan Masci; Luca M. Gambardella; Jurgen Schmidhuber (2011). "Flexible, High-Performance Convolutional Neural Networks for Image Classification" (PDF). *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence-Volume Volume Two*. **2**: 1237–1242.

[22] Krizhevsky, Alex. "ImageNet Classification with Deep Convolutional Neural Networks"

[23] Yamaguchi, Kouichi; Sakamoto, Kenji; Akabane, Toshio; Fujimoto, Yoshiji (November 1990). *A Neural Network for Speaker-Independent Isolated Word Recognition*. First International Conference on Spoken Language Processing (ICSLP 90). Kobe, Japan.

[24] Ciresan, Dan; Meier, Ueli; Schmidhuber, Jürgen (June 2012). *Multi-column deep neural networks for image classification*. *2012 IEEE Conference on Computer Vision and Pattern Recognition*. New York, NY: Institute of Electrical and Electronics Engineers (IEEE). pp. 3642–3649.

[25] LeCun, Yann. "LeNet-5, convolutional neural networks"

[26] Henry Z. Lo. and Cohen, Joseph Paul “Academic Torrents: Scalable Data Distribution.” Neural Information Processing Systems Challenges in Machine Learning (CiML) Workshop, 2016, http://arxiv.org/abs/1603.04395.

[27] Cohen, Joseph Paul, and Henry Z. Lo. “Academic Torrents: A Community-Maintained Distributed Repository.” Annual Conference of the Extreme Science and Engineering

[28] Discovery Environment, 2014, http://doi.org/10.1145/2616498.2616528.

[29] Cohen, J. P., Morrison, P., Dao, L., Roth, K., Duong, T. Q., & Ghassemi, M. (2020). Covid-19 image data collection: Prospective predictions are the future. arXiv preprint arXiv:2006.11988.

[30] Paiva, O., 2020. CORONACASES.ORG - Helping Radiologists To Help People In More Than 100 Countries! | Coronavirus Cases - 冠状病毒病例. [online] Coronacases.org.

[31] Glick, Y., 2020. Viewing Playlist: COVID-19 Pneumonia Radiopaedia.Org. Radiopaedia.org.

[32] Ma Jun, Ge Cheng, Wang Yixin, An Xingle, Gao Jiantao, Yu Ziqi, … He Jian. (2020). COVID-19 CT Lung and Infection Segmentation Dataset (Version 1.0) [Data set]. Zenodo.

[33] Shi, Feng, et al. "Large-scale screening to distinguish between COVID-19 and community-acquired pneumonia using infection size-aware classification." *Physics in Medicine & Biology* 66.6 (2021): 065031.

[34] Fang, Mengjie, et al. "CT radiomics can help screen the coronavirus disease 2019 (COVID-19): a preliminary study." *Science China Information Sciences* 63.7 (2020): 1-8.

[35] Wang, Shuo, et al. "A fully automatic deep learning system for COVID-19 diagnostic and prognostic analysis." *European Respiratory Journal* 56.2 (2020).

[36] Li, Lin, et al. "Artificial intelligence distinguishes COVID-19 from community acquired pneumonia on chest CT." *Radiology* (2020).

Diagnosis of CoVID using image processing methods on Lung X-Ray

Dhanamjayulu C [1], Rohan Mathur [2], Advait Marathe [2]

1. *Assistant Professor, SELECT, Vellore Institute of Technology, Vellore, Tamil Nadu, India - 632014*
2. *Student, SELECT, Vellore Institute of Technology, Vellore, Tamil Nadu, India - 632014*

dhanamjayulu.c@vit.ac.in
rohan.mathur2018@vitstudent.ac.in
advait.marathe2018@vitstudent.ac.in

ABSTRACT

The CoVID-19 pandemic has affected many lives drastically. Whether it be the education sector or economy, it has proven to be one of the most prominent challenges so far. Till December 2020, over 3.1 million people have died due to the pandemic. Only effective testing, social distancing and isolation can help us to curb the spread of the pandemic. A potent and rapid way of testing can help us in this challenge a lot. Thus, we should develop an effective testing service. Although the tests are available, still the method of physical contact with the test subject is inevitable. Thus, in this project, we aim to develop a contact-less yet efficient way to detect the infection using image processing on the X-Ray lung scan. Such a method helps us harness the existing framework to the fullest for rapid testing of the population. And since the process is contactless, it also assures the safety of the staff. The results are automated, the medical personnel can focus on more challenging aspects of the pandemic, as in the more effective and cautious treatment of an individual. Image processing is one of the effective ways to detect any anomaly in the human body. The CoVid diagnosis consists of performing an X-Ray on a patient and the fibrosis and shadowing of the lungs, due to insufficient oxygen supply, is observed. The severity of the symptoms can help us to classify the gravity of a health emergency.

KEYWORDS - coronavirus; COVID-19; deep learning; convolution neural network; X-Ray;

I. INTRODUCTION

A pandemic is a disease globally affecting several populations. The globe has witnessed many pandemics within the twentieth century. The stream of contagious disease viruses is the main reason for pandemics. These viruses show ever-changing behaviour with the ever-changing seasons. Thus, predicting the action of such viruses is a formidable step to prevention. Health professionals typically create correct predictions concerning most viruses. But some viruses have such unusual behaviour. Thus, the square measure is troublesome to predict. Such viruses cause pandemics as humans do not have the immunity to resist such viruses. The latest coronavirus malady called COVID-19 has appeared and unfolds very quickly. Since its discovery in Dec 2019 in a metropolis, China, the infection has already met 199 countries and territories. The Severe Acute metabolism Syndrome Coronavirus two (SARS-CoV-2) causes COVID-19 [1]. The virus could be an RNA (RNA) virus from the Coronavirus family, the family from which many viruses cause respiratory illness. The added severe variety of coronaviruses is Severe Acute metabolism Syndrome Coronavirus (SARS-CoV) and Middle EastRespiratory Syndrome Coronavirus (MERS-CoV). COVID-19 can cause metabolism ailments starting from the common cold to fatal diseases like respiratory disorder. The number of cases worldwide has reached 5,817,385 resulting in the deaths of 362,705 people as of May 30 2020 as per a report revealed by the World Health Organization (WHO) [2]. The correct information concerning the emergence of COVID-19 is still unknown. However, the initial cases have established links with the Huanan (Southern China) food Wholesale Market [3, 4]. The infection is contagious, and the virus unfolds amongst humans via metabolism droplets, physical contact, and additionally through faecal-oral transmission [5]. Various cases of respiratory disorder of unknown cause were rumoured in a metropolis, China in December 2019. The patients showed similar clinical characteristics with infectious agent pneumonia [6]. The patients suffering from COVID-19 infection square measure were discovered to have severe respiratory disorders with abnormal observations on chest computerized axial tomography (CT) examination [7]. The inconvenience of medication for this infection needs efficient identification strategies to dominate the malady. Common cold to respiratory disorder is caused by a gaggle of viruses called CoV. These diseases embody respiratory, enteric, renal, and medicine diseases. These viruses are classified into four genres particularly alpha-CoV, beta-CoV, gamma-CoV, and delta-CoV [8]. The virus affects people of all age teams and genders. A research study reveals that two teams of people square measure specifically full of this infection. The first group of people squares measures people who square measure higher than sixty years old. The second cluster is of these people. The United Nations agency has some underlying medical conditions like polygenic disease, cardiovascular infection, and cardiovascular disease. The common symptoms of COVID-19 embody fever, dry cough, and respiratory issues like shortness of breath, muscular soreness, and

fatigue. In some patients, symptoms and vomiting also are rumoured. The severity of the infection ranges from delicate contagious disease to respiratory disorder inflicting metabolism ailments. The advanced stage of the infection can even cause organ failures and Acute metabolism Distress Syndrome (ARDS) resulting in the deaths of the patients. The fast human to human transmission of the disease could be a matter of subtle concern for the regulative authorities globally. The management of COVID-19 mostly depends on the identification at the proper time. The offered methods for identification comprise of laboratory tests like Reverse-Transcription enzyme Chain Reaction (RT-PCR), a period RT-PCR (RRT-PCR), and Reverse Transcription Loop-mediated equal Amplification (RT-LAMP) test [9, 10]. The laboratory tests have some limitations. Firstly, the check needs testing kits that have restricted handiness within the provided chain. Secondly, the test needs time overwhelming thanks to the laboratory processes concerned. The X-Ray facilities are accessible easily worldwide and the results are square measure also made at a quick pace. Therefore, the chest XRay pictures are also used for police work in the presence of COVID-19. An automatic method supporting chest X-Ray pictures for support in clinical deciding is vital for disease management. In line with the United Nations agency, the infection can be controlled by stopping the chain of transmission. The officials have rumoured that testing and isolation square measure the two key actions that square measure help break the chain of transmission. Therefore, the correct identification is significant in dominant COVID-19. The detection of COVID-19 may be done at the AN earlier stage with chest pictures compared to the PCR testing. The chest X-Ray pictures may be analyzed by victimization artificial intelligence techniques [11]. Numerous techniques for identification of COVID-19 using machine learning techniques on imaging square measure are offered within the literature. A transfer learning model for the identification of coronavirus from chest X-Ray images is given in referee [12]. Another technique with improved accuracy given a segmentation-based approach. The strategy classified the input pictures as simple, virus infection, and COVID-19 [13]. A deep learning-based model is applied to CT pictures for the detection of COVID-19. Some researchers have additionally developed public datasets consisting of chest X-Ray images of COVID-19 patients [14, 15]. a technique named COVID-Net is developed and applied on these public datasets for identification of COVID-19 [14]. The employment of deep learning for identification from the chest X-Ray pictures provides rapid results. Deep learning models' square measures are being used widely for medical image processing. In [16], the detection of the respiratory disorder is concluded in victimization convolution neural networks. During this paper, an automatic technique for the identification of COVID-19 from a deep network is projected. The projected network utilizes the feature generated by multiresolution analysis. The mix of wave transforms alongside the deep network brings multiple benefits. The wave decomposition is fed into the network. The network used isn't

the traditional Convolutional Neural Network (CNN). A depthwise divisible network is employed during this work.

II. BACKGROUND

In this section, we will discuss some basics about the convolutional neural network and some basic concepts to know before the implementation of the same via software code.

A. The Convolutional Neural Network (ConvNet) -

In the field of deep learning, the convolutional neural network is used for visual imaging processes. This neural network algorithm is inspired by the animal cortex. The animal cortex captures the image via the eye and then the brain interprets the image. For example, the eye is a convex lens that captures the image as an inverted virtual image and then the brain inverts it again to make it upright and then we interpret it based on past experiences.

CNN uses the same sort of algorithm. The images are pre-processed and then interpreted based on past data or labelled data. CNN's are regularized versions of multilayer perceptrons. Multilayer perceptrons usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "full connectivity" of these networks makes them prone to overfitting data. Typical ways of regularization, or preventing overfitting, include: penalizing parameters during training (such as weight decay) or trimming connectivity (skipped connections, dropout, etc.) CNN's take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble patterns of increasing complexity using smaller and simpler patterns embossed in their filters. Therefore, on a scale of connectivity and complexity, CNNs are on the lower extremity.

B. The building blocks of CNN

A convolutional neural organization comprises an information layer, covered up layers and a yield layer. In any feed-forward neural organization, any center layers are called covered up in light of the fact that their information sources and yields are concealed by the initiation capacity and last convolution. In a convolutional neural organization, the secret layers incorporate layers that perform convolutions. Normally this incorporates a layer that plays out a speck result of the convolution portion with the layer's info framework.

This item is generally the Frobenius internal item, and its initiation work is usually ReLU. As the convolution bit slides along the information grid for the layer, the convolution activity produces an element map, which thus adds to the contribution of the following layer. This is trailed by different layers, for example, pooling layers, completely associated layers, and standardization layers.

1) Convolutional layers

In a CNN, the information is a tensor with a shape: (number of data sources) x (input stature) x (input width) x (input channels). In the wake of going through a convolutional layer, the picture gets disconnected to a component map, likewise called an initiation map, with shape: (number of information sources) x (highlight map height) x (include map width) x (highlight map channels). A convolutional layer inside a CNN by and large has the accompanying ascribes:

- Convolutional channels/bits characterized by a width and tallness (hyper-boundaries).

- The quantity of info channels and yield channels (hyper-boundaries). One layer's info channels should rise to the quantity of yield channels (likewise called profundity) of its information.

- Extra hyperparameters of the convolution activity, like cushioning, step, and enlargement.

Convolutional layers convolve the information and pass its outcome to the following layer. This is like the reaction of a neuron in the visual cortex to a particular stimulus.[17]. Each convolutional neuron measures information just for its open field. Albeit completely associated feedforward neural organizations can be utilized to learn includes and characterize information, this engineering is for the most part unrealistic for bigger data sources like high-goal pictures. It would require an extremely high number of neurons, even in shallow engineering, because of the enormous information size of pictures, where every pixel is an applicable information inclusion. For example, a completely associated layer for a (little) picture of size 100 x 100 has 10,000 loads for every neuron in the subsequent layer. All things being equal, convolution lessens the quantity of free boundaries, permitting the organization to be more profound [18]. For instance, paying little mind to picture size, utilizing a 5 x 5 tiling area, each with similar shared loads, requires just 25 learnable boundaries.

Using regularized weights over fewer parameters avoids the vanishing gradients and exploding gradients problems seen during backpropagation in traditional neural networks [19,20] Furthermore, convolutional neural networks are ideal for data with a grid-like topology (such as images) as spatial relations between separate features are taken into account during convolution and/or pooling.

2) Pooling layers

Convolutional organizations may incorporate nearby or potentially worldwide pooling layers alongside conventional convolutional layers. Pooling layers decrease the components of information by consolidating the yields of neuron bunches at one layer into a solitary neuron in the following layer. Nearby pooling joins little bunches, tiling sizes, for example, 2 x 2 are usually utilized. Worldwide pooling follows up on every one of the neurons of the element map.[21,22] There are two normal sorts of pooling in well known use: max and normal. Max pooling utilizes the greatest worth of every neighborhood bunch of neurons in the component map, [23,24] while normal pooling takes the normal worth.

3) Fully connected layers

Completely associated layers interface each neuron in one layer to each neuron in another layer. It is equivalent to a conventional multi-facet perceptron neural organization (MLP). The leveled grid goes through a completely associated layer to order the pictures.

4) Receptive field

In neural organizations, every neuron gets a contribution from some number of areas in the past layer. In a convolutional layer, every neuron gets contribution from just a limited space of the past layer called the neuron's responsive field. Regularly the territory is a square (for example 5 by 5 neurons). Though, in a completely associated layer, the open field is the whole past layer. In this manner, in each convolutional layer, every neuron takes contribution from a bigger region in the contribution than past layers. This is expected to apply the convolution again and again, which considers the worth of a pixel, just as its encompassing pixels. When utilizing expanded layers, the quantity of pixels in the responsive field stays consistent, yet the field is all the more meagerly populated as its measurements develop when joining the impact of a few layers.

5) Weights

Every neuron in a neural organization figures a yield esteem by applying a particular capacity to the info esteems obtained from the open field in the past layer. The capacity that is applied to the info esteems is controlled by a vector of loads and a predisposition (commonly genuine numbers). Learning consists of iteratively changing these inclinations and loads.

The vector of loads and the predisposition are called channels and address specific highlights of the info (e.g., a specific shape). A distinctive component of CNNs is that numerous neurons can have a similar channel. This diminishes the memory impression in light of the fact that a solitary inclination and a solitary vector of loads are utilized across all responsive fields that share that channel, instead of each open field having its own predisposition and vector weighting. [25]

C. Basic mathematical formulae governing the neural network

Portion convolution isn't just utilized in CNN's but on the other hand is a vital component of numerous other Computer Vision calculations. It is an interaction where we take a little network of numbers (called bit or channel), ignore our picture and change it depending on the qualities from the channel. Resulting highlight map esteems are determined by the accompanying equation, where the information picture is meant by f and our portion by h. The files of lines and segments of the outcome framework are set apart with m and n individually.

$$\mathbf{G[m, n] = (f * h)[m, n] = \sum_{j}\sum_{k} h[j, k]\, f[m - j, n - k]} \qquad \mathbf{1)}$$

Subsequent to putting our channel over a chosen pixel, we take each worth from the piece and increase them two by two with relating values from the picture. At last, we summarize everything and put the outcome in the correct spot in the yield highlight map. Above we can perceive how such an activity looks on a miniature size, yet the thing is significantly really fascinating, is the thing that we can accomplish by performing it on a full picture.

Since our picture recoils each time we perform convolution, we can do it just a predetermined number of times, before our picture vanishes totally. Also, in the event that we take a gander at how our piece travels through the picture we see that the effect of the pixels situated on the edges is a lot more modest than those in the focal point of the picture.

To take care of both of these issues we can cushion our picture with an extra line. For instance, on the off chance that we utilize 1px cushioning, we increment the size of our photograph to 8x8, so the yield of the convolution with the 3x3 channel will be 6x6. Generally, by and by, we fill in extra cushioning with zeros. Contingent upon if we use cushioning, we are managing two sorts of convolution — Valid and Same. Naming is very tragic, so for clarity: Valid — implies that we utilize the first picture, Same — we utilize the line around it so the pictures at the information and yield are a similar size. In the subsequent case, the cushioning width should meet the accompanying condition, where p is cushioning and f is the channel measurement (typically odd).

$$\mathbf{p} = \frac{f-1}{2} \qquad \mathbf{2)}$$

These days, we don't have to waste time with backpropagation — profound learning systems do it for us, yet I feel it merits knowing what's happening in the engine. Very much like in thickly associated neural organizations, we will probably compute subordinates and later use them to refresh the upsides of our boundaries in a cycle called inclination drop.

In our estimations, we will utilize a chain rule - which I referenced in past articles. We need to survey the impact of the adjustment of the boundaries on the subsequent highlights map, and consequently on the eventual outcome. Before we begin to delve into the subtleties, let us concede to the numerical documentation that we will utilize - to make my life simpler, I will relinquish the full documentation of the fractional subordinate for the abbreviated one noticeable underneath. In any case, recollect that when I utilize this documentation, I will consistently mean the halfway subordinate of the expense work.

$$\mathbf{dA^{[l]}} = \frac{\partial L}{\partial A^{[l]}} \quad \mathbf{dZ^{[l]}} = \frac{\partial L}{\partial Z^{[l]}} \quad \mathbf{dW^{[l]}} = \frac{\partial L}{\partial W^{[l]}} \quad \mathbf{db^{[l]}} = \frac{\partial L}{\partial b^{[l]}} \qquad \mathbf{3)}$$

III. PROPOSED METHODOLOGY

The proposed method will have the following steps -

A. Metadata visualization

First of all, we will dive into the data present in the form of metadata. A script can be exclusively created for visualization of the type of data present. This script shows the various fields in the metadata, such as patient name, country of origin, findings via X-Ray, the axis of X-Ray, etc. These data can be visually presented in the form of pie charts or bar graphs.

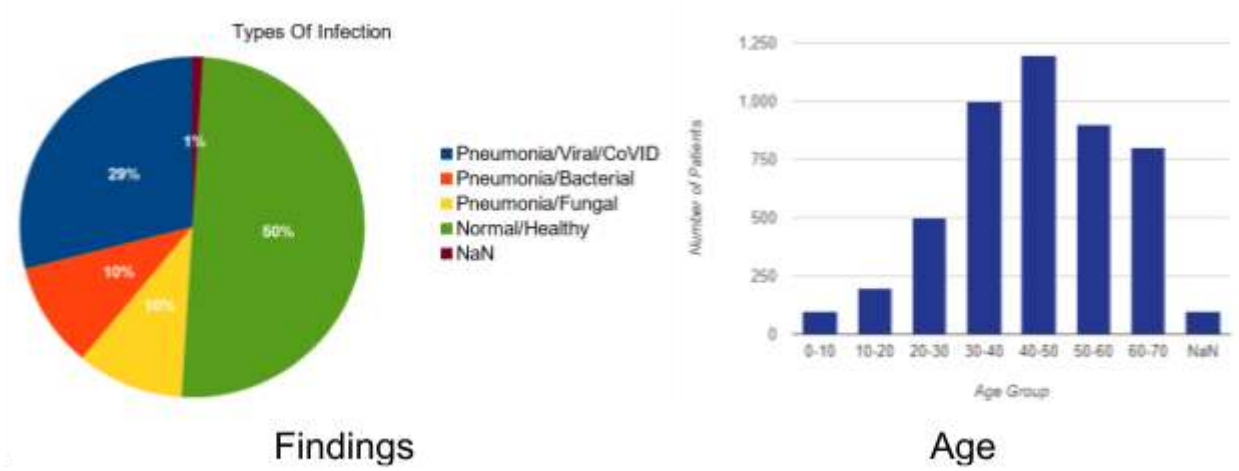


Figure 1 - Sample Distribution Charts

B. Data Cleaning

The next step is to remove the images that have no findings. Such images can hinder the processing capabilities of the neural network. These images can be removed with the help of a script that can go through the entries with ‘no findings’ and can remove the same cols and the images from the dataset. Also, the same can be repeated for the cols with other missing data entries, but since our model just predicts the infection based on the image itself, we can ignore the rest of the other missing values.

C. Preliminary data segregation

In this step, we will be extracting the required images by the means of an automated script. The script will select images based on the type of findings and the type of X-Ray view mentioned in the metadata file. This script will look for all those images in the main repository data and then store them into a custom output directory.

D. Model construction

Our model consists of four convolution 2D layers, two pooling layers and one flattening layer. The convolution 2D layers use 'ReLU' as an activation function, whereas the flattening layer uses the sigmoid function for the same. Now, we will look into each building block of the model and discuss their significance -

1) The Convolution 2D layer

The convolution neural network layer is usually represented by (size_of_feature_map, number_of_channels_in_feature_map, size_of_image). Here, in this model, we will be using the image size of 150 x 150 and the number of feature map channels as 3.

$$\boldsymbol{conv}\ (\mathbf{I}, \mathbf{K})_{\mathbf{x,\,y}} = \sum_{i=1}^{n_H} \sum_{j=1}^{n_W} \sum_{k=1}^{n_C} \mathbf{K_{i,\,j,\,k}}\ \mathbf{I_{x\,+\,i\,-\,1,\,y\,+\,j\,-\,1,\,k}} \qquad \mathbf{4)}$$

Where I denotes the image and K denotes the kernel.
i,j,k denote the coordinates of the image
n_H, n_W, n_C denote the height, width and channels of the feature map respectively.

$$\boldsymbol{dim}\ (\boldsymbol{conv}\ (\mathbf{I}, \mathbf{K})) = \left(\left[\frac{n_H + 2p - f}{s} + 1\right], \left[\frac{n_W + 2p - f}{s} + 1\right]\right);\ \mathbf{s > 0}$$
$$= (\mathbf{n_H + 2p - f,\ n_W + 2p - f\ });\ \mathbf{s = 0} \qquad \mathbf{5)}$$

where [x] is the floor function of x

There are some special types of convolution:

- **Valid convolution:** p = 0
- **Same convolution:** p = $\frac{f-1}{2}$
- **1 ×1 convolution:** f = 1

Where p, f denotes the padding and the filter size respectively.
s is the stride index, which is the number of iterations taken to convolute the image

2) The activation function

The activation function in any layer of the model is used to determine the weights of the feature map so that it can determine the correct output. Here, we will be using two types of activation functions -

i) ReLU Function

The ReLU function is a simple function which purely returns the maximum value from a given input and zero (for a = 0). It is generally used to eliminate all the negative weights from the feature map and get desirable predictions.

$$\mathbf{f(x) = max\ (x, ax)} \qquad \mathbf{6)}$$

ii) Sigmoid Function

This function is rather a complex one and it is given by the equation -

$$\mathbf{S(x) = 1 - S(-x) =} \frac{1}{1+e^{-x}} = \frac{e^{x}}{1+e^{x}} \qquad \mathbf{7)}$$

This function is used to diminish all the unnecessary features from the feature map.

3) Pooling Layer

Pooling layers are used to downsample the image features for faster processing of the image. The weights are pooled into a lower order matrix.

$$\textbf{\textit{dim}} \textbf{ (}\textbf{\textit{pooling}}\textbf{ (image)) = (}\left[\frac{n_H + 2p - f}{s} + 1\right], \left[\frac{n_W + 2p - f}{s} + 1\right], \mathbf{n_C)};$$

$$= (\mathbf{n_H + 2p - f, n_W + 2p - f, n_C}); \mathbf{s = 0} \qquad \textbf{8)}$$

4) Flattening Layer

The flattening layer as the name suggests converts higher dimensional feature map output given by the convolution network into a single value. The single value can be boolean (true or false), binary (0 or 1), or a set of integers depicting the final result interpreted from the image.

5) Densing Layer

The densing layer suggests the densely connected neurons. It means that all the neurons are given as input to the layer and the output is made fully connected to each neuron. Every densing layer is governed by an activation function. In general, consider j^{th} node of the i^{th} layer we have the following equations -

$$\mathbf{z_j^{[i]}} = \sum_{l=1}^{n_{i-1}} \mathbf{w_{j,l}^{[i]}}\ \mathbf{a_l^{[i-1]}} + \mathbf{b_j^{[i]}}$$

$$\mathbf{a_j^{[i]}} = \mathbf{\Psi^{[i]}}\ (\mathbf{z_j^{[i]}}) \qquad \textbf{9)}$$

The input $a^{[i-1]}$ might be the result of a convolution or a pooling layer with the dimensions ($n_H^{[i-1]}$, $n_W^{[i-1]}$, $n_C^{[i-1]}$). In order to be able to plug it into fully connected layer we flatten the tensor to a 1D vector having the dimension
($n_H^{[i-1]} \times n_W^{[i-1]} \times n_C^{[i-1]}$, 1), thus :

$$\mathbf{n_{i-1}} = \mathbf{n_H^{[i-1]} \times n_W^{[i-1]} \times n_C^{[i-1]}} \qquad \textbf{10)}$$

The learned parameters at the l^{th} layer are -

- **Weights ($\mathbf{w_{j,l}}$)** with $n_{l-1} \times n_l$ parameters.
- **Bias** with n_l parameters.

6) Dropout Layer

As the name suggests, it drops out some samples from the model. This step ensures that the model is not overfitted (any model with an accuracy greater than 97% is termed as an overfitted model). The overfitted model can result in hazy predictions. Here, we will summarise the Convolution network.

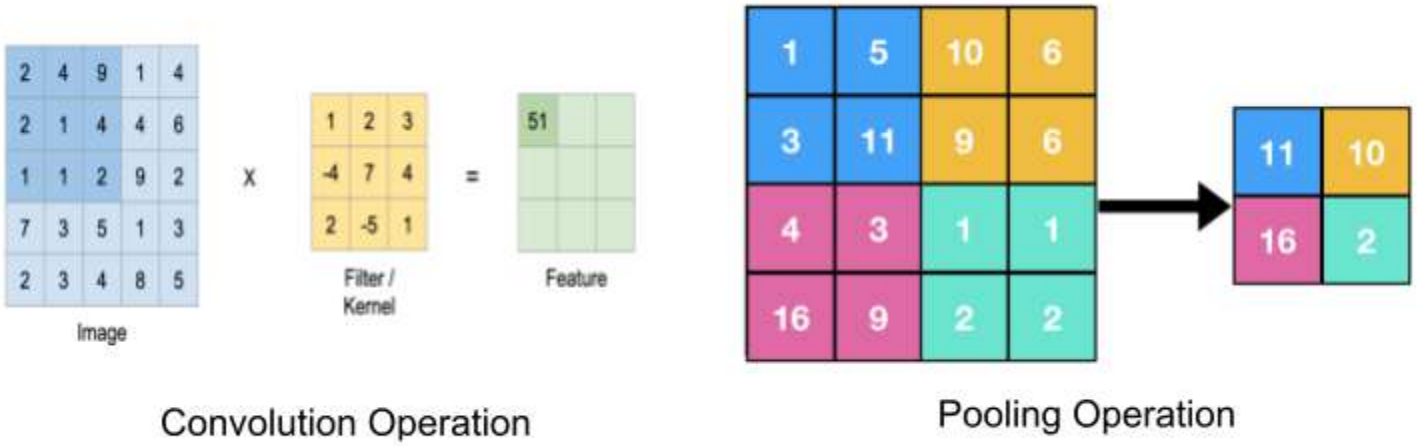


Figure 2 - The visual representation of various operations

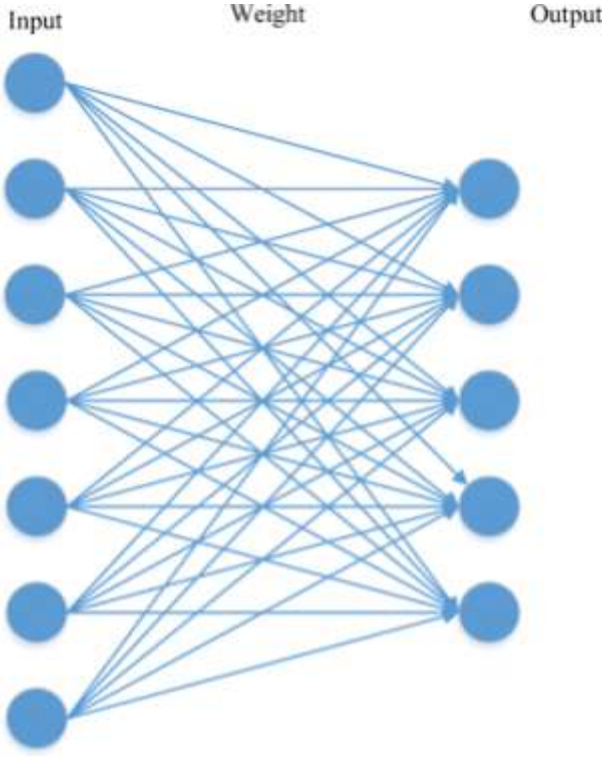


Figure 3 - The densing layer or fully connected layer

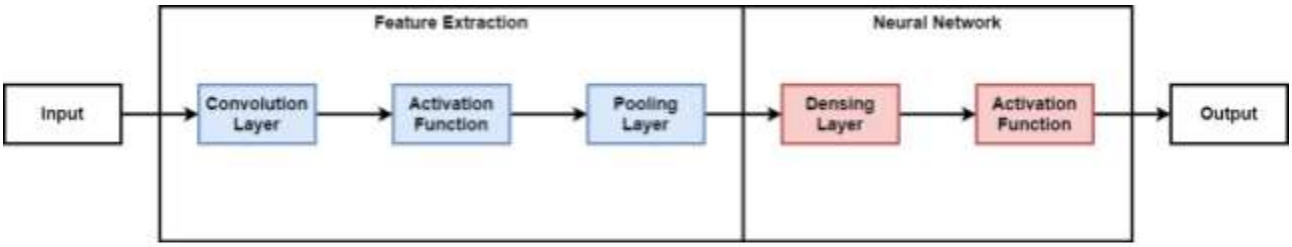


Figure 4 - Basic flowchart of CNN

IV. DATASET

The selected dataset has been gathered from sources [26]-[32]. The metadata along with the dataset was pre-processed with the script. The dataset was further truncated manually. The dataset contains 198 X-Ray images of CoVID patients and 636 images of normal X-Ray. A total of 834 images have been fully verified before training the model.
The 59 images from the CoVID dataset and 190 images from the normal dataset are used for testing purposes and the rest is used for training purposes.

Table 1 - Summary of Dataset

	Training (70%)	**Testing (30%)**
CoVID	139	59
Normal	446	190
Total	585	249

V. RESULTS AND DISCUSSION

The model was executed using the Keras framework and with the Tensorflow backend. The model summary, its general flow and all the experimental results are presented in this section. First of all, the model appends the images into two lists - covid_images and normal_images. The images are then displayed using the matplotlib function.

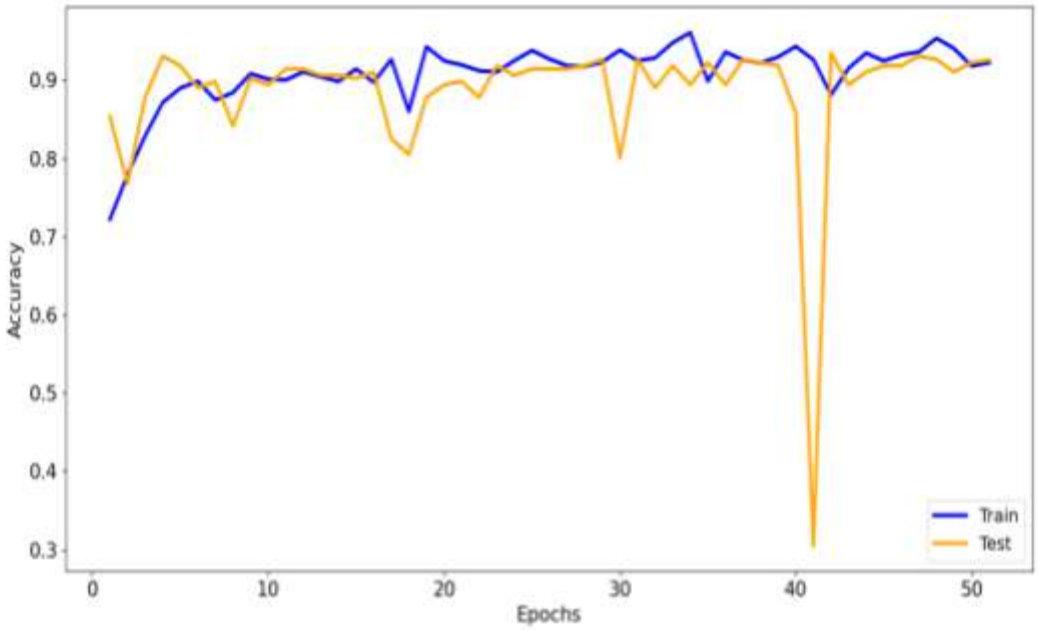
Next, the model was deployed using the keras framework. Here, is the model summary:

Normal X-Ray CoVID Patient X-Ray

Figure 5 - Images used for training

Table 2 - Summary of Model				
Layer	**Activation Function Or Dropout**	**Shape**	**Number of Channels**	**Parameters**
Conv2D_1	ReLU	(148,148,72)	3	896
Pooling Layer	-	(74,74,32)	-	0
Conv2D_2	ReLU	(72,72,64)	3	18496
Pooling Layer	-	(36,36,64)	-	0
Conv2D_3	ReLU	(36,36,128)	3	73856
Conv2D_4	ReLU	(36,36,256)	3	295168
Flatten	-	262144	-	0
Dense_1	25%	32	-	8388640
Dense_2	Sigmoid	1	-	33
Total Parameters - 8,777,089				
Trainable Parameters - 8,777,089				

The dataset was split into 70% for training and 30% for testing. The model was then trained for 50 epochs for a batch size of 6. Overall, it took over an average of 35 seconds per epoch for



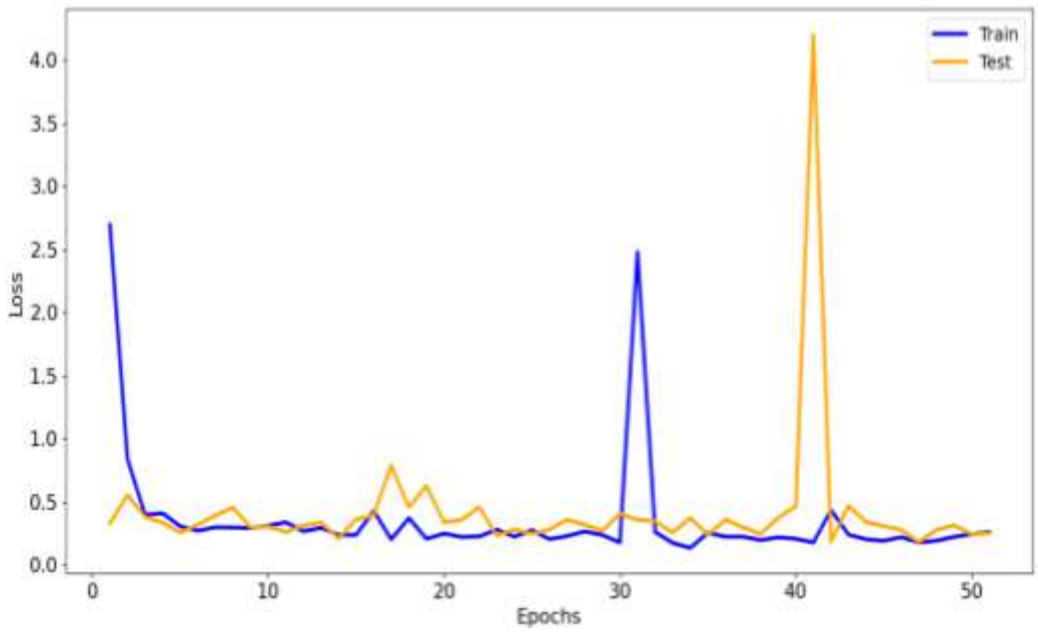


Figure 6 - Accuracy and loss plotted over epochs

training. The model accuracy and value accuracy, along with model loss along with value loss was plotted using the script.

The confusion matrix of the model was obtained as follows -

Table 3 - Confusion Matrix of the Model		
Condition	**Predicted**	**Actual**
CoVID	59	59
Normal	190	190
Total	249	249

The model performance is shown as follows -

Table 4 - Performance of the Model				
Condition	**Accuracy**	**Precision**	**Sensitivity**	**F1 Score**
CoVID	92.9 %	96	95	94
Normal	92.9 %	96	95	94

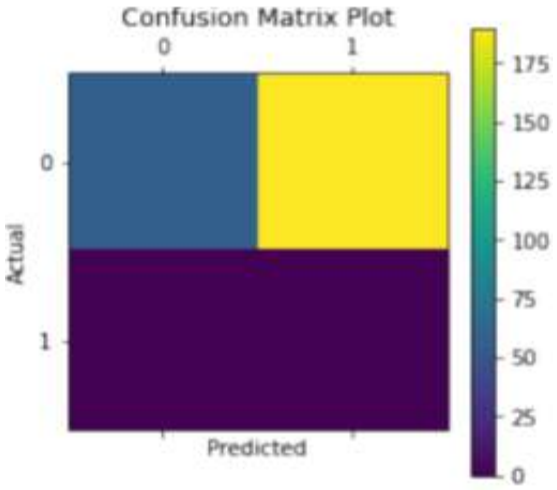


Figure 7 - Confusion Matrix plotted by script

As it can be seen from the plots and the other parameters, the model is pretty accurate and also has a lesser number of parameters to be learnt. The testing and training accuracy and loss tends to coincide as the model training is coming to conclusion.

VI. CONCLUSION

This project has developed a model to accurately differentiate between the CoVID X-Ray image and normal X-Ray. This model can be pretty useful as the training period of the model is pretty less and it can still give accurate results. The model is proven effective against most of the previous projects. Here, is the comparison of all previous models with our proposed model -

Table 5 - Comparative Analysis of the Model

Algorithm	Accuracy	Precision	Sensitivity	F1 Score
[33]	83	88	87	86
[34]	82%	87	86	85
[35]	87%	91	90	89
[36]	89%	93	92	91
Proposed	**92.9 %**	**96**	**95**	**94**

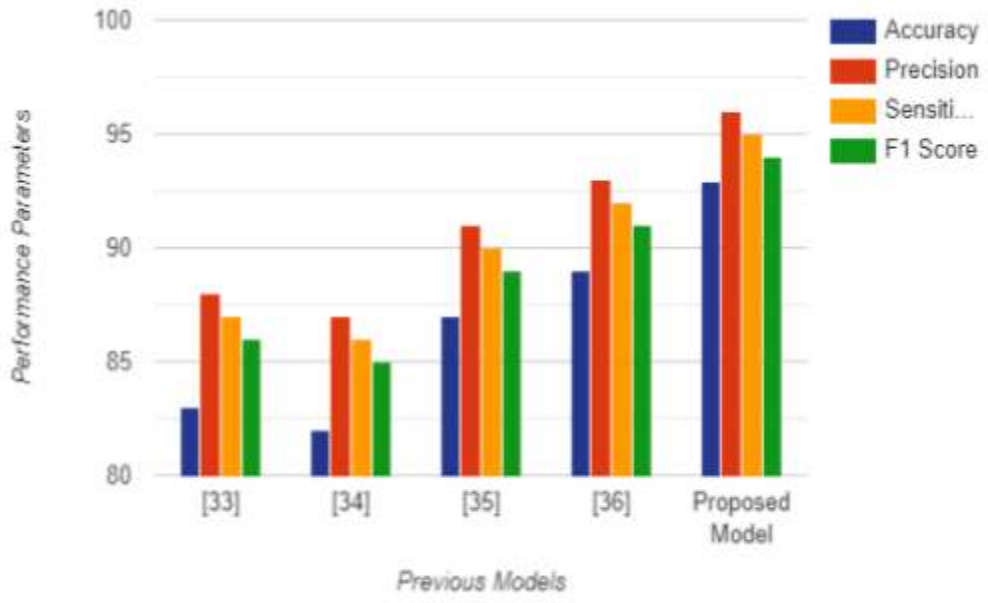


Figure 8 - Comparative Analysis of Model

Here is the flowchart summarising our model -

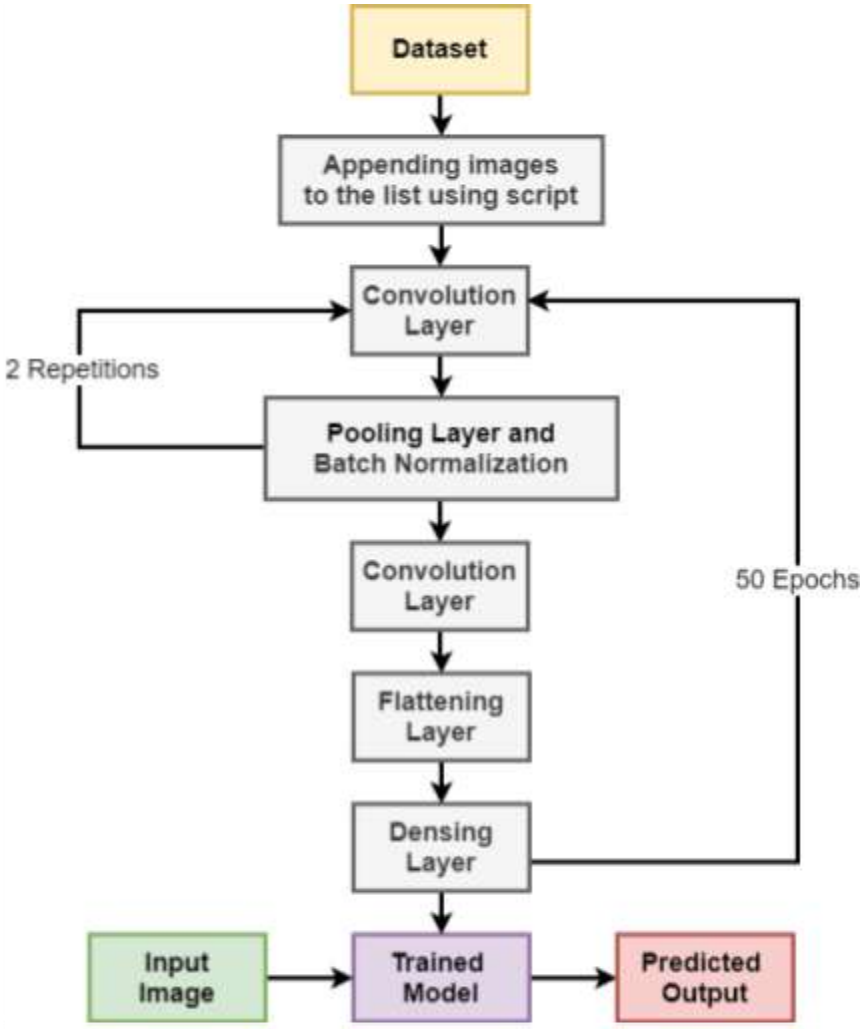


Figure 9 - The flowchart of proposed model

References –

[1] A. C. Walls, Y. J. Park, M. A. Tortorici, A. Wall, A.T. McGuire, and D. Veesler, Structure, function, and antigenicity of the SARS-CoV-2 spike glycoprotein, Cell, vol. 181, no. 2, pp. 281–292, 2020.

[2] World Health Organization, Coronavirus Disease 2019 (COVID-19), Situation Report–81,https://www.who.int/docs/default-source/coronaviruse/situationreports/20200410-sitrep-81-covid-19.pdf, 2020.

[3] N. S. Chen, M. Zhou, X. Dong, J. M. Qu, F. Y. Gong, Y. Han, Y. Qiu, J. L. Wang, Y. Liu, Y. Wei, et al., Epidemiological and clinical characteristics of 99 cases of 2019 novel coronavirus pneumonia in Wuhan, China: A descriptive study, Lancet, vol. 395, no. 10 223, pp. 507–513, 2020.

[4] Q. Li, X. H. Guan, P.Wu, X. Y.Wang, L. Zhou, Y. Q. Tong, R. Q. Ren, K. S. M. Leung, E. H. Y. Lau, J. Y.Wong, et al., Early transmission dynamics in Wuhan, China, of novel coronavirus–infected pneumonia, New England Journal of Medicine, vol. 382, no. 13, pp. 1199–1207, 2020.

[5] J. F. W. Chan, S. F. Yuan, K. H. Kok, K. K. W. To, H. Chu, J. Yang, F. F. Xing, J. L. Liu, C. C. Y. Yip, R. W. S. Poon, et al., A familial cluster of pneumonia associated with the 2019 novel coronavirus indicating person-to-person transmission: A study of a family cluster, Lancet, vol. 395, no. 10 223, pp. 514–523, 2020.

[6] D. W. Wang, B. Hu, C. Hu, F. F. Zhu, X. Liu, J. Zhang, B.B. Wang, H. Xiang, Z. S. Cheng, Y. Xiong, et al., Clinical characteristics of 138 hospitalized patients with 2019 novel coronavirus–infected pneumonia in Wuhan, China, JAMA, vol. 323, no. 11, pp. 1061–1069, 2020.

[7] C. L. Huang, Y. M. Wang, X. W. Li, L. L. Ren, J. P.Zhao, Y. Hu, L. Zhang, G. H. Fan, J. Y. Xu, X. Y. Gu, et al., Clinical features of patients infected with 2019 novel coronavirus in Wuhan, China, Lancet, vol. 395, no. 10 223, pp. 497–506, 2020.

[8] X. Y. Ou, Y. Liu, X. B. Lei, P. Li, D. Mi, L. L. Ren, L. Guo, R. X. Guo, T. Chen, J. X. Hu, et al., Characterization of spike glycoprotein of SARS-CoV-2 on virus entry and its immune cross-reactivity with SARS-CoV, Nature Communications, vol. 11, p. 1620, 2020.

[9] P. Zhai, Y. B. Ding, X. Wu, J. K. Long, Y. J. Zhong, and Y. M. Li, The epidemiology, diagnosis and treatment of COVID-19, International Journal of Antimicrobial Agents, vol. 55, no. 5, p. 105 955, 2020.

[10] X. Z. Xie, Z. Zhong, W. Zhao, C. Zheng, F. Wang, and J. Liu, Chest CT for typical 2019-nCoV pneumonia: Relationship to negative RT-PCR testing, Radiology, vol. 296, no. 2, p. 200 343, 2020.

[11] D. S. Kermany, M. Goldbaum,W. J. Cai, C. C. S. Valentim, H. Y. Liang, S. L. Baxter, A. McKeown, G. Yang, X. K. Wu, F. B. Yan, et al., Identifying medical diagnoses and treatable diseases by image-based deep learning, Cell, vol. 172, no. 5, pp. 1122–1131, 2018.

[12] S. Wang, B. Kang, J. L. Ma, X. J. Zeng, M. M. Xiao, J. Guo, M. J. Cai, J. Y. Yang, Y. D. Li, X. F. Meng, et al., A deep learning algorithm using CT images to screen for CoronaVirus Disease (COVID-19), https://doi.org/10.1101/2020.02.14.20023028, 2020.

[13] X. W. Xu, X. G. Jiang, C. L. Ma, P. Du, X. K. Li, S. Z. Lv, L. Yu, Y. F. Chen, J. W. Su, G. J. Lang, et al., Deep learning system to screen coronavirus disease 2019 pneumonia, arXiv preprint arXiv: 2002.09334, 2020.

[14] J. P. Cohen, P. Morrison, and L. Dao, COVID-19 image data collection, arXiv preprint arXiv: 2003.11597, 2020.

[15] O. Gozes, M. Frid-Adar, H. Greenspan, P. D. Browning, H. Q. Zhang, W. B. Ji, A. Bernheim, and E. Siegel, Rapid AI development cycle for the coronavirus (COVID-19)
pandemic: Initial results for automated detection & patient monitoring using deep learning CT image analysis, arXiv preprint arXiv: 2003.05037, 2020.

[16] D. Varshni, K. Thakral, L. Agarwal, R. Nijhawan, and A. Mittal, Pneumonia detection using CNN based feature extraction, in Proc. 2019 IEEE Int. Conf. Electrical, Computer and Communication Technologies (ICECCT), Coimbatore, India, 2019, pp. 1–7.
[17] "Convolutional Neural Networks (LeNet) – DeepLearning 0.1 documentation". *DeepLearning 0.1*. LISA Lab. Retrieved 31 August 2013.

[18] Habibi, Aghdam, Hamed (2017-05-30). *Guide to convolutional neural networks: a practical application to traffic-sign detection and classification*. Heravi, Elnaz Jahani. Cham, Switzerland.

[19] Venkatesan, Ragav; Li, Baoxin (2017-10-23). Convolutional Neural Networks in Visual Computing: A Concise Guide. CRC Press

[20] Balas, Valentina E.; Kumar, Raghvendra; Srivastava, Rajshree (2019-11-19). *Recent Trends and Advances in Artificial Intelligence and the Internet of Things*. Springer Nature

[21] Ciresan, Dan; Ueli Meier; Jonathan Masci; Luca M. Gambardella; Jurgen Schmidhuber (2011). "Flexible, High-Performance Convolutional Neural Networks for Image Classification" (PDF). *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence-Volume Volume Two*. **2**: 1237–1242.

[22] Krizhevsky, Alex. "ImageNet Classification with Deep Convolutional Neural Networks"

[23] Yamaguchi, Kouichi; Sakamoto, Kenji; Akabane, Toshio; Fujimoto, Yoshiji (November 1990). *A Neural Network for Speaker-Independent Isolated Word Recognition*. First International Conference on Spoken Language Processing (ICSLP 90). Kobe, Japan.

[24] Ciresan, Dan; Meier, Ueli; Schmidhuber, Jürgen (June 2012). *Multi-column deep neural networks for image classification*. *2012 IEEE Conference on Computer Vision and Pattern Recognition*. New York, NY: Institute of Electrical and Electronics Engineers (IEEE). pp. 3642–3649.

[25] LeCun, Yann. "LeNet-5, convolutional neural networks"

[26] Henry Z. Lo. and Cohen, Joseph Paul “Academic Torrents: Scalable Data Distribution.” Neural Information Processing Systems Challenges in Machine Learning (CiML) Workshop, 2016, http://arxiv.org/abs/1603.04395.

[27] Cohen, Joseph Paul, and Henry Z. Lo. “Academic Torrents: A Community-Maintained Distributed Repository.” Annual Conference of the Extreme Science and Engineering

[28] Discovery Environment, 2014, http://doi.org/10.1145/2616498.2616528.

[29] Cohen, J. P., Morrison, P., Dao, L., Roth, K., Duong, T. Q., & Ghassemi, M. (2020). Covid-19 image data collection: Prospective predictions are the future. arXiv preprint arXiv:2006.11988.

[30] Paiva, O., 2020. CORONACASES.ORG – Helping Radiologists To Help People In More Than 100 Countries! | Coronavirus Cases – 冠状病毒病例. [online] Coronacases.org.

[31] Glick, Y., 2020. Viewing Playlist: COVID-19 Pneumonia Radiopaedia.Org. Radiopaedia.org.

[32] Ma Jun, Ge Cheng, Wang Yixin, An Xingle, Gao Jiantao, Yu Ziqi, … He Jian. (2020). COVID-19 CT Lung and Infection Segmentation Dataset (Version 1.0) [Data set]. Zenodo.

[33] Shi, Feng, et al. "Large-scale screening to distinguish between COVID-19 and community-acquired pneumonia using infection size-aware classification." *Physics in Medicine & Biology* 66.6 (2021): 065031.

[34] Fang, Mengjie, et al. "CT radiomics can help screen the coronavirus disease 2019 (COVID-19): a preliminary study." *Science China Information Sciences* 63.7 (2020): 1-8.

[35] Wang, Shuo, et al. "A fully automatic deep learning system for COVID-19 diagnostic and prognostic analysis." *European Respiratory Journal* 56.2 (2020).

[36] Li, Lin, et al. "Artificial intelligence distinguishes COVID-19 from community acquired pneumonia on chest CT." *Radiology* (2020).

Metadata_Visualization

May 11, 2021

1 Understand the type of data contained

[1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
# import data
metadata_dir = '../metadata.csv'
metadata = pd.read_csv(metadata_dir)
# what info do we have?
print(metadata.columns)
```

```
Index(['patientid', 'offset', 'sex', 'age', 'finding', 'RT_PCR_positive',
       'survival', 'intubated', 'intubation_present', 'went_icu', 'in_icu',
       'needed_supplemental_O2', 'extubated', 'temperature', 'pO2_saturation',
       'leukocyte_count', 'neutrophil_count', 'lymphocyte_count', 'view',
       'modality', 'date', 'location', 'folder', 'filename', 'doi', 'url',
       'license', 'clinical_notes', 'other_notes', 'Unnamed: 29'],
      dtype='object')
```

[2]:
```
print(metadata.head(10))
```

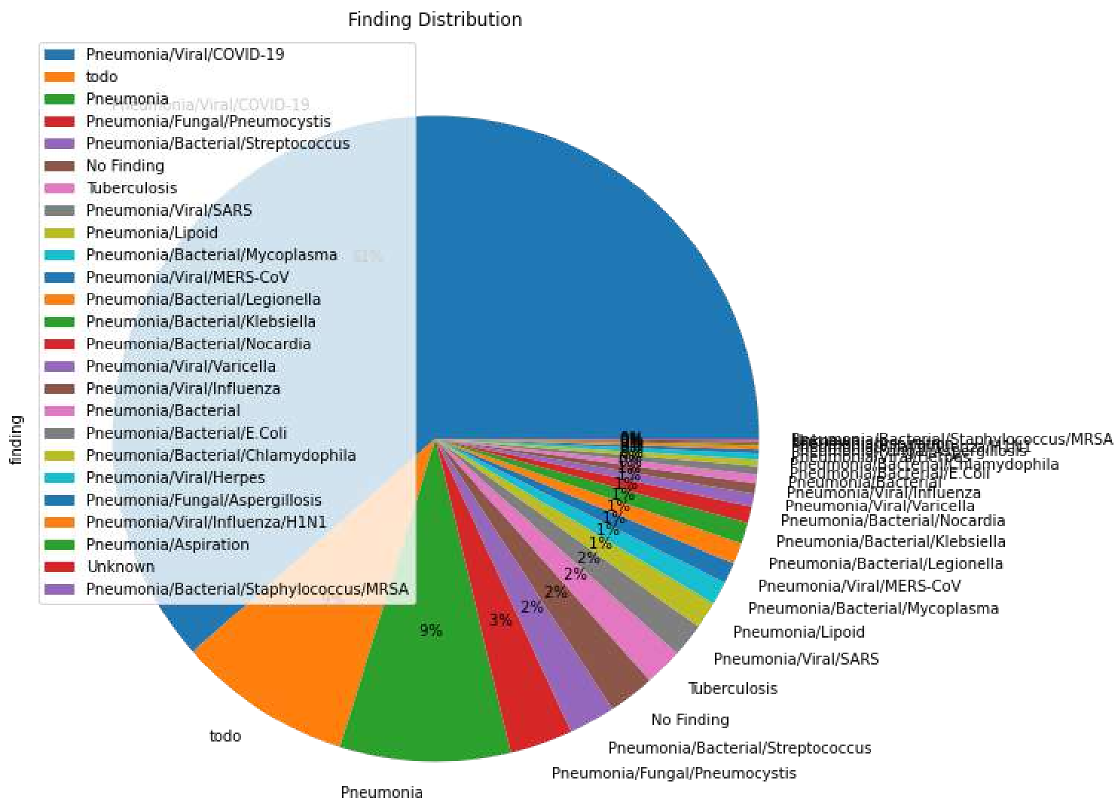
```
  patientid  offset  sex   age                   finding RT_PCR_positive  \
0         2     0.0    M  65.0  Pneumonia/Viral/COVID-19               Y
1         2     3.0    M  65.0  Pneumonia/Viral/COVID-19               Y
2         2     5.0    M  65.0  Pneumonia/Viral/COVID-19               Y
3         2     6.0    M  65.0  Pneumonia/Viral/COVID-19               Y
4         4     0.0    F  52.0  Pneumonia/Viral/COVID-19               Y
5         4     5.0    F  52.0  Pneumonia/Viral/COVID-19               Y
6         5     NaN  NaN   NaN                 Pneumonia             NaN
7         6     0.0  NaN   NaN  Pneumonia/Viral/COVID-19               Y
8         6     4.0  NaN   NaN  Pneumonia/Viral/COVID-19               Y
9         3     4.0    M  74.0      Pneumonia/Viral/SARS             NaN

  survival intubated intubation_present went_icu  …              date  \
0        Y         N                  N        N  …  January 22, 2020
1        Y         N                  N        N  …  January 25, 2020
2        Y         N                  N        N  …  January 27, 2020
3        Y         N                  N        N  …  January 28, 2020
```

```
4      NaN         N                  N        N  …  January 25, 2020
5      NaN         N                  N        N  …  January 30, 2020
6      NaN         Y                  Y        Y  …              2017
7      NaN         Y                  Y        Y  …   January 6, 2020
8      NaN         Y                  Y        Y  …  January 10, 2020
9        N       NaN                NaN      NaN  …              2004

                                              location  folder  \
0        Cho Ray Hospital, Ho Chi Minh City, Vietnam  images
1        Cho Ray Hospital, Ho Chi Minh City, Vietnam  images
2        Cho Ray Hospital, Ho Chi Minh City, Vietnam  images
3        Cho Ray Hospital, Ho Chi Minh City, Vietnam  images
4  Changhua Christian Hospital, Changhua City, Ta…  images
5  Changhua Christian Hospital, Changhua City, Ta…  images
6                                                NaN  images
7  Wuhan Jinyintan Hospital, Wuhan, Hubei Provinc…  images
8  Wuhan Jinyintan Hospital, Wuhan, Hubei Provinc…  images
9     Mount Sinai Hospital, Toronto, Ontario, Canada  images

                                              filename  \
0  auntminnie-a-2020_01_28_23_51_6665_2020_01_28_…
1  auntminnie-b-2020_01_28_23_51_6665_2020_01_28_…
2  auntminnie-c-2020_01_28_23_51_6665_2020_01_28_…
3  auntminnie-d-2020_01_28_23_51_6665_2020_01_28_…
4                              nejmc2001573_f1a.jpeg
5                              nejmc2001573_f1b.jpeg
6                                     ARDSSevere.png
7                                 lancet-case2a.jpg
8                                 lancet-case2b.jpg
9  SARS-10.1148rg.242035193-g04mr34g0-Fig8a-day0…

                            doi  \
0           10.1056/nejmc2001272
1           10.1056/nejmc2001272
2           10.1056/nejmc2001272
3           10.1056/nejmc2001272
4           10.1056/NEJMc2001573
5           10.1056/NEJMc2001573
6                            NaN
7  10.1016/S0140-6736(20)30211-7
8  10.1016/S0140-6736(20)30211-7
9           10.1148/rg.242035193

                                                 url   license  \
0  https://www.nejm.org/doi/full/10.1056/NEJMc200…       NaN
1  https://www.nejm.org/doi/full/10.1056/NEJMc200…       NaN
2  https://www.nejm.org/doi/full/10.1056/NEJMc200…       NaN
3  https://www.nejm.org/doi/full/10.1056/NEJMc200…       NaN
```

```
4  https://www.nejm.org/doi/full/10.1056/NEJMc200…       NaN
5  https://www.nejm.org/doi/full/10.1056/NEJMc200…       NaN
6  https://en.wikipedia.org/wiki/File:ARDSSevere.png  CC BY-SA
7  https://www.thelancet.com/journals/lancet/arti…       NaN
8  https://www.thelancet.com/journals/lancet/arti…       NaN
9     https://pubs.rsna.org/doi/10.1148/rg.242035193       NaN

                                    clinical_notes other_notes Unnamed: 29
0  On January 22, 2020, a 65-year-old man with a …         NaN         NaN
1  On January 22, 2020, a 65-year-old man with a …         NaN         NaN
2  On January 22, 2020, a 65-year-old man with a …         NaN         NaN
3  On January 22, 2020, a 65-year-old man with a …         NaN         NaN
4   diffuse infiltrates in the bilateral lower lungs         NaN         NaN
5  progressive diffuse interstitial opacities and…         NaN         NaN
6  Severe ARDS. Person is intubated with an OG in…         NaN         NaN
7  Case 2: chest x-ray obtained on Jan 6 (2A). Th…         NaN         NaN
8  Case 2: chest x-ray obtained on Jan 6 (2A). Th…         NaN         NaN
9  SARS in a 74-year-old man who developed sympto…         NaN         NaN

[10 rows x 30 columns]
```

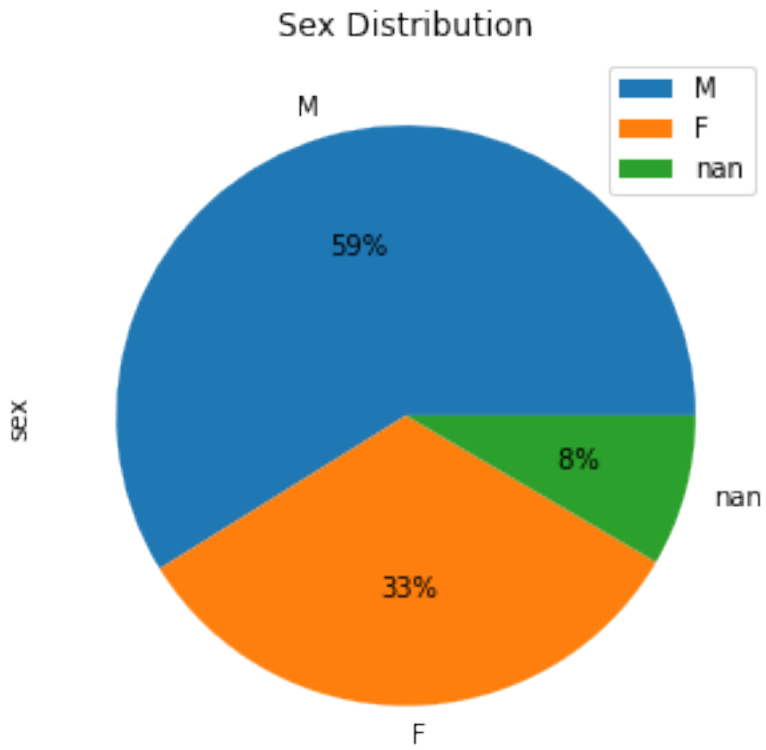
1.1 Plot some readily available values (*nan* where data is unknown)

[3]:
```
# Finding distribution (nan if unknown)
ax = metadata['finding'].value_counts(dropna=False).plot.pie(y='Finding',␣
 ↪legend = True, autopct='%2.0f%%', figsize = (10,10), title = 'Finding␣
 ↪Distribution')
```

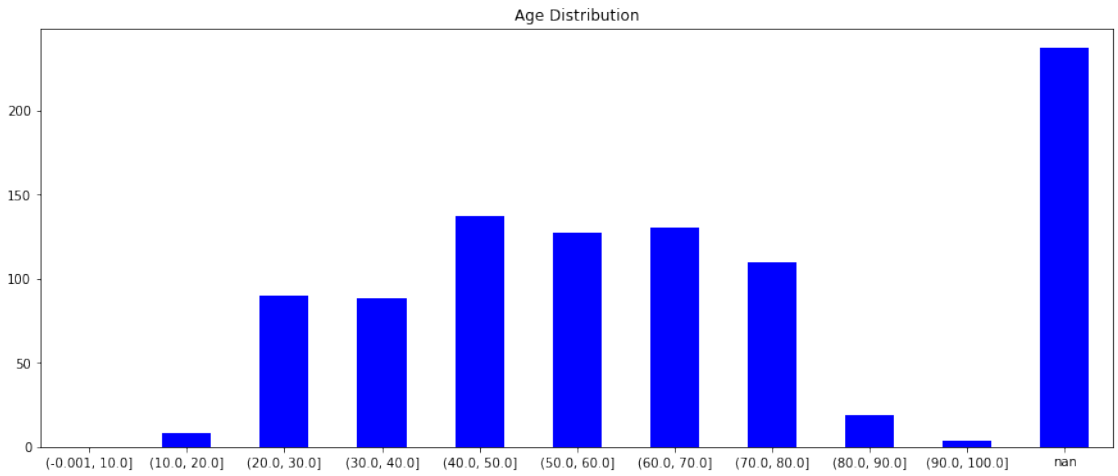


[4]:

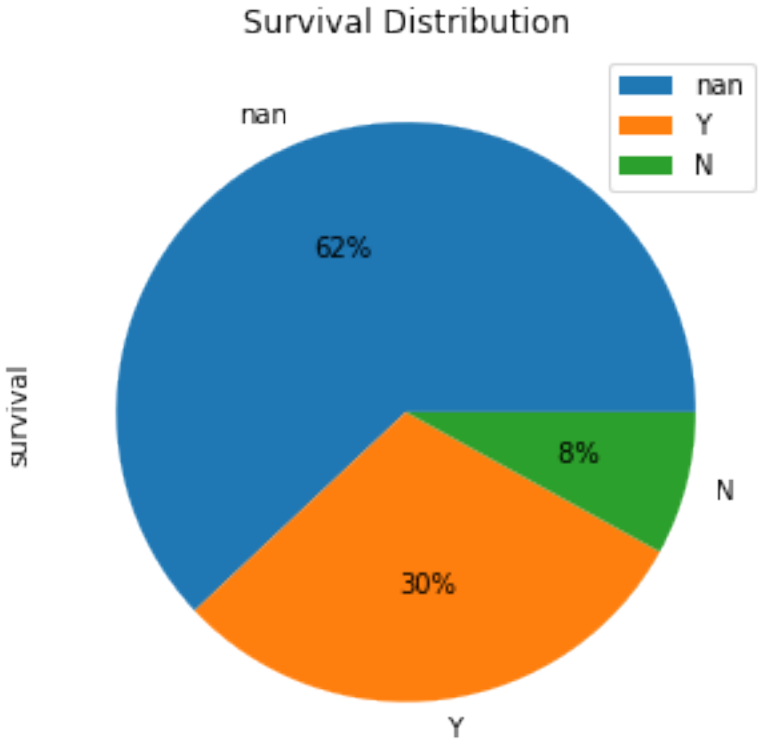
```
# Sex distribution (nan if unknown)
ax = metadata['sex'].value_counts(dropna=False).plot.pie(y='sex', legend =␣
↪True, autopct='%2.0f%%', figsize = (5,5), title = 'Sex Distribution')
```



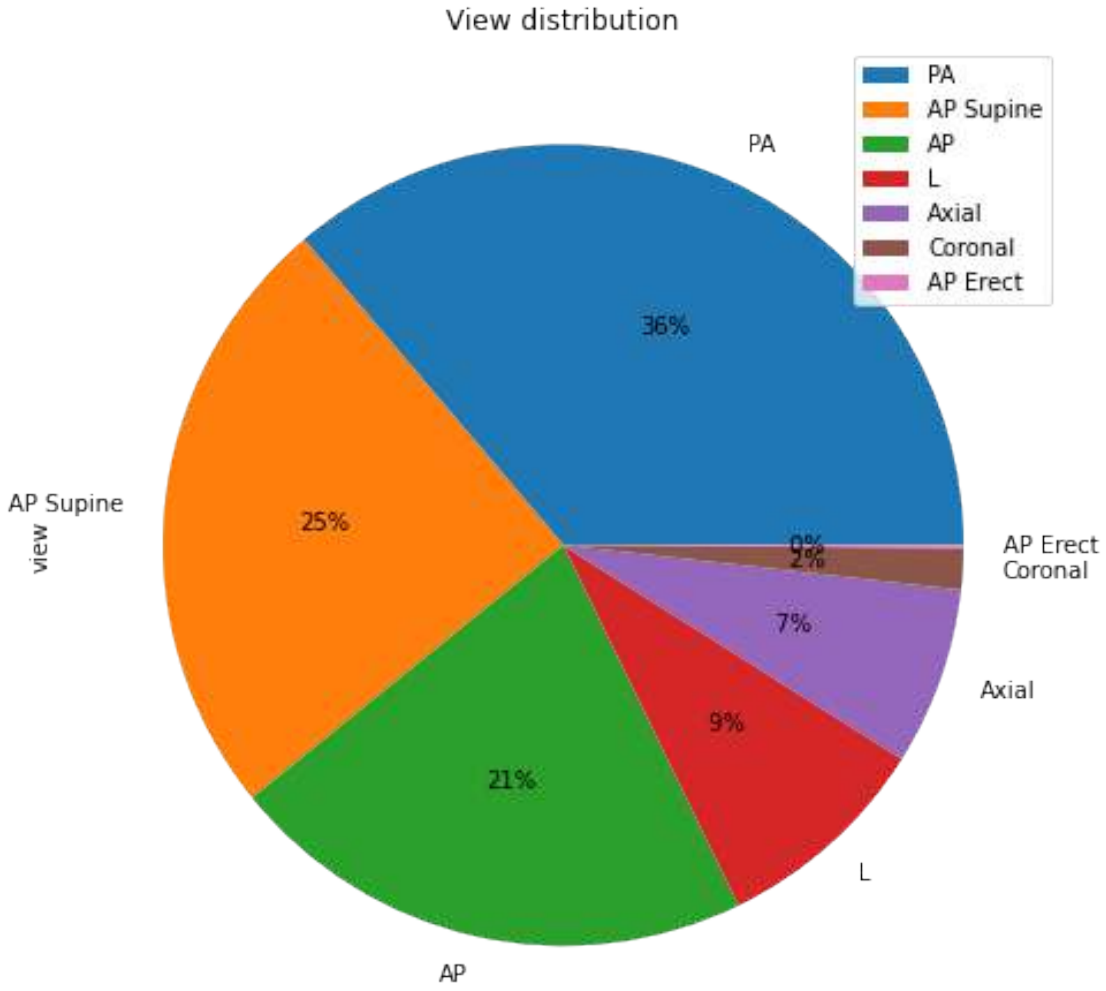
[5]:
```
# Now age (nan in unknown)
out = pd.cut(metadata['age'], bins=np.arange(0,110,10).tolist(),␣
 ↪include_lowest=True)
ax = out.value_counts(sort=False, dropna=False).plot.bar(rot=0, color="b",␣
 ↪figsize=(15,6), title= "Age Distribution")
#ax.set_xticklabels([c[1:-1].replace("," to") for c in out.cat.categories])
plt.show()
```



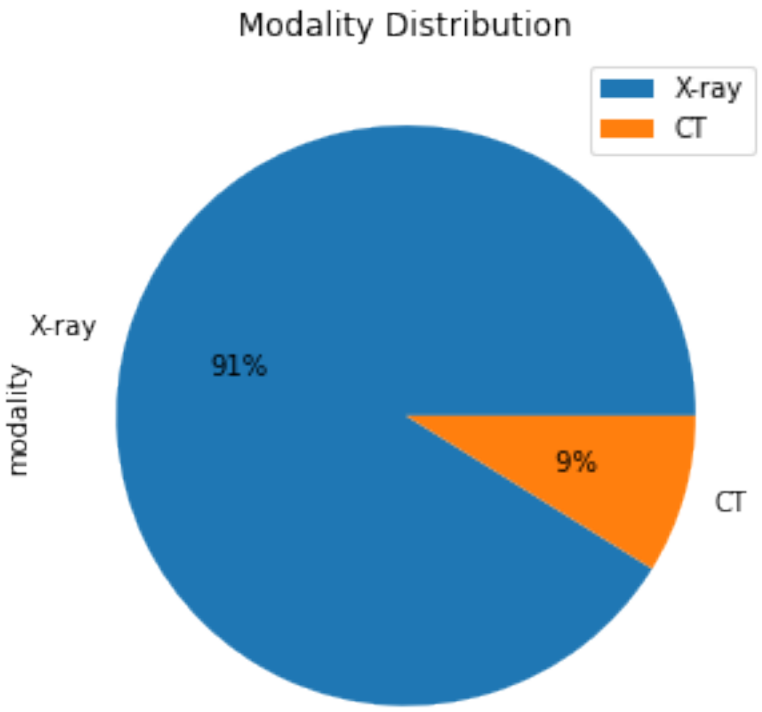
[6]:
```
# Survival
ax = metadata['survival'].value_counts(dropna=False).plot.pie(y='survival',␣
 ↪legend = True, autopct='%2.0f%%', figsize = (5,5), title = 'Survival␣
 ↪Distribution')
```



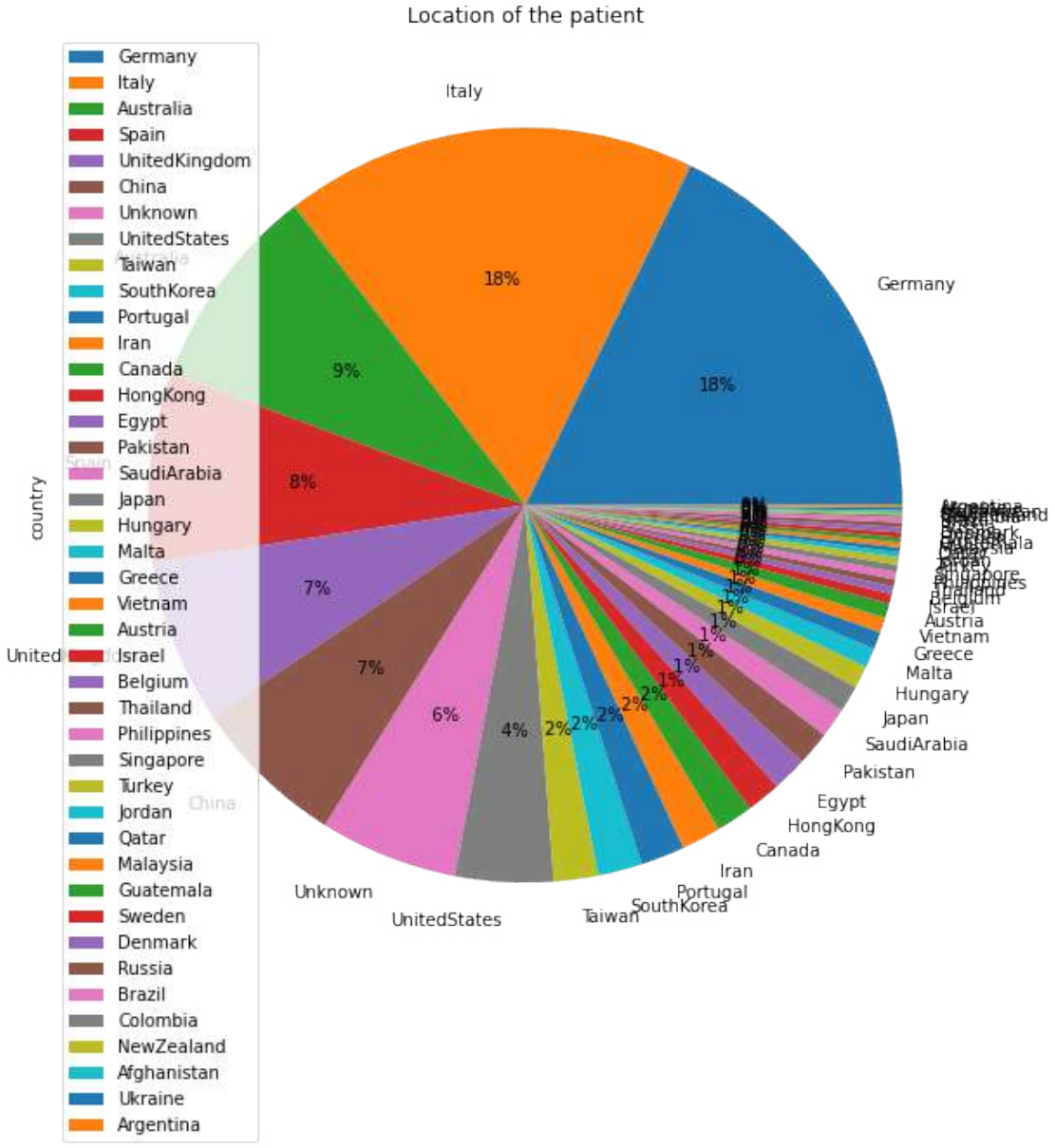
[7]:
```
# View
ax = metadata['view'].value_counts(dropna=False).plot.pie(y='view', legend =␣
 ↪True, autopct='%2.0f%%', figsize = (8,8), title = 'View distribution')
```



[8]:
```
# Modality
ax = metadata['modality'].value_counts(dropna=False).plot.pie(y='modality',␣
↪legend = True, autopct='%2.0f%%', figsize = (5,5), title = 'Modality␣
↪Distribution')
```



[9]:
```
# Location
metadata['country'] = metadata['location'].apply(lambda x: x.split(',')[-1].
 ↪replace(" ","") if x is not np.nan else "Unknown")
ax = metadata['country'].value_counts(dropna=False).plot.pie(y='country',␣
 ↪legend = True, autopct='%2.0f%%', figsize = (10,10), title = 'Location of␣
 ↪the patient')
```


Location of the patient
country
Germany
Italy
Australia
Spain
UnitedKingdom
China
Unknown
UnitedStates
Taiwan
SouthKorea
Portugal
Iran
Canada
HongKong
Egypt
Pakistan
SaudiArabia
Japan
Hungary
Malta
Greece
Vietnam
Austria
Israel
Belgium
Thailand
Philippines
Singapore
Turkey
Jordan
Qatar
Malaysia
Guatemala
Sweden
Denmark
Russia
Brazil
Colombia
NewZealand
Afghanistan
Ukraine
Argentina
18%
18%
9%
8%
7%
7%
6%
4%
2%
2%
2%
2%
1%
1%
1%
1%
1%
1%
1%
Germany
Italy
Unknown
UnitedStates
Taiwan
SouthKorea
Portugal
Iran
Canada
HongKong
Egypt
Pakistan
SaudiArabia
Japan
Hungary
Malta
Greece
Vietnam
Austria
Israel

Proposed_Neural_Network

May 11, 2021

[1]:
```
# Initialising keras framework
from keras import backend as K
from keras.preprocessing.image import ImageDataGenerator,load_img, img_to_array
from keras.models import Sequential, Model
from keras.layers import Conv2D, MaxPooling2D,GlobalAveragePooling2D
from keras.layers import Activation, Dropout, BatchNormalization, Flatten,␣
 ↪Dense, AvgPool2D,MaxPool2D
from keras.models import Sequential, Model
from keras.applications.vgg16 import VGG16, preprocess_input
from keras.optimizers import Adam, SGD, RMSprop
# Importing necessary modules
import tensorflow as tf
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

[2]:
```
DATASET_DIR = "Dataset"
```

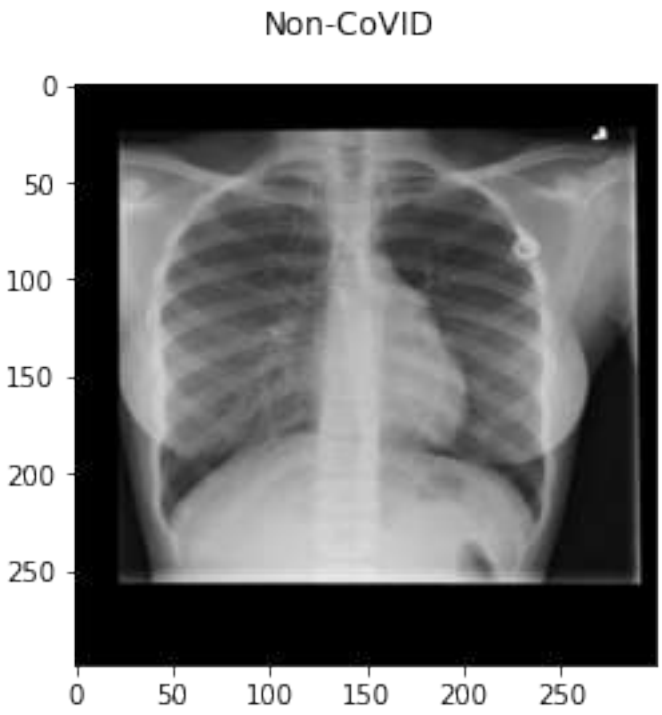
[3]:
```
os.listdir(DATASET_DIR)
```

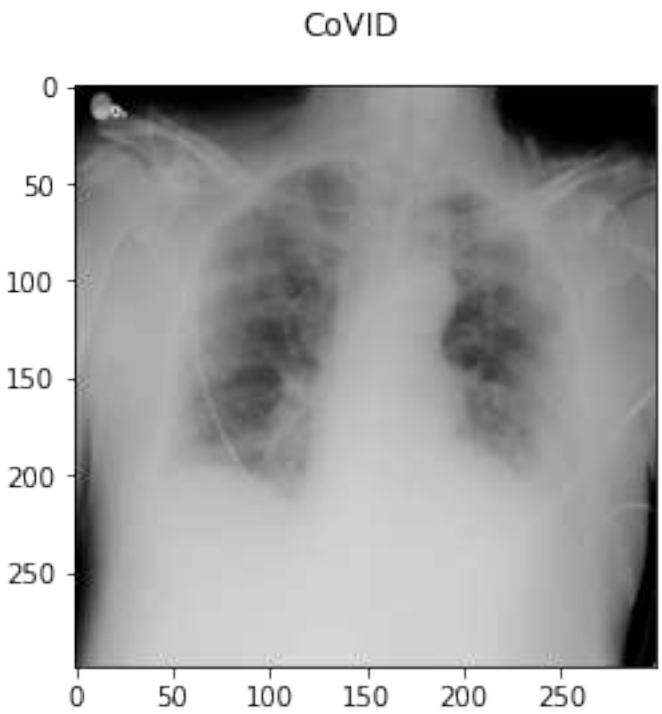
[3]:
```
['covid', 'normal']
```

[4]:
```
import glob
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
%matplotlib inline
# Appending images into list and displaying it
normal_images = []
for img_path in glob.glob(DATASET_DIR + '/normal/*'):
    normal_images.append(mpimg.imread(img_path))

fig = plt.figure()
fig.suptitle('Non-CoVID')
plt.imshow(normal_images[0], cmap='gray')

covid_images = []
```

```
for img_path in glob.glob(DATASET_DIR + '/covid/*'):
    covid_images.append(mpimg.imread(img_path))

fig = plt.figure()
fig.suptitle('CoVID')
plt.imshow(covid_images[0], cmap='gray')
```

[4]: <matplotlib.image.AxesImage at 0x207c200d340>





[5]:
```
print(len(normal_images))
print(len(covid_images))
```

```
636
198
```

[6]:
```
IMG_W = 150
IMG_H = 150
CHANNELS = 3

INPUT_SHAPE = (IMG_W, IMG_H, CHANNELS)
NB_CLASSES = 2
EPOCHS = 50
BATCH_SIZE = 6
```

[7]:
```
# First Convolution 2D Layer
model = Sequential()
model.add(Conv2D(32, (3, 3), input_shape=INPUT_SHAPE))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
# Second Convolution 2D Layer
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
```

```
# Third Convolution 2D Layer
model.add(Conv2D(128,(3,3)))
model.add(Activation("relu"))
# Fourth Convolution 2D Layer
model.add(Conv2D(256,(3,3)))
model.add(Activation("relu"))
# Flattening Layer
model.add(Flatten())
model.add(Dense(32))
model.add(Dropout(0.25))
model.add(Dense(1))
model.add(Activation("sigmoid"))
```

[8]:
```
model.compile(loss='binary_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

[9]:
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 148, 148, 32)      896
_________________________________________________________________
activation (Activation)      (None, 148, 148, 32)      0
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 74, 74, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 72, 72, 64)        18496
_________________________________________________________________
activation_1 (Activation)    (None, 72, 72, 64)        0
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 36, 36, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 34, 34, 128)       73856
_________________________________________________________________
activation_2 (Activation)    (None, 34, 34, 128)       0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 32, 32, 256)       295168
_________________________________________________________________
activation_3 (Activation)    (None, 32, 32, 256)       0
_________________________________________________________________
flatten (Flatten)            (None, 262144)            0
_________________________________________________________________
dense (Dense)                (None, 32)                8388640
_________________________________________________________________
dropout (Dropout)            (None, 32)                0
```

```
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 33
_________________________________________________________________
activation_4 (Activation)    (None, 1)                 0
=================================================================
Total params: 8,777,089
Trainable params: 8,777,089
Non-trainable params: 0
_________________________________________________________________
```

[10]:
```
train_datagen = ImageDataGenerator(rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    validation_split=0.3)

train_generator = train_datagen.flow_from_directory(
    DATASET_DIR,
    target_size=(IMG_H, IMG_W),
    batch_size=BATCH_SIZE,
    class_mode='binary',
    subset='training')

validation_generator = train_datagen.flow_from_directory(
    DATASET_DIR,
    target_size=(IMG_H, IMG_W),
    batch_size=BATCH_SIZE,
    class_mode='binary',
    shuffle= False,
    subset='validation')

history = model.fit_generator(
    train_generator,
    steps_per_epoch = train_generator.samples // BATCH_SIZE,
    validation_data = validation_generator,
    validation_steps = validation_generator.samples // BATCH_SIZE,
    epochs = EPOCHS)
```

```
Found 585 images belonging to 2 classes.
Found 249 images belonging to 2 classes.
Epoch 1/50

D:\Python\Anaconda\lib\site-
packages\tensorflow\python\keras\engine\training.py:1844: UserWarning:
`Model.fit_generator` is deprecated and will be removed in a future version.
Please use `Model.fit`, which supports generators.
  warnings.warn('`Model.fit_generator` is deprecated and '

97/97 [==============================] - 32s 322ms/step - loss: 2.6960 -
```

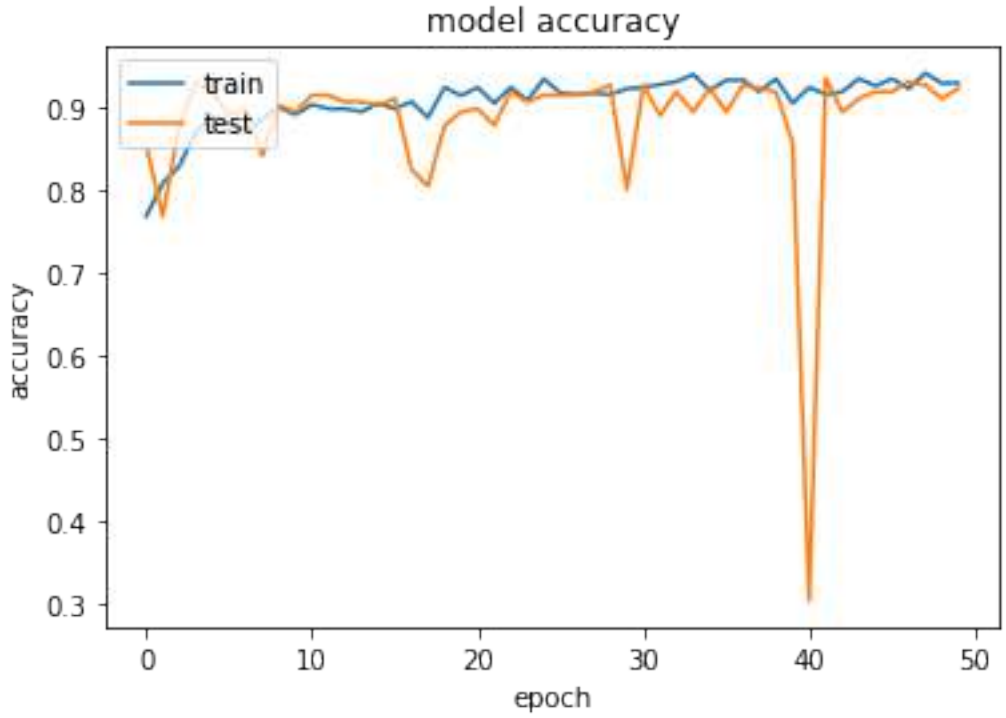
```
accuracy: 0.7224 - val_loss: 0.3297 - val_accuracy: 0.8537
Epoch 2/50
97/97 [==============================] - 30s 311ms/step - loss: 0.8401 -
accuracy: 0.7789 - val_loss: 0.5527 - val_accuracy: 0.7683
Epoch 3/50
97/97 [==============================] - 30s 310ms/step - loss: 0.3980 -
accuracy: 0.8289 - val_loss: 0.3805 - val_accuracy: 0.8780
Epoch 4/50
97/97 [==============================] - 30s 311ms/step - loss: 0.4091 -
accuracy: 0.8716 - val_loss: 0.3344 - val_accuracy: 0.9309
Epoch 5/50
97/97 [==============================] - 30s 311ms/step - loss: 0.3059 -
accuracy: 0.8900 - val_loss: 0.2562 - val_accuracy: 0.9187
Epoch 6/50
97/97 [==============================] - 30s 310ms/step - loss: 0.2708 -
accuracy: 0.8988 - val_loss: 0.3184 - val_accuracy: 0.8902
Epoch 7/50
97/97 [==============================] - 30s 309ms/step - loss: 0.2984 -
accuracy: 0.8748 - val_loss: 0.3917 - val_accuracy: 0.8984
Epoch 8/50
97/97 [==============================] - 30s 309ms/step - loss: 0.2966 -
accuracy: 0.8844 - val_loss: 0.4554 - val_accuracy: 0.8415
Epoch 9/50
97/97 [==============================] - 30s 314ms/step - loss: 0.2917 -
accuracy: 0.9086 - val_loss: 0.2969 - val_accuracy: 0.9024
Epoch 10/50
97/97 [==============================] - 30s 314ms/step - loss: 0.3112 -
accuracy: 0.9014 - val_loss: 0.3038 - val_accuracy: 0.8943
Epoch 11/50
97/97 [==============================] - 30s 311ms/step - loss: 0.3377 -
accuracy: 0.9004 - val_loss: 0.2606 - val_accuracy: 0.9146
Epoch 12/50
97/97 [==============================] - 30s 309ms/step - loss: 0.2665 -
accuracy: 0.9108 - val_loss: 0.3113 - val_accuracy: 0.9146
Epoch 13/50
97/97 [==============================] - 30s 310ms/step - loss: 0.2935 -
accuracy: 0.9043 - val_loss: 0.3364 - val_accuracy: 0.9065
Epoch 14/50
97/97 [==============================] - 30s 311ms/step - loss: 0.2376 -
accuracy: 0.8986 - val_loss: 0.2096 - val_accuracy: 0.9065
Epoch 15/50
97/97 [==============================] - 30s 314ms/step - loss: 0.2371 -
accuracy: 0.9145 - val_loss: 0.3617 - val_accuracy: 0.9024
Epoch 16/50
97/97 [==============================] - 31s 320ms/step - loss: 0.4281 -
accuracy: 0.8976 - val_loss: 0.3933 - val_accuracy: 0.9106
Epoch 17/50
97/97 [==============================] - 31s 318ms/step - loss: 0.2020 -
```

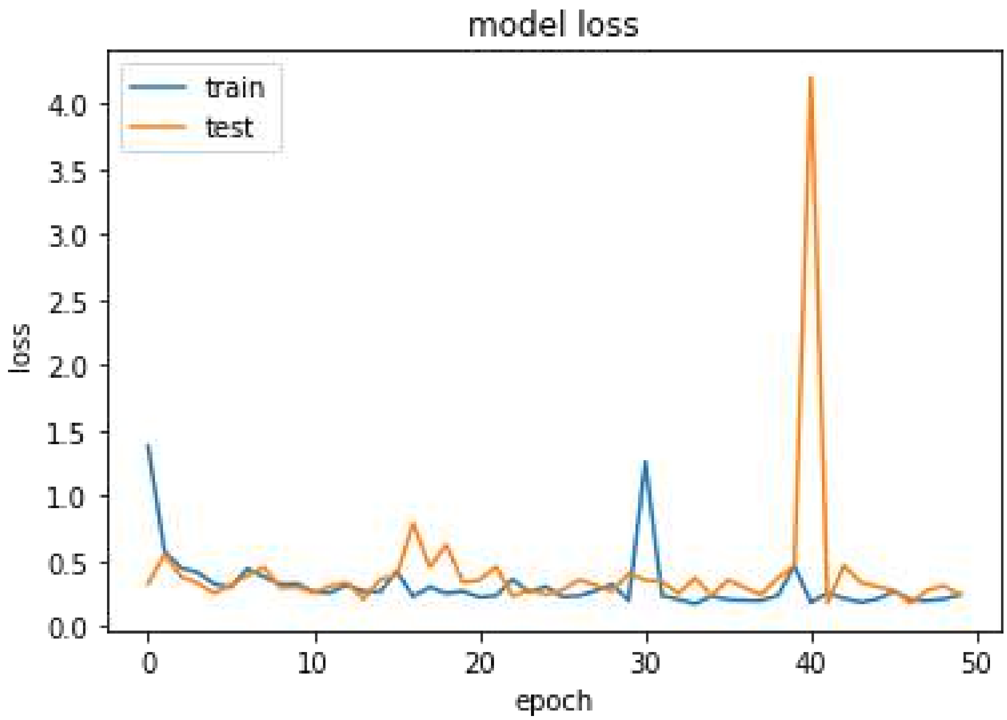
```
accuracy: 0.9268 - val_loss: 0.7896 - val_accuracy: 0.8252
Epoch 18/50
97/97 [==============================] - 30s 310ms/step - loss: 0.3722 -
accuracy: 0.8602 - val_loss: 0.4580 - val_accuracy: 0.8049
Epoch 19/50
97/97 [==============================] - 30s 309ms/step - loss: 0.2068 -
accuracy: 0.9429 - val_loss: 0.6266 - val_accuracy: 0.8780
Epoch 20/50
97/97 [==============================] - 30s 311ms/step - loss: 0.2486 -
accuracy: 0.9249 - val_loss: 0.3381 - val_accuracy: 0.8943
Epoch 21/50
97/97 [==============================] - 30s 310ms/step - loss: 0.2212 -
accuracy: 0.9197 - val_loss: 0.3567 - val_accuracy: 0.8984
Epoch 22/50
97/97 [==============================] - 31s 320ms/step - loss: 0.2263 -
accuracy: 0.9122 - val_loss: 0.4586 - val_accuracy: 0.8780
Epoch 23/50
97/97 [==============================] - 30s 310ms/step - loss: 0.2798 -
accuracy: 0.9106 - val_loss: 0.2321 - val_accuracy: 0.9187
Epoch 24/50
97/97 [==============================] - 30s 310ms/step - loss: 0.2238 -
accuracy: 0.9252 - val_loss: 0.2816 - val_accuracy: 0.9065
Epoch 25/50
97/97 [==============================] - 30s 310ms/step - loss: 0.2757 -
accuracy: 0.9382 - val_loss: 0.2467 - val_accuracy: 0.9146
Epoch 26/50
97/97 [==============================] - 30s 310ms/step - loss: 0.2037 -
accuracy: 0.9271 - val_loss: 0.2800 - val_accuracy: 0.9146
Epoch 27/50
97/97 [==============================] - 30s 311ms/step - loss: 0.2291 -
accuracy: 0.9186 - val_loss: 0.3585 - val_accuracy: 0.9146
Epoch 28/50
97/97 [==============================] - 30s 310ms/step - loss: 0.2650 -
accuracy: 0.9177 - val_loss: 0.3185 - val_accuracy: 0.9187
Epoch 29/50
97/97 [==============================] - 30s 311ms/step - loss: 0.2371 -
accuracy: 0.9231 - val_loss: 0.2714 - val_accuracy: 0.9268
Epoch 30/50
97/97 [==============================] - 31s 315ms/step - loss: 0.1775 -
accuracy: 0.9389 - val_loss: 0.4056 - val_accuracy: 0.8008
Epoch 31/50
97/97 [==============================] - 32s 327ms/step - loss: 2.4819 -
accuracy: 0.9250 - val_loss: 0.3564 - val_accuracy: 0.9268
Epoch 32/50
97/97 [==============================] - 32s 330ms/step - loss: 0.2612 -
accuracy: 0.9286 - val_loss: 0.3460 - val_accuracy: 0.8902
Epoch 33/50
97/97 [==============================] - 31s 316ms/step - loss: 0.1734 -
```

```
accuracy: 0.9481 - val_loss: 0.2548 - val_accuracy: 0.9187
Epoch 34/50
97/97 [==============================] - 30s 307ms/step - loss: 0.1346 -
accuracy: 0.9610 - val_loss: 0.3736 - val_accuracy: 0.8943
Epoch 35/50
97/97 [==============================] - 31s 317ms/step - loss: 0.2546 -
accuracy: 0.8989 - val_loss: 0.2383 - val_accuracy: 0.9228
Epoch 36/50
97/97 [==============================] - 30s 313ms/step - loss: 0.2233 -
accuracy: 0.9362 - val_loss: 0.3579 - val_accuracy: 0.8943
Epoch 37/50
97/97 [==============================] - 32s 329ms/step - loss: 0.2232 -
accuracy: 0.9261 - val_loss: 0.2984 - val_accuracy: 0.9268
Epoch 38/50
97/97 [==============================] - 31s 323ms/step - loss: 0.1969 -
accuracy: 0.9222 - val_loss: 0.2457 - val_accuracy: 0.9228
Epoch 39/50
97/97 [==============================] - 30s 312ms/step - loss: 0.2186 -
accuracy: 0.9299 - val_loss: 0.3736 - val_accuracy: 0.9187
Epoch 40/50
97/97 [==============================] - 30s 312ms/step - loss: 0.2069 -
accuracy: 0.9434 - val_loss: 0.4666 - val_accuracy: 0.8577
Epoch 41/50
97/97 [==============================] - 30s 313ms/step - loss: 0.1776 -
accuracy: 0.9263 - val_loss: 4.2025 - val_accuracy: 0.3049
Epoch 42/50
97/97 [==============================] - 31s 323ms/step - loss: 0.4341 -
accuracy: 0.8812 - val_loss: 0.1838 - val_accuracy: 0.9350
Epoch 43/50
97/97 [==============================] - 30s 312ms/step - loss: 0.2395 -
accuracy: 0.9154 - val_loss: 0.4697 - val_accuracy: 0.8943
Epoch 44/50
97/97 [==============================] - 31s 317ms/step - loss: 0.2033 -
accuracy: 0.9349 - val_loss: 0.3388 - val_accuracy: 0.9106
Epoch 45/50
97/97 [==============================] - 30s 312ms/step - loss: 0.1903 -
accuracy: 0.9248 - val_loss: 0.3066 - val_accuracy: 0.9187
Epoch 46/50
97/97 [==============================] - 31s 317ms/step - loss: 0.2201 -
accuracy: 0.9325 - val_loss: 0.2779 - val_accuracy: 0.9187
Epoch 47/50
97/97 [==============================] - 30s 314ms/step - loss: 0.1776 -
accuracy: 0.9362 - val_loss: 0.1802 - val_accuracy: 0.9309
Epoch 48/50
97/97 [==============================] - 32s 333ms/step - loss: 0.1914 -
accuracy: 0.9538 - val_loss: 0.2765 - val_accuracy: 0.9268
Epoch 49/50
97/97 [==============================] - 31s 315ms/step - loss: 0.2222 -
```

```
accuracy: 0.9406 - val_loss: 0.3146 - val_accuracy: 0.9106
Epoch 50/50
97/97 [==============================] - 30s 312ms/step - loss: 0.2443 -
accuracy: 0.9182 - val_loss: 0.2439 - val_accuracy: 0.9228
```

[11]:
```
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```





[12]:
```
print("training_accuracy", history.history['accuracy'][-1])
print("validation_accuracy", history.history['val_accuracy'][-1])
```

```
training_accuracy 0.9291882514953613
validation_accuracy 0.922764241695404
```

[13]:
```
label = validation_generator.classes
```

[14]:
```
pred= model.predict(validation_generator)
predicted_class_indices=np.argmax(pred,axis=1)
labels = (validation_generator.class_indices)
labels2 = dict((v,k) for k,v in labels.items())
predictions = [labels2[k] for k in predicted_class_indices]
print(predicted_class_indices)
print (labels)
print (predictions)
```

```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
```

```
{'covid': 0, 'normal': 1}
['covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid',
'covid', 'covid', 'covid', 'covid', 'covid', 'covid', 'covid']
```

[17]:
```python
from sklearn.metrics import confusion_matrix

cf = confusion_matrix(predicted_class_indices,label)
cf
```

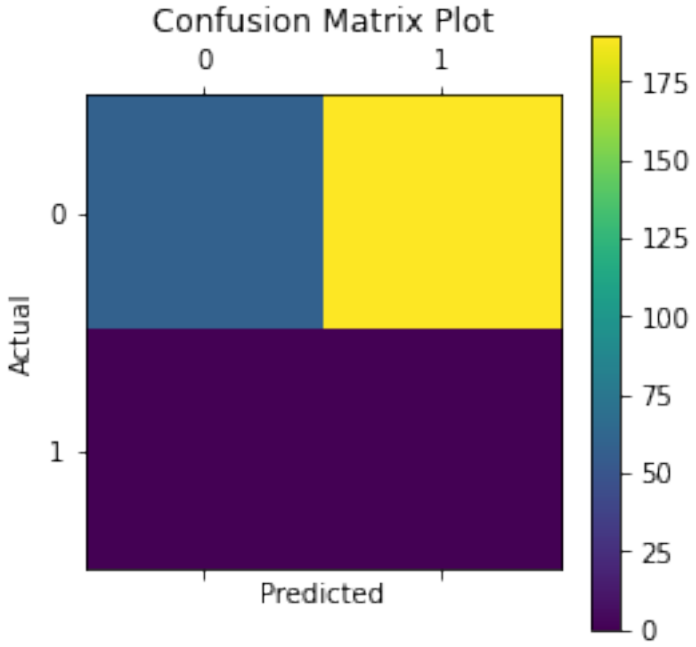
[17]:
```
array([[ 59, 190],
       [  0,   0]], dtype=int64)
```

[18]:
```python
exp_series = pd.Series(label)
pred_series = pd.Series(predicted_class_indices)
pd.crosstab(exp_series, pred_series, rownames=['Actual'],␣
 ↪colnames=['Predicted'],margins=True)
```

[18]:

Predicted	0	All
Actual		
0	59	59
1	190	190
All	249	249

[19]:
```
plt.matshow(cf)
plt.title('Confusion Matrix Plot')
plt.colorbar()
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show();
```



[ ]:

Segregating_CoVID_Patients_Data

May 11, 2021

[1]:
```
'''
This code finds all images of patients of a specified VIRUS and X-Ray view and␣
 ↪stores selected image to an OUTPUT directory
+ It uses metadata.csv for searching and retrieving images name
+ Using ./images folder it selects the retrieved images and copies them in␣
 ↪output folder
Code can be modified for any combination of selection of images
'''

import pandas as pd
import shutil
import os

# Selecting all combination of 'COVID-19' patients with 'PA' X-Ray view
virus = "Pneumonia/Viral/COVID-19" # Virus to look for
x_ray_view = "PA" # View of X-Ray

metadata = "../metadata.csv" # Meta info
imageDir = "../images" # Directory of images
outputDir = '../output/' # Output directory to store selected images

if not os.path.isfile(outputDir):  # check if directory already exist
    os.mkdir(outputDir)  # create a directory

metadata_csv = pd.read_csv(metadata)

# loop over the rows of the COVID-19 data frame
for (i, row) in metadata_csv.iterrows():
        if row["finding"] != virus or row["view"] != x_ray_view:
                continue

        filename = row["filename"].split(os.path.sep)[-1]
        filePath = os.path.sep.join([imageDir, filename])
        shutil.copy2(filePath, outputDir)
```

Verify_And_Modify_Dataset

May 11, 2021

[1]:
```
import torch
import torchvision
import torchxrayvision as xrv
from tqdm import tqdm
import pandas as pd
import numpy as np
import sys
```

[2]:
```
'''d_covid19_base = xrv.datasets.COVID19_Dataset(views=["PA", "AP", "AP␣
↪Supine"],
                                                 imgpath="../images",
                                                 csvpath="../metadata.csv")'''
```

[3]:
```
d_covid19 = xrv.datasets.COVID19_Dataset(views=["PA", "AP", "AP Supine"],␣
↪imgpath="../images", csvpath="../metadata.csv")
print(d_covid19)
```

```
{'Aspergillosis': {0.0: 695, 1.0: 2},
 'Aspiration': {0.0: 696, 1.0: 1},
 'Bacterial': {0.0: 644, 1.0: 53},
 'COVID-19': {0.0: 219, 1.0: 478},
 'Chlamydophila': {0.0: 696, 1.0: 1},
 'E.Coli': {0.0: 693, 1.0: 4},
 'Fungal': {0.0: 671, 1.0: 26},
 'H1N1': {0.0: 696, 1.0: 1},
 'Herpes ': {0.0: 694, 1.0: 3},
 'Influenza': {0.0: 693, 1.0: 4},
 'Klebsiella': {0.0: 688, 1.0: 9},
 'Legionella': {0.0: 688, 1.0: 9},
 'Lipoid': {0.0: 689, 1.0: 8},
 'MERS-CoV': {0.0: 687, 1.0: 10},
 'MRSA': {0.0: 696, 1.0: 1},
 'Mycoplasma': {0.0: 692, 1.0: 5},
 'No Finding': {0.0: 679, 1.0: 18},
 'Nocardia': {0.0: 693, 1.0: 4},
 'Pneumocystis': {0.0: 673, 1.0: 24},
 'Pneumonia': {0.0: 29, 1.0: 668},
 'SARS': {0.0: 681, 1.0: 16},
```

```
 'Staphylococcus': {0.0: 696, 1.0: 1},
 'Streptococcus': {0.0: 679, 1.0: 18},
 'Tuberculosis': {0.0: 686, 1.0: 11},
 'Varicella': {0.0: 692, 1.0: 5},
 'Viral': {0.0: 181, 1.0: 516}}
COVID19_Dataset num_samples=697 views=['PA', 'AP', 'AP Supine'] data_aug=None
```

[4]:
```
missing = []
for i in range(len(d_covid19)):
    idx = len(d_covid19)-i-1
    try:
        # start from the most recent
        a = d_covid19[idx]
    except KeyboardInterrupt:
        break;
    except:
        missing.append(d_covid19.csv.iloc[idx].filename)
        print("Error with {}".format(i) + d_covid19.csv.iloc[idx].filename)
        print(sys.exc_info()[1])
```

[5]:
```
missing
```

[5]:
```
[]
```

[6]:
```
for i in ['76093afc.jpg',
 'fd389adb.jpg',
 '48b98ca2.jpg',
 '5cc1a119.jpg',
 '81089cb4.jpg',
 '3c3a2a35.jpg',
 'beef3909.jpg',
 '8251e162.jpg',
 '47e3c811.jpg',
 '9ff4a125.jpg',
 '54da03cf.jpg',
 'f8a26220.jpg',
 'bf333f43.jpg',
 '851007d4.jpg',
 'a2e1a826.jpg',
 '8a81d526.jpg',
 '8abe5256.jpg',
 'f6cfef3f.jpg',
 '7998d604.jpg',
 '626eb0ef.jpg',
 'a3111116.jpg',
 '78e9a055.jpg',
 '839e1363.jpg',
 '7a17a765.jpg',
```

```
'7929f04a.jpg',
'd59d92b8.jpg',
'91cda775.jpg',
'7bb9327c.jpg',
'981e154a.jpg',
'40954b81.jpg',
'5c87a1d7.jpg',
'bf39b989.jpg',
'924f9c14.jpg',
'96da829b.jpg',
'b4e9a53a.jpg',
'b606e1d0.jpg',
'd6b8d378.jpg',
'ac00512e.jpg',
'17ad0a56.jpg',
'8e438fce.jpg',
'3d388a98.jpg',
'a7abee59.jpg',
'cf35d0c4.jpg',
'93fd0adb.jpg',
'1bc3008e.jpg',
'1930e42f.jpg',
'3161e216.jpg',
'0578e08b.jpg',
'b10c49ca.jpg',
'7a2d2695.jpg',
'fce2b5d4.jpg',
'59cb1744.jpg',
'88267e40.jpg',
'a132d8b6.jpg',
'5f7a99b2.jpg',
'563118e4.jpg',
'bb4c4038.jpg',
'bace1e45.jpg',
'add529f3.jpg',
'6f7008af.jpg',
'b39206a9.jpg',
'9a9b2393.jpg',
'd22964a4.jpg',
'6c5b3802.jpg',
'c08a4f41.jpg',
'ffe8b4cb.jpg',
'f567c33c.jpg',
'00d96e05.jpg',
'd15bf071.jpg',
'7a030330.jpg',
'3c8a0876.jpg',
```

```
 'bd85e252.jpg',
 'b6e58409.jpg',
 '9f3f2d91.jpg',
 '6b5af975.jpg',
 'c9280a30.jpg']:
    print(i.replace(".jpg",""))
```

```
76093afc
fd389adb
48b98ca2
5cc1a119
81089cb4
3c3a2a35
beef3909
8251e162
47e3c811
9ff4a125
54da03cf
f8a26220
bf333f43
851007d4
a2e1a826
8a81d526
8abe5256
f6cfef3f
7998d604
626eb0ef
a3111116
78e9a055
839e1363
7a17a765
7929f04a
d59d92b8
91cda775
7bb9327c
981e154a
40954b81
5c87a1d7
bf39b989
924f9c14
96da829b
b4e9a53a
b606e1d0
d6b8d378
ac00512e
17ad0a56
8e438fce
3d388a98
```

```
a7abee59
cf35d0c4
93fd0adb
1bc3008e
1930e42f
3161e216
0578e08b
b10c49ca
7a2d2695
fce2b5d4
59cb1744
88267e40
a132d8b6
5f7a99b2
563118e4
bb4c4038
bace1e45
add529f3
6f7008af
b39206a9
9a9b2393
d22964a4
6c5b3802
c08a4f41
ffe8b4cb
f567c33c
00d96e05
d15bf071
7a030330
3c8a0876
bd85e252
b6e58409
9f3f2d91
6b5af975
c9280a30
```

[7]:
```
csv = pd.read_csv("../metadata.csv", dtype="str")
```

[8]:
```
csv = csv[~csv.filename.isin(missing)]
```

[9]:
```
csv
```

[9]:

	patientid	offset	sex	age	finding	RT_PCR_positive	\
0	2	0	M	65	Pneumonia/Viral/COVID-19	Y	
1	2	3	M	65	Pneumonia/Viral/COVID-19	Y	
2	2	5	M	65	Pneumonia/Viral/COVID-19	Y	
3	2	6	M	65	Pneumonia/Viral/COVID-19	Y	
4	4	0	F	52	Pneumonia/Viral/COVID-19	Y	

```
..        …    …  ..  ..                      …             …
945       479      0   F  40                Pneumonia            NaN
946       479     70   F  40                Pneumonia            NaN
947       480    NaN   M  26                Pneumonia            NaN
948       481    NaN   M  50                Pneumonia            NaN
949       481    NaN   M  50                Pneumonia            NaN

    survival intubated intubation_present went_icu  …             date  \
0          Y         N                   N        N  …  January 22, 2020
1          Y         N                   N        N  …  January 25, 2020
2          Y         N                   N        N  …  January 27, 2020
3          Y         N                   N        N  …  January 28, 2020
4        NaN         N                   N        N  …  January 25, 2020
..       …       …                 …       …  …               …
945      NaN       NaN                 NaN      NaN  …              NaN
946      NaN       NaN                 NaN      NaN  …              NaN
947      NaN       NaN                 NaN      NaN  …              NaN
948      NaN       NaN                 NaN      NaN  …              NaN
949      NaN       NaN                 NaN      NaN  …              NaN

                                          location  folder  \
0          Cho Ray Hospital, Ho Chi Minh City, Vietnam  images
1          Cho Ray Hospital, Ho Chi Minh City, Vietnam  images
2          Cho Ray Hospital, Ho Chi Minh City, Vietnam  images
3          Cho Ray Hospital, Ho Chi Minh City, Vietnam  images
4    Changhua Christian Hospital, Changhua City, Ta…  images
..                                                 …     …
945                                United Kingdom  images
946                                United Kingdom  images
947                                           NaN  images
948                                           NaN  images
949                                           NaN  images

                                          filename                  doi  \
0    auntminnie-a-2020_01_28_23_51_6665_2020_01_28_…  10.1056/nejmc2001272
1    auntminnie-b-2020_01_28_23_51_6665_2020_01_28_…  10.1056/nejmc2001272
2    auntminnie-c-2020_01_28_23_51_6665_2020_01_28_…  10.1056/nejmc2001272
3    auntminnie-d-2020_01_28_23_51_6665_2020_01_28_…  10.1056/nejmc2001272
4                              nejmc2001573_f1a.jpeg  10.1056/NEJMc2001573
..                                               …                   …
945       072ecaf8c60a81980abb57150a8016_jumbo-9.jpeg                 NaN
946       ff33c406392b968d483174c97eb857_jumbo-9.jpeg                 NaN
947                                   000001-266.jpg                  NaN
948                                   000001-272.jpg                  NaN
949                                   000002-268.jpg                  NaN

                                              url          license  \
```

```
0    https://www.nejm.org/doi/full/10.1056/NEJMc200…              NaN
1    https://www.nejm.org/doi/full/10.1056/NEJMc200…              NaN
2    https://www.nejm.org/doi/full/10.1056/NEJMc200…              NaN
3    https://www.nejm.org/doi/full/10.1056/NEJMc200…              NaN
4    https://www.nejm.org/doi/full/10.1056/NEJMc200…              NaN
..                                                 …              …
945  https://radiopaedia.org/cases/multifocal-round…      CC BY-NC-SA
946  https://radiopaedia.org/cases/multifocal-round…      CC BY-NC-SA
947                  https://www.eurorad.org/case/947  CC BY-NC-SA 4.0
948                  https://www.eurorad.org/case/934  CC BY-NC-SA 4.0
949                  https://www.eurorad.org/case/934  CC BY-NC-SA 4.0

                                       clinical_notes  \
0    On January 22, 2020, a 65-year-old man with a …
1    On January 22, 2020, a 65-year-old man with a …
2    On January 22, 2020, a 65-year-old man with a …
3    On January 22, 2020, a 65-year-old man with a …
4     diffuse infiltrates in the bilateral lower lungs
..                                                 …
945  Asthmatic. Shortness of breath and wheeze. Rou…
946  Asthmatic. Shortness of breath and wheeze. The…
947  fire-eater accidentally ingested a paraffin mi…
948  The patient, a heavy smoker, was referred to t…
949  The patient, a heavy smoker, was referred to t…

                                         other_notes Unnamed: 29
0                                                NaN         NaN
1                                                NaN         NaN
2                                                NaN         NaN
3                                                NaN         NaN
4                                                NaN         NaN
..                                                 …           …
945  Case courtesy of Dr Ian Bickle, Radiopaedia.or…         NaN
946  Case courtesy of Dr Ian Bickle, Radiopaedia.or…         NaN
947                                              NaN         NaN
948                                              NaN         NaN
949                                              NaN         NaN

[950 rows x 30 columns]
```

[10]:
```
csv.to_csv("../metadata.csv", index=False)
```