The application of Deep Learning Models for the diagnosis Lung Disease using chest X-ray images.

Jawad Asif

***Abstract*—This article highlights the significance of deep learning models to accurately diagnosing the lung diseases, specially when the entire world is dealing with pandemic situation. Any disease diagnosed in early stage enable medical intervention and improve patients survival rate. Radiology examination is the key screening method for lungs diseases, and helpful distinguishing between COVID and pneumonia patients. The imaging technique, chest X-Ray, provides insight of lung conditions, which together with deep learning techniques, can accurately and timely identify the diseases which reduces the burden on the health system. In this regard, we proposed and compare different deep learning approaches, to detect the COVID-19 and pneumonia cases occurred from chest X-Ray images, and compared their performances. In this vein, author proposed four different Convolutional Neural Networks and analysed on X-ray images. The experiment results show that the models can provide test accuracy up to 92.57% with augmenting data. However DenseNet-121 outperformed the others model and got 94% of accuracy, however ResNet50 did not showed satisfactory results in comparison. It is worth noting that, different models have their own strength and weakness influenced by factors such as the dataset and desired outcomes.**

***Index Terms*—COVID-19, Pneumonia, Deep Learning Convolutional Neural Network, Densenet121, ResNet, Chest X-Ray**

I. Introduction

There has been a great advancement in last few years in analysing the human data to predict different disease using advance technology. The Cronavirus Disease 2019 (COVID-19) pandemic has devastating impact on human health. Effectively screening the infected people are very crucial to reduce the risk of spreading the virus and also to provide treatment to the patient.

Early studies have shown that individual infected with COVID-19 has very diverse radiographic images from pneumonia and normal cases and X-ray images are very helpful in diagnosing COVID-19 [1]. In modern health system, the use chest X-ray imaging system and portable devices are increasing a this makes faster and accessible radiography examinations [3].

However, to detect lung disease from radiographic images, with naked eye, a challenging task for experts and time consuming. Therefore, computer aided solution is required to assist radiologist in detecting precisely the COVID-19 virus from X-ray images.Machine learning techniques, such as supervised and unsupervised models can be significantly applied on dataset to analyse the extracted feature and make prediction on it. The use of these techniques provide a deeper understanding of X-ray images and lung diseases and which leads to identify the infected person accurately and timely.

In the field of medical imaging for lung disease diagnosis, deep learning techniques have been explored for their potential in improving diagnosis accuracy. The objective to conduct this research is to study and develop deep learning models to accurately classify COVID-19 and pneumonia images, which can be deploy as a software tool to assist health system. Therefore, an automated tool will allows experts to correctly separate the COVID-19 patients from less severe patients and providing life saving best medical treatment, is also another benefit of this research. [4]

Researchers have reported different deep learning techniques for detection of COVID-19 from chest X-ray images. One approach is the use of deep onvolutional neural networks (CNN), which show most prominent results. Some other techniques are also proposed like Gray Level Co-Occurrence Matrix (GLCM) and Histogram of Oriented Gradients (HOG) with machine learning algorithms like Support Vector Machines (SVM) and Artificial Neural Networks (ANN) [6], for the classification of the images.

In addition, the study aims to determine the most optimal deep learning model to precisely classifying low arousal valence level. It also focus on impact of different data classification techniques which extracts the important features from images for better accuracy, also reducing the dimensionality of the data which in consequence reduce the complexity of the model as well as boost the training process and prevent the overfitting.

The content of the paper is organized in the following way. In section 2, author presents some related works and their procedures and results. In section 3, author describes the data preprocessing and model inputs. Section 4 provides details about signal and features. Section 5, describe learning framework. In section 6 author describes its own findings and results. The conclusion and future works is discussed in section 7.

II. Related Work

In response to the outrageous pandemic effects on entire world, for faster identification of the virus, various artificial

[†]Department of Information Engineering, University of Padova, email:rossi@dei.unipd.it

intelligence (AI) approaches have been proposed. Particularly more focusing on radiography images with promising results and accuracy to detect the COVID-19 patients. In addition, there has been a significant progress in open-source AI driven solution for detection of the cases to promote accessibility. The project has been initiated by Cohen et al. [7], for the collection of an open source COVID-19 Image Data Collection through radiography with CXR and CT annotated images.

I. Densenet-121 The author in article highlights the use of Densenet-121, a deep learning convolutional neural network, and focus on the detection of lung disease by analysing the radiography images. The study is conducted by Sriporn et al. [10] The capability of Densnet-121 to handle the high dimensional data and to utilize effective the dense layer by passing information into them, force the author to select this model. The author trained the network to classify different categories of the classes and analyse its performance, by inputting the CT images.

The author reported the results into his article, the Densenet121 was able to achieve a high accuracy of 92.3%, which shows its effectively classifying the CT images. Moreover, the author has also compared the other models with Desnsenet121 including Inception-V3 and ResNet-50 but Densenet121 outperformed these models interms of both accuracy and computational efficiency.

II. SVM with HOG To extract the features from chest X-ray images Fernandez Grandon et al. [6] conduct a study aiming to use two feature extraction techniques, the gray-level co-occurrence matrix (GLCM) and the histogram of oriented gradients (HOG). It is feature extraction computer vision technique which computes the gradient orientation histogram of the images which provide information about the shape and direction of the different features within images.

After extraction of the feature from images,they used machine learning algorithms, support vector machine (SVM) and artificial neural networks (ANN), and given these extracted features as an input to the models to classify the images. SVM is a linear classifier that set a boundaries between different classes of dataset by setting a margin between them while ANN is deep learning model which uses several layers of neurons to make prediction based on extracted features given as input.

III. COVID-Net After the release of the COVIDx dataset and suggested COVID-Net [8], vairous investigation has been conducted for the identification of COVID-19 using X-ray images and many used them this dataset and model into their studies. This dataset consist of a total of 13,975 CXR images collected from 13,870 patient cases. Proposed network architecture COVID-Net in figure[], they achieved an accuracy of 93.3% on the COVIDx test dataset. They also figured the sensitivity of 95% for normal images , for non-COVID 94% and for COVID 91%.

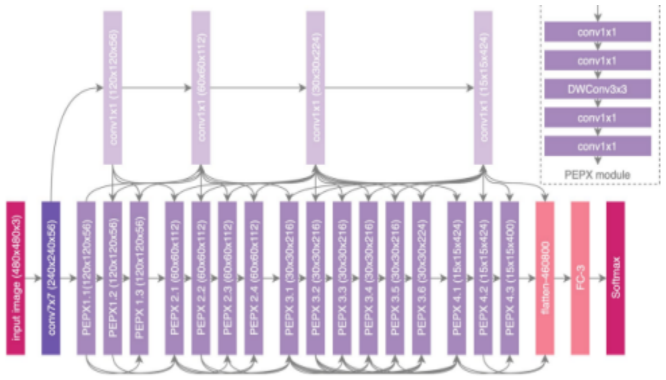

Fig. 1: COVID-Net

In another article [4], using chest X-ray images, a deep learning-based model was proposed to predict pneumonia in COVID-19 patients. A model has been built using various deep learning features, including dense, AveragePooling2D, flatten, and dropout and VGG16 model based on CNN, used to classify chest X-ray images of COVID-19 patients and pneumonia patients and predict COVID-19 patients with pneumonia. The model predicted pneumonia with an average accuracy of 91.69%, sensitivity of 95.92%, and specificity of 100%. The model also efficiently reduced training loss and increased accuracy.

Due to significant advancement in past few years in the field of AI and stateoftheart achievements in computer vision field, researcher has focused more on deep convolutional neural networks and came up with promising results dealing with images.

III. Processing Pipeline

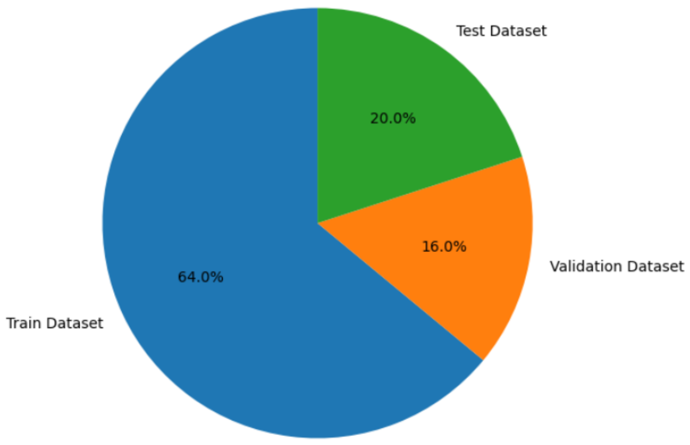

Fig. 2: DenseNet-121

In this processing pipeline section, I will briefly explain the data processing approaches that has been applied on the dataset before feeding to model for classification. The dataset consist of chest X-ray images and they are divided into three type of class into three different folders, namely

Covid, pneumonia and normal. Each folder contain 1525 images and in total they are 4575.Since the images size is different from each other so I resize the images into 256 X 256 size and converted into gray-scale and then saved into pickle file for faster access. The best practice before inputting data into model is to apply rescaling and normalization. Rescaling allows that the values of pixels are within a specific range,[0, 1], and this is achieved by dividing the pixel values by the maximum value 255 for grayscale images.

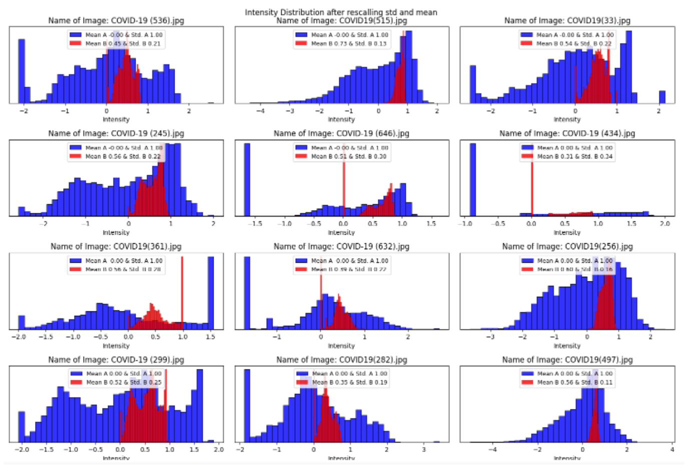

Fig. 3: Pixel Distribution

Similarly, normalizing data involves subtracting the mean value from each pixel and dividing it by the standard deviation which ensure mean of zero and standard deviation of one to reduce the effect of varying scales or brightness levels across different images for the better convergence of model and better learn from the input features. Then the data is splitted into 80% for training and 20% for testing. For validation I have splitted again the training set into 20% for validation set.

In second stage of preprocessing, I augmenting the data using keras library *ImageDataGenerator*. It involve different kind of transformation and training data to make the data more diverse and add some variability into training samples. This helps to stop overfitting and improve the generalization ability of model by including variation such as rotation, random zoom, random flip.

I have utilized different deep learning models including four custom convolutional neural network, a Densenet-121 and ResNet50. I have trained the models on given dataset have not used pretrained weights. Then we test the model accuracy on test set and finally I plotted the confusion matrix, to better understanding where the model is weakly performing.

IV. Signals and Features

The primary focus of this research is how long classification performance could be enhanced by using X-ray image dataset. This dataset is composed of 4575 samples collected from various resources. It consist of three different folder named as covid, pneumonia and normal, each folder contains 1525 images in it. These samples are posteroanterior (PA) chest X-ray, which is commonly used in the field of medical treatment for detecting and controlling lung diseases. These X-ray COVID-19 samples were collected from different websites, such as GitHub, Radiopaedia, The Cancer Imaging Archive (TCIA), and the Italian Society of Radiology (SIRM). Whereas, normal cases X-ray samples and pneumonia samples were collected from the Kaggle repository and the NIH dataset.Moreover, there is also a metadata.csv file associated with each folder.

To increase the number of COVID-19 samples, a dataset of 912 augmented images was collected from Mendeley [5]. This comprehensive collection of lungs X-ray images serves as a valuable resource for the development of machine learning algorithms aimed at diagnosing and monitoring lung diseases. The dataset provides a detail representation of lungs X-ray images so consequently, machine learning algorithm is able to implement on it.

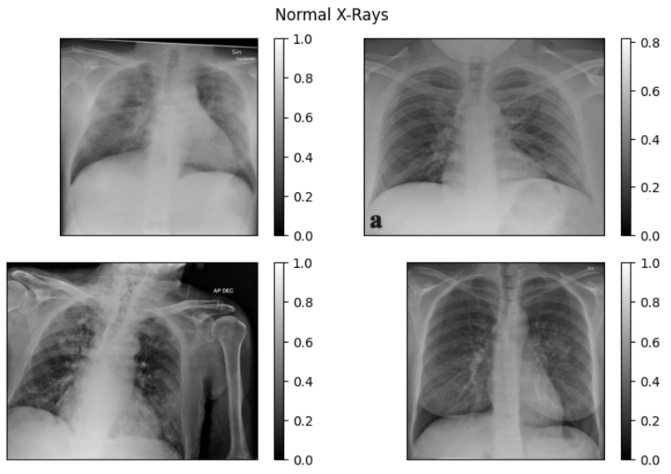

Fig. 4: Grayscale Images

After applying preprocessing techniques to load, resize, converting into grayscaling and rescaling the dataset by using tensor flow libraries the dataset is ready to make further compatible with deep learning models.Then the images are resized to 256x256 (width and height) pixels. This is done on purpose because large size images consume more memory and more computational resources, whereas smaller images are easier to manage but they do not retain all the necessary information for the analysis. The optimal image size depends on the particular use case and its a trade-off between computational cost and information content.

The resized images are then stored as arrays, along with labels and metadata, in a Python dictionary. The labels represent the class of the image, such as 'covid', 'pneumonia', or 'normal'. The metadata is a description of the dataset, including the size of the images, i.e., 256x256 pixels. The dictionary is then saved to a file using the joblib library, with the filename specified as pklname widthxheightpx.pkl, where pklname is a user-specified string, and width and height are the target dimensions of the images. In this way, the preprocessing step enables efficient and convenient storage and retrieval of the X-ray images, and the resizing step ensures that the images are of a manageable size while retaining enough information for the analysis. For our deployed CNN the process of converting categorical labels into numerical values is known as label encoding. In this specific case, the labels 'covid', 'normal' and 'pneumonia' are transformed into the numerical values of 0, 1 and 2, respectively. This numerical representation is then further transformed into a one-hot encoded representation to prepare the data for model training. The one-hot encoded representation is a vector of size (1,3), where the elements are either 0 or 1, and there is exactly one 1 in each vector, indicating the presence of a specific label. This process of transforming categorical labels into one-hot encoded arrays is necessary to ensure that the model can learn the relationships between the inputs and outputs effectively.

V. Learning Framework

In this section, author is presenting learning algorithms and techniques that is used and developed to solve the probelm at hand. There CNN models, a DenseNet-121 and Resnet50 has been utilized in this research. First we have developed and test A model that consist of convolutional Neural Network and some dense layers, to get the output. The structure and architectures are as follows.

A. Convolutional Neural Network: 2D convolutional layer plays a vital role in deep learning model. Mathematically it can be represented as follows: the input feature map is represented by a tensor X of shape (N, H, W, C'input), where N is the batch size, H and W are the height and width of the feature map, and CË™input is the number of channels. The 2D convolutional layer applies a set of filters represented by a tensor W of shape (f'height, f'weight, C'input, C'output), where f'h and f'w are the height and width of the filters and C'output is the number of filters. The stride of the convolution operation is represented by (s'height, s'weight), and the output of the 2D convolutional layer is represented by a tensor Y of shape (N, H'output, W'output, C'output). The height and width of the output feature map can be computed using the following formulas:

$$H_{output} = 1 + \frac{H - f_{height}}{s_{height}} \tag{1}$$

$$W_{output} = 1 + \frac{W - f_{weight}}{s_{weight}} \tag{2}$$

Every component of Y is determined as the dot product of local receptive field values in X and corresponding wieghts in W. A b bias and then an activation function f is added to introduce some non-linearity in model which allows the model to learn complex pattern and improve performance. ReLu set the output same if the input is greater or equal to zero. However, if the input is negative it is set to zero. Mathematically it is written as:

$$f(x) = \max(0, x) \tag{3}$$

As described earlier, We have made 3 custom CNN models and named it as Model-1, Model-3, Model-4, The basic structure is the same but slightly modifying and adding some more Convolutinal Layers.

1. Model-1 : Our first CNN consist of an input layer two different blocks of CNN layers, that perform different kind of operation to extract features from input and make prediction. Very first convolutional layer of the model is two dimensional (2d), consist of 25 filters, kernel size of (5,5), stride of (1,1) and ReLU activation function and the padding is set to *same* and ReLu activation function (Sigmoid function is also applied but it perform worst than ReLu). Then the proceeding layer is a maxpool2D layer with pool size of (2, 2) and stride of (2, 2), that decreases the spatial dimensions of the feature while keeping the most important information. The activation of the proceeding layer is then normalized with the help of batch normalization, with normalization parameters (axis = -1, momentum = 0.99, and epsilon = 0.001). Considered that Model-2 is same in the implementation only trained with augmented data.

Second block of CNN layer is almost same but with increase in filter upto 50 and stride of 2 X 2. To get the ouptut form model, feature maps are then flattened and passed through 2 dense layer with 100 neurons and 50 neurons with ReLU activation along with Dropout layer with 0.25. It means that during training, 25% of the input units (neurons) will be randomly set to zero at each iteration. This is done on purpose to prevent overfitting and improve generalization.

Flatten layer is used to transform multi-dimensional shape (2D or 3D) into a long single vector and helps the structure of model to allow to connect to fully connected dense layer, where every neuron is connected to every other neuron in the previous layer.

The last output layer is always a ,dense layer with 3 neurons because we only have three different classes to predict, and softmax activation fucntion is used. It is significantly used for multicalss classfication which enables to make probabilistic predictions and ease the training procedure.

2. Model-3 : It consist of three blocks of CNN layers. First CNN block consist of 2d convolutional layer with with 25 filters and kernel size 1 5 X 5 and stride of 1. Following with maxpooling layer. Second Convolutional layer contains 2d convolutional layer with inreased in filter from 25 to 50 and same maxpooling layer but an additional batchnormalization layer , with normalization parameters (axis = -1, momentum = 0.99, and epsilon = 0.001). Third block is similar with organization of layer with improved kernel size upto 70 and an dropout layer with 0.25 of value. Then flatten layers is and two dense layers has been applied with 100 and 50 neuron

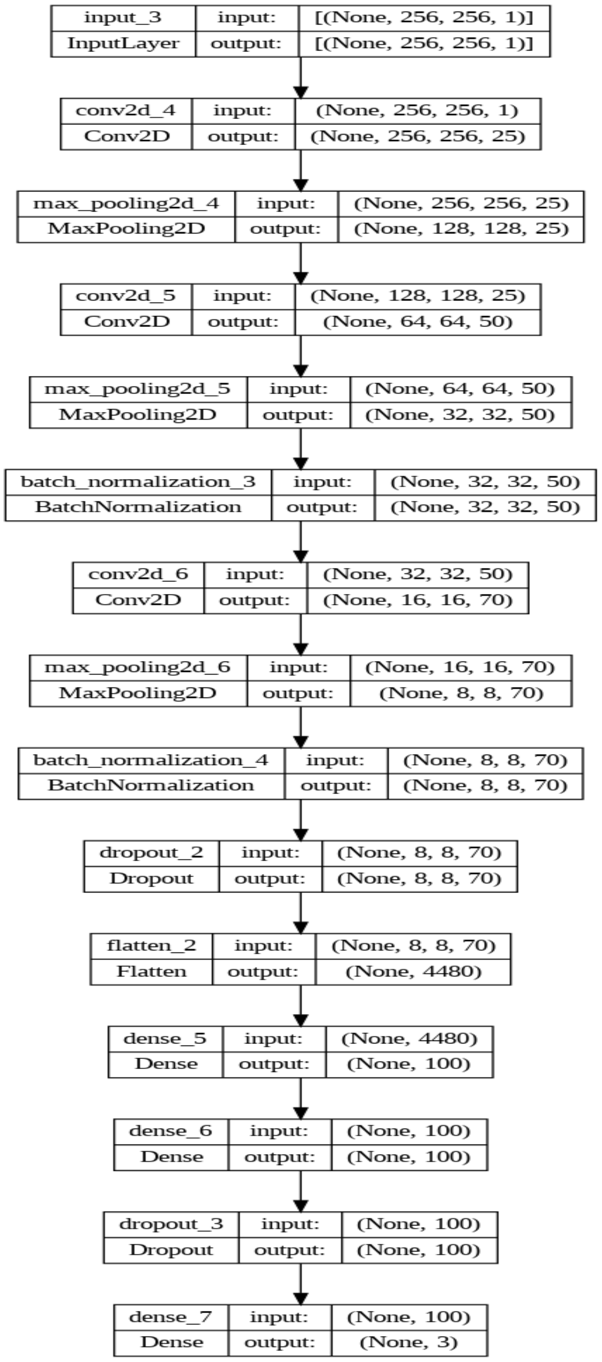
input_3 | input: | [(None, 256, 256, 1)]
InputLayer | output: | [(None, 256, 256, 1)]
conv2d_4 | input: | (None, 256, 256, 1)
Conv2D | output: | (None, 256, 256, 25)
max_pooling2d_4 | input: | (None, 256, 256, 25)
MaxPooling2D | output: | (None, 128, 128, 25)
conv2d_5 | input: | (None, 128, 128, 25)
Conv2D | output: | (None, 64, 64, 50)
max_pooling2d_5 | input: | (None, 64, 64, 50)
MaxPooling2D | output: | (None, 32, 32, 50)
batch_normalization_3 | input: | (None, 32, 32, 50)
BatchNormalization | output: | (None, 32, 32, 50)
conv2d_6 | input: | (None, 32, 32, 50)
Conv2D | output: | (None, 16, 16, 70)
max_pooling2d_6 | input: | (None, 16, 16, 70)
MaxPooling2D | output: | (None, 8, 8, 70)
batch_normalization_4 | input: | (None, 8, 8, 70)
BatchNormalization | output: | (None, 8, 8, 70)
dropout_2 | input: | (None, 8, 8, 70)
Dropout | output: | (None, 8, 8, 70)
flatten_2 | input: | (None, 8, 8, 70)
Flatten | output: | (None, 4480)
dense_5 | input: | (None, 4480)
Dense | output: | (None, 100)
dense_6 | input: | (None, 100)
Dense | output: | (None, 100)
dropout_3 | input: | (None, 100)
Dropout | output: | (None, 100)
dense_7 | input: | (None, 100)
Dense | output: | (None, 3)

Fig. 5: Model 3 structure

respectively. The last layer is always similar as in the Model-1.

3. Model-4 : The configuration of this model is slightly different from Model-3 but it also contains three blocks of CNN layers. First CNN block consist of 2d convolutional layer with with 25 filters and kernel size 1 5 X 5 and stride of 1. Following with maxpooling layer and *BatchNormalization* with same settings in the previous models. Second block also have 2d convolutional layer with filter upto 50, a maxpooling layer, batchnormalization layer , with normalization parameters (axis = -1, momentum = 0.99, and epsilon = 0.001). and droputout layer with 30% of ratio.

Similarly as in Model-3, third block have same layers with 75 filters in 2d convolutional layers and kernel size of 3 X 3, an additional dropout layer with a value of 0.4. Follwing with flatten layer and two dense layer. 100 neuron in first dense layer and an incrase of 150 neuron in the second dense layer. The last layer is similar with first and second model.

The optimization and loss strategy are having similar setting for all the models tested in this stud, details are described as follows.

Optimization Methods: For optimization, both SGD and adam have been tested but SGD was not suitable for this problem. Adam optimizer is used with a learning rate of 0.0001, the learning rate determines the step size to be taken during each parameter update, it force the convergence and stability of the training process. Adam (adaptive moment estimation) updates the model parameters based from the loss calculated during training process. It combines Adaptive Gradient Algorithm and Root Mean Square Propagation and automatically adapts the learning rate.

Categorical Crossentropy: To predict probability over all classes *categorical crossentropy* loss is used, which is most commonly used to multi-class problem. Basically, it computes mismatch between the predicted distribution and the true class distribution of the different categories by computing average negative logarithm of the predicted class for the true class labels. It allows the model to assign high probabilities to the correct classes and penalizes deviations from the true labels. It is written as follows:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(p_{ij}) \tag{4}$$

B. DenseNet121: In 2017, Huang et al. in their paper titled ”Densely Connected Convolutional Networks” has introduced this concept to address the gradient vanishing problem and encouraging better information flow throughout the entire network. This means that when the channel for information from the input to the output layers lengthens, certain data may ’vanish’ or be lost, reducing the network’s capacity to train effectively. They introduced dense connections between layers. whereas , in conventional CNNs, each layer takes input only from its immediate predecessor. However, DenseNet-121 allows each layer to receive inputs not only from the previous layer but also directly from all preceding layers. This connectivity creates a dense and direct information pathway, which enables the network to extract features from different depths and increase overall learning capability fot the model.

Components of DenseNet121:

1) Connectivity

2) DenseBlocks
3) Growth Rate
4) Bottleneck layers

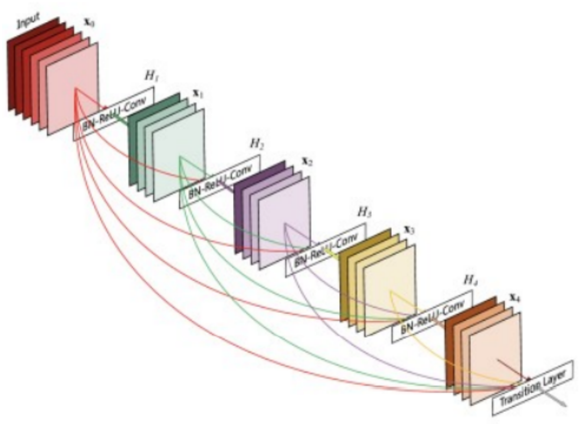

Fig. 6: Dense Architucture

I. Connectivity The connectivity in DenseNet121 is achieved by concatenating the feature maps from all preceding layers and passing them as input to each layer rather then only adding them. It require fewer parameters than conventional CNN by discarding redundant feature map. Mathematically we can write that ,the very first layer receives the feature of all previous layers, $x_0, ..., x_{l-1}$ as input:

$$x_l = H_l([x_1, x_2, ..., x_{l-1}]) \quad (5)$$

where $[x_0, x_1, ..., x_{l-1}]$ shows the concatenation of the feature-maps, such as the output produced in all the layers preceding $l(0, ..., l-1)$. The multiple inputs of H_l are concatenated into a single tensor to ease implementation. we say this model as Dense Convolutional Network (DenseNet) due to its dense connectivity.

II. Denseblocks: is th fundamental component of DenseNet-121. The basic idea on DenseBlocks is to reuse the feature and enable the network to learn more compact representations. By connecting each layer to every preceding layer, the network can access a wide range of feature maps from different depths which in result improve feature learning, as the layers can support both low-level and high-level features during the learning process. To put into practise, DenseNets are divided into DenseBlocks, with the dimensions of the feature-maps remaining constant inside a block but changing the number of filters between them. Transition Layers are the layers between the blocks that lower the number of channels to half of what they were before. From the equation above, H_l is a composite function which perform three consecutive operations: batch normalization (BN), a rectified linear unit (ReLU) and a convolution (Conv). The transition layers are the layers that execute downsampling the input via convolution and pooling operations between two attacted blocks, whereas the feature maps within the dense block are the same size to permit feature concatenation.

III. Growth Rate: Refers to the number of additional feature maps each layer produces comparing to the number of input feature maps it receives. For example, if a layer within a DenseBlock has k input feature maps and a growth rate of g, then it will produce $k + g$ output feature maps. Passing through every dense layer, the feature map increases in size, with each layer contributing K features on top of the global state (existing features). This parameter, K is referred to as the network's growth rate, and it controls the quantity of data added to each layer of the network. The l^{th} layer has k feature maps if each function H_l produces them.It balances model capacity and computational ability which makes it widely used architecture in the deep learning.

IV. Bottleneck Layers: Total number of the input can be very large despite the fact the each layer produces only K output feature map. Forimproving the efficiency and computational speed, a 1x1 convolution layer can be included as a bottleneck layer before each 3x3 convolution [11]. The number of pixels shifted over the input matrix is referred to as the stride. The filters are shifted 'n' pixels at a time with a stride of 'n' and by default its value is 1. The model structure shows that each denseblock has different number of repeated layers habing two convolutions each. Bottleneck layer is a 1x1 kernel, and a 3x3 kernel, performs the convolution operation. Also, each transition layer has a 1x1 convolutional layer and also with stride of 2 of average pooling layer of 2 X 2. Consequently, DenseNet-121 has the following layers:

1) 1 7x7 Convolution
2) 58 3x3 Convolution
3) 61 1x1 Convolution
4) 4 AvgPool
5) 1 Fully Connected Layer

So there is in total 4 AvgPool and 121 Convolutional layer as its name suggest.

C. ResNet50 : Resifual in ResNet is the key innovation of the architecture, which introduce the blocks of residual connection. These blocks allows the network to be deeper while avoiding the degradation problem, where increasing the network depth leads to diminishing accuracy. ResNet-50, specifically, it consists of fifty layers, making it a deep model.

VI. Results

For implementation, author uses keras and tensorflow libraries, a python programming framework. As mentioned earlier in preprocessing section, the dataset has been resized to 256 X 256 pixel size and then it has been rescalled with maximum value of grayscale which is 255. Moreover, dataset is also augmented using library to make the model more robust. As the author has tested four custom made CNN models in which one is tested without augmentation. DenseNet121 and Resnet has also been tested just sake of comparability.

Author trained each model for 12 epochs and ran the models on GPUs available on by google colab. Moreover, used Adam optimizer [ba33] with a learning rate of 0.0001, which is a replacement optimization algorithm for stochastic gradient descent for training deep learning models. Adam combines the best properties of the AdaGrad and RMSProp algorithms to provide an optimization algorithm that can handle sparse gradients on noisy problems. [12]. Author uses Categorical cross-entropy as the loss function. It's used in multi-class classification models with two or more output labels. A single-hot category encoding value in the form of 0s and 1s is allocated to the output label. If the output label is in integer form, Keras is used to transform it to categorical encoding. Then, we defined metrics, including some functions that are used to judge the performance of our models. Metric functions are similar to loss functions, except that the results from evaluating a metric are not used when training the model. The used metric functions are:

1) **Loss**
2) **Accuracy**
3) **Precision**
4) **Recall**

To analyze the learning curves of a model, I plot the loss, accuracy, recall, and precision metrics for both the training and validation datasets over 12 Epochs. These curves provides into depth, how model's behave over time.

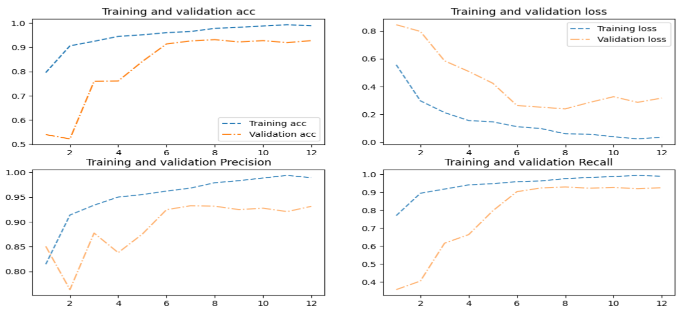

Fig. 7: CNN Model 1 Without Augmentation

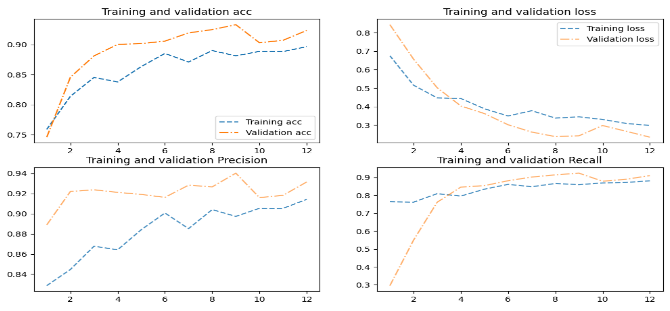

Fig. 8: CNN Model 2

Learning Curves: Figure 7 and 8 shows the loss, accuracy precion and recall curves for the custom CNN Model-1 and Model-2, first model which is trained without augmentation, there has been a significant difference between two learning lines, which indicated that model is overfitting to the training data and perform poorly on unseen validation data. However, Model-2, trained with augmentating data, shows remarkable performance. Despite the data augmentation, the model retains its efficiency and achieves the highest accuracy among the custom CNN models evaluated.

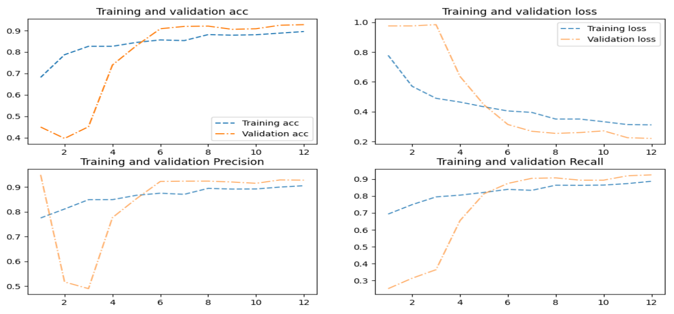

Fig. 9: Improved CNN Model 3

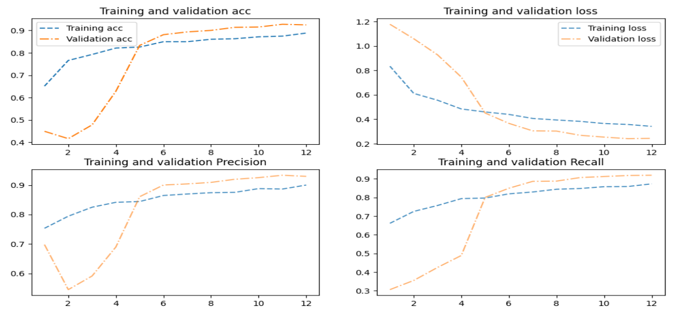

Fig. 10: Improved CNN Model 4

Furthermore, figure 9 and 10 show the learning curves for the Model-3 and Model-4,adding more CNN layers, batch normalization and some dropout layers has not made the models to get the higher accuracy but the model are able to learn in a more refined manner. The learning curves show that as these enhancements were introduced, the difference between the validation and training metrics became remarkably small. This indicates that as the models approached to convergence, both validation and training, loss and accuracy, values becoming increasingly similar.

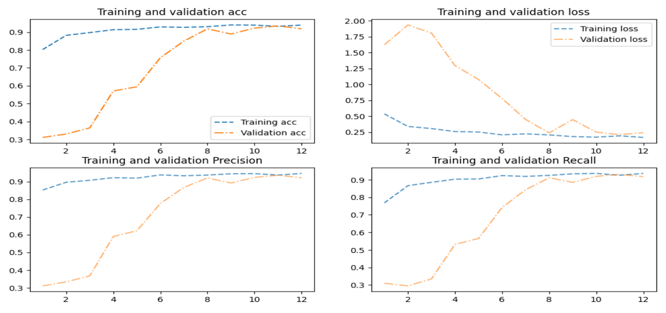

Fig. 11: DenseNet-121

Similarly, for DenseNet-121, a considerably larger model, by examining the learning curves, we can see a point reached where the validation and training measurement metrics values nearly coincide. This convergence of metrics further highlights the strong performance of the model.

Loss and Accuraacy: To analyse the performances of models on unseen data, author tested all models on test dataset, in the Table we can see the results..

TABLE 1: Evaluation on Test dataset

Model	Loss	Accuracy
CNN Model-1 (No Aug)	0.3258	**0.9311**
CNN Model-2	0.2424	**0.9257**
CNN Model-3	0.2233	0.9224
CNN Model-4	0.2525	0.9158
DenseNet-121	**0.2127**	**0.9388**
ResNet50	0.3534	0.8831

In table 1, represent the Loss and Accuracy of all the models that has been tested during implementation.Starting with first model-1, it gained the second highest accuracy, however this model has been trained without augmenting data, the gap between learning curve remains consistently high throughout the training process, and they do not converge really well and its overfitting the training data.

On the other side, the rest of three custom CNN models were trained using augmented data. Among these models, CNN Model-2 achived the highest accuracy of **92.57%** percent. This should be taken into account that, this model only utilized two convolutional layers, proceeding with batch normalization and dropout layers. Despite its simple architecture, CNN Model-2 got exceptional accuracy performance.

During evaluation author tested the CNN Model-3 and Model-4 on same dataset but adding one more convolutional layer block in Model-3 and one conolutional layer with dropout layer in Model-4 Respectively. Despite these modification, We can see from the result, the accuracy has been dropped slightly. However, Model-3 has improved its loss and achieved a significant figure of **0.2233%**. As mentioned earlier, DenseNet-121 outperformed all the models with highest accuracy of **93.88%**. This accuracy is not marginally higher than that exhibited by a simple two layer CNN Model-2. But the performance of DenseNet-121 shows the advantages of its deep architecture and dense connectivity patterns in capturing complex patterns and features within the images. Meanwhile, ResNet50 could not performed well and got the lowest accuracy and reach only untill 88%, suggesting that its architectre may not suitable for the specific task or dataset.

By introducing variations in the training data through augmentation, the model becomes more robust and adaptable, leading to improved performance on unseen data. The augmentation process effectively increases the diversity of the training dataset, enabling the model to learn more generalized features, which translates into better performance on the validation dataset.

Confusion Matrix: A confusion matrix provides a tabular representation of the predicted and actual classifications made by a classification algorithm on a given dataset. It allows us to understand the performance of the model by analyzing the counts of true positive, true negative, false positive, and false negative predictions.

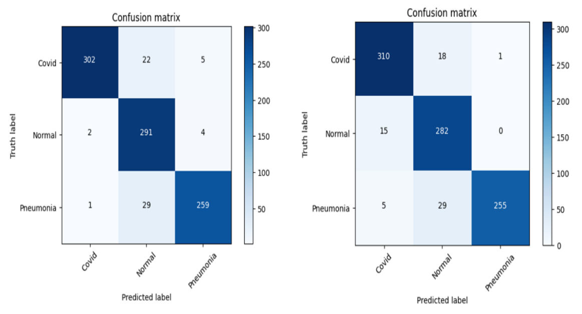

Fig. 12: Confusion Matrix, (a) Model-1 , (b) Model-2

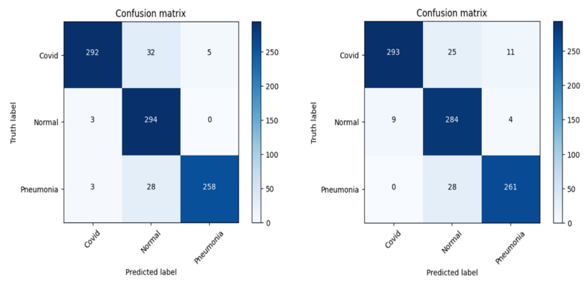

Fig. 13: Confusion Matrix, (a) Model-3 , (b) Model-4

In matrix, Rows representing ground truth labes while the columns showing the predicted labels by the models, Each cell in the matrix indicates the number of instances that counted into particular combination of true and predicted labels.

In figure 12, we can visualize the confusion matrix for Model-1 (trained without augmenation) and Model-2, performances. We can see the Model-1 is able to correctly predict 302 times that the image belong to covid cases, 22 times it was covid but model recognize it as normal case and 5 times it predict as pneumonia. We can also see the performance of Model-3 and Model-4 in figure 13.

VII. Concluding Remarks

Accurately diagnosing Lung diseases can contribute significantly to mitigate their impact and minimize the global death rate. However, conventional methods that is being used is not precise to accurately detect disease and they are also time consuming. This research is conducted to study and improve the classification of Lung diseases using chest X-ray image dataset by utilizing the effectiveness of Convolutional Neural Networks. After training and analysing the performance of various models, the findings indicate that the custom build CNN model, Model-1, Model-2 , Model-3 and Model-4 achieved the accuracy of 93.11% , 92.57%,92.24%, 91.58%, 93.88%, 88.31%respectively. Despite of adding more convolutional layers, batch-normalization and dropout layers, does not allows to improve the performance, however then make models to reduce the margin between learning curves that indicates model, learn pattern efficiently and improve performance on unseen data.

Author also tested the DenseNet-121 and ResNet50 model only for the sake of comparision. DenseNet-121 was the top performer amongst all the network and got almost 94% of accuracy and got lowest loss with value of 0.2127, while Resnet performed poorly.

By analysing the confusion matrix, it can be more authenticated that the Model-2 with high accuracy, is also perform better over correctly classifying different classes with minimum errors.

In-addition, author also observe that the augmenting the data also helps the model to learn more complex pattern in the input images and validation and training loss becomes significantly close to each other. During analysing, author also observe that, using augmentation on the model as part of its layer, does not helps model to understand the pattern correctly and got result very different after each training. As dataset consist of images of processing images and without a GPU can be difficult due to the high computational requirements of modern ML algorithms. These algorithms often involve complex mathematical operations, like matrix multiplications, which demand high processing power. Author is delighted with the positive outcome of this challenging task, as it has bben successfully tackled the issue of overfitting and assessed the performance of different architectures. For futurework, author is interested improving models accuracy by applying different machine learning approaches.

References

[1] M.Y. Ng, E. Y. Lee, J. Yang, F. Yang, X. Li, H. Wang, M. M.s. Lui, C. S.Y. Lo, B. Leung, P.L. Khong, et al., "Imaging profile of the covid-19 infection: *radiologic findings and literature review," Radiology*: Cardiothoracic Imaging, vol. 2, no. 1, p. e200034, 2020.

[2] B. Y. Yang, L. M. Barnard, J. M. Emert, C. Drucker, L. Schwarcz, C. R. Counts, D. L. Murphy, S. Guan, K. Kume, K. Rodriquez, et al., "*Clinical characteristics of patients with coronavirus disease 2019 (covid - 19) receiving emergency medical services in king county, washington* ," JAMA network open, vol. 3, no. 7, pp. e2014549 - e2014549, 2020.

[3] A. Nair, J. Rodrigues, S. Hare, A. Edey, A. Devaraj, J. Jacob, A. Johnstone, R. McStay, E. Denton, and G. Robinson, "*A british society of thoracic imaging statement: considerations in designing local imaging diagnostic algorithms for the covid -19 pandemic*," Clinical radiology, vol. 75, no. 5, pp. 329 334, 2020.

[4] M. Hasan, S. Ahmed, Z. Abdullah, M. Monirujjaman Khan, D. Anand, A. Singh, M. AlZain, and M. Masud, "Deep learning approaches for detecting pneumonia in covid-19 patients by analyzing chest x-ray images," Mathematical Problems in Engineering, vol. 2021, 2021.

[5] Asraf, A. and Islam, Z. (2021),"COVID19, Pneumonia and Normal Chest X-ray PA Dataset," Mendeley Data, vol. 1, doi: 10.17632/jctsfj2sfn.1.

[6] Fernandez-Grandon, C. I. Soto, D. Zabala-Blanco, W. Alavia, and V. Garcia, *ANN classification using GLCM and HOG features for COVID-19 and Pneumonia detection from Chest X-rays,a C in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Nov. 2021.*

[7] J. P. Cohen, L. Dao, K. Roth, P. Morrison, Y. Bengio, A. F. Abbasi, B. Shen, H. K. Mahsa, M. Ghassemi, H. Li, et al., *"Predicting covid-19 pneumonia severity on chest x-ray with deep learning," Cureus, vol. 12, no. 7, 2020.*

[8] Wang, L., et al. *"COVID-Net: A tailored deep convolutional neural network design for detection of COVID-19 cases from chest X-ray images,"* Sci Rep, vol. 10, no. 19549, 2020.

[9] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers, *"Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases,"* in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2097 2106, 2017.

[10] K. Sriporn, C.-F. Tsai, C.-E. Tsai, and P. Wang, "Analyzing Lung Disease Using Highly Effective Deep Learning Techniques," Healthcare (Basel), vol. 8, no. 2, pp. 107, Jun. 2020.

[11] L. v. d. M. Gao Huang, Zhuang Liu and K. Q. Weinberger, "Densely connected convolutional networks," 2018.

[12] D. P. Kingma and J. Ba,"Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.