Determining Pneumonia via X-ray image using Neural Network

BY

Md. Nayemur Rahman
ID: 181-15-1963

Md. Ashraful Alam
ID: 181-15-1844
AND

Fahim Ahammad Niloy
ID: 181-15-1872

This Report Presented in Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Computer Science and Engineering

Supervised By

Ohidujjaman
Assitant Professor
Department of CSE
Daffodil International University

Co-Supervised By

Md. Mahfujur Rahman
Senior Lecturer
Department of CSE
Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

JANUARY 2022

APPROVAL

This project titled "**Determining Pneumonia via X-ray image using Neural Network**", submitted by **Md. Nayemur Rahman, Md. Ashraful Alam** and **Fahim Ahammad Niloy** to the Department of Computer Science and Engineering, Daffodil International University, has been accepted as satisfactory for the partial fulfilment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on **18th January, 2022**.

DECLARATION

We hereby declare that this project has been done by us under the supervision of **Ohidujjaman, Assitant Professor, Department of CSE** Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere for the award of any degree or diploma.

Supervised by:

[illegible]

____

Ohidujjaman
Assitant Professor
Department of CSE
Daffodil International University

Co-Supervised by:

[illegible]

____

Md. Mahfujur Rahman
Senior Lecturer
Department of CSE
Daffodil International University

Submitted by:

Md. Nayemur Rahman

____

Md. Nayemur Rahman
ID: 181-15-1963
Department of CSE
Daffodil International University

MD. Ashraful Alam

____

Md. Ashraful Alam
ID: 181-15-1844
Department of CSE
Daffodil International University

Fahim Ahammad Niloy

____

Fahim Ahammad Niloy
ID: 181-15-1872
Department of CSE
Daffodil International University

ACKNOWLEDGEMENT

We are grateful to our supervisor, **Ohidujjaman**, **Assitant Professor**, Department of CSE, Daffodil International University, Dhaka, and desire to express our heartfelt gratitude. We encountered several implementation issues with this project, but our supervisor's extensive experience and genuine interest in the topic of "*Artificial Intelligence*" aided us in completing it. This assignment could not have been completed without his superb scholarly leadership, composure, encouragement, enthusiastic supervision, helpful remarks, good advice, and examining numerous inferior drafts and revising them at each stage. It would not have been feasible to finish without his unwavering support.

We would like to express our heartfelt gratitude to our co-supervisor, Md. Mahfujur Rahman, who assisted us in this project in a variety of ways. We are grateful to Dr. Touhid Bhuiyan, Professor and Head, Department of CSE, other academic members, and the entire staff of Daffodil International University's CSE Department.

We are grateful to Daffodil International University's project committee for giving us the opportunity to work on this proheadject. We are also grateful to some of our classmates for their encouragement and assistance.

Finally, we express our deepest gratitude to Almighty Allah for His favor. Finally, we appreciate our parents' love, prayers, unwavering support, and for preparing us for the future.

ABSTRACT

Despite the fact that pneumonia is a vaccine-preventable disease, it has become a global health concern. Every year, 800,000 children die from pneumonia. Pneumonia has been the cause of death in 17 percent of Bangladeshi kids under the age of 5. X-rays of the chest can be used to detect it. A qualified radiologists is essential for analysis. Even with an experienced radiologist, analyzing chest X-rays is tough. Machines can readily complete jobs that are difficult for humans to complete. Pneumonia can be diagnosed using machine learning and image processing. Machine learning (ML) is a branch of artificial intelligence that helps computers learn and make decisions on their own. Efficient pretrained transfer learning based VGG16, VGG19 and ResNet50 architectures are used to develop the proposed model which could assist both doctors and patients. The convolutional neural network is one of the machine learning algorithms. Image processing has also been used to help computers comprehend photos and identify Pneumonia more precisely. We used augmentation approaches even after training our model with over 6,000 pictures. With the use of a chest x-ray report, users will be able to quickly determine if they have Pneumonia in a relatively short period of time. In order to improve the accuracy, we added several extra layers to all the model. To enhance the number of images in a balanced way, data augmentation techniques were applied. The comparison between VGG16, VGG19, and ResNet50 is shown in this project. Whereas, VGG16 has obtained 94.5 percent accuracy, which is the greatest among the other algorithms implemented, indicating that it can classify normal and pneumonia. Other image-based classification problems can also apply the proposed architecture.

Keywords: VGG16, Artificial Intelligence, CNN, Image Processing, Data Augmentation, Deep Learning.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
Introduction

1.1 Introduction

Our health is the most important component of our lives. Humans seem to want to be happy regardless of their lifetime, but then all know that satisfaction is reliant upon his health. We must be disease-free if we want to be healthy. Thousands, if not millions, of diseases exist in our globe. To stay healthy, we must constantly combat all of these disorders. Coronavirus has had a huge impact on our daily life in recent years. As a result of the sickness, the whole globe came to a standstill. We have, however, figured out how to combat this infection over time. We can defend ourselves from this infection by following a few easy guidelines and regulations. In the same way, we can defend us from many other diseases. However, we wouldn't want to cover every one of these diseases in this study because it would take a long time and take us away from the key points. We'd rather talk about the disease called Pneumonia. Medical science is one of the most essential sectors of human civilization, and every day saving thousands of lives. Pneumonia is a severe respiratory infection that affects one or both lungs of a person [1]. This disease might be bacterial, viral, or fungal. One major difference between viral and bacterial pneumonia is that viral pneumonia normally improves on its own, whereas bacterial pneumonia requires antibiotic treatment [2]. Coughing, fever, chest pain, breathing problems, vomiting, and excessive exhaustion are all common Pneumonia signs [1].

Children and older persons with weakened immune systems, are particularly vulnerable to this disease. Approximately 80,000 kids under the age of 5 in our country are admitted to hospitals each year with Pneumonia infection. Every 39 seconds, a child dies and per year over 50,000 children die because of Pneumonia from Pneumonia. Pneumonia is the most common illness among children in South Asian countries. Pneumonia has a higher death rate than other lethal diseases such as malaria, AIDS, and tuberculosis combined. Approximately 80,000 kids under the age of 5 are admitted to hospitals each year with Pneumonia viral infection. Pneumonia can be cured if discovered early enough. Currently, chest X-rays are widely available and most popular methods for detecting pneumonia [3].

CT scanning is not preferred over X-ray scanning because X-ray scanning takes less time than CT scanning. Doctors are under a lot of pressure to check and treat every patient. Many patients die in inconvenient ways as a result of improper pneumonia diagnosis and treatment. Diseases, since we all understand, respond better to early detection. Furthermore, diagnosis of the pneumonia disease takes so much time, and required capable radiologist. Another problem is that, characteristics that describe the disease's actual appearance are particularly associated with many other diseases. So, it is hard to diagnose this disease for a radiologist. As a result, we require a simple and effective method for obtaining pneumonia findings, so that anyone can quickly identify Pneumonia and take appropriate action. In this day and age, we prefer speedy digital solutions. In locations where there is a shortage of qualified radiologists, computer-assisted diagnosis employing deep learning and machine learning approaches is very effective. Deep learning algorithms overcome all of these difficulties, and overall disease prediction accuracy is sometimes higher than that of a regular radiologist [4]. We employed a customized Vgg16 model to perform optimum classification tasks in this study, and we also tested whether this model can automatically predict a pneumonic or normal patient efficiently. The proposed model accuracy is almost 90 percent and it can successfully detect pneumonia disease. We hope that this research-based project will benefit both doctors and patients.

1.2 Statement of the Problem

As our country is a developing country, our medical system is not yet very developed and efficient. Yet old equipment is used in medical services, which sometimes gives wrong results. Our health service is not as advanced as that of rich nations. Additionally, the prior health system was unable to deliver adequate services to patients due to our country's large population. Villagers need move to towns to get good medical care. In many patients it is too late to go from village to town. The disease becomes more severe due to late treatment. We know that getting a test report takes a long time. As a matter of fact, emergency care patients typically face challenges. The major reason for this issue is that current approaches for detecting various diseases are not automatic. Diagnosis of pneumonia is often a matter of time and requires a skilled radiologist. Most of the hospitals are short of test equipment and due to patient pressure, patients are late in getting test results. In some complicated cases, reports from doctors are often inaccurate. Trained radiologists are expensive so many hospitals are unable to hire them. As a result, we need an automated system that can identify pneumonia efficiently by chest X-ray images.

1.3 Scope and Limitation of the study

The fourth revolution will arrive by the help of AI, in which our written programs will be written in less time. As a result, AI will be the period of the future. Without AI, we won't be able to think up anything new. With the help of AI, everything will be automated. In our study, we used ML algorithms, DNN, AI, & image processing, and we feel that our study will contribute to those same sectors. Our study's main purpose is to gain a better understanding of Machine and Deep learning techniques. This study also help us acquire knowledge and skills in the field of artificial intelligence. The study also provides ideas on how to extract features from image files and solve various image classification problems.

We had to go through several complications while developing the project. Processing an image type data is complicated and takes more time than text data or any other type of data. As a result, high-end picture file handling devices are required, which are relatively expensive. We had a difficult time gathering these X-ray images. We gathered data from the real world. Because of the corona period, collecting this data was difficult. There was a lot of useless information in the data we gathered. We had to process the collected data and feed the model without those inputs, that was really challenging part for us. We had to check the accuracy of multiple models to acquire good accuracy, which took long period of time.

1.4 Motivation

Pneumonia attacks a great amount of people in poor and developing countries, especially kids, and is distinguished by health conditions such as filthy conditions, congestion, and obesity, but also a shortage of qualified medical care. Early detection of pneumonia can reduce mortality. Chest X-rays are used to diagnose pneumonia. As a result, we must rely on a radiologist to diagnose the condition. We are developing an automated technique to detect Pneumonia in order to reduce our reliance on radiologists. Our main motivation is to relieve the load on patients, doctors, and radiologists.

1.5 Objectives

The major goal of this study is to develop a low-cost, automated tool for radiologists and medical specialists to cross-validate their diagnoses and identify alternative potentially undetected outcomes.

Other goals include:

- ✓ Diagnose Pneumonia as quickly as possible utilizing the X-ray report.
- ✓ Learn about machine learning and artificial intelligence (AI).
- ✓ For the picture classification application, use a variety of Deep Learning and Transfer Learning models.
- ✓ Learn how to use a computer to read visual data.
- ✓ Identify pneumonia patients more quickly and precisely.
- ✓ Patients will be able to diagnose Pneumonia with only a chest X-ray report.
- ✓ Contribute to society in a meaningful way.

CHAPTER 2
Background Study

2.1 Literature Review

While working on this research, we collected a number of research papers from the internet for learning purpose. Many researchers applied a variety of models to accurately diagnose Pneumonia. In this research, we upgraded the VGG16 Neural Network model by adding layers to identify Pneumonia. We use pneumonia and regular patient chest X-ray to train the proposed model.

Chest X-ray images give important medical information that can help in the prevention and treatment of hospital patients. The manual testing procedure was ineffective when dealing with Thousands of medical imaging data. A lot of work has been done in the fields of medical disease diagnosis and machine learning. However, there hasn't been much work on pneumonia detection, only a few researches have been found. Deep learning was used by some of the researchers to diagnose pneumonia using an image x-ray, and the outcomes were generally positive [5]. The recent upgrade to the deep learning model has helped algorithms gain better accuracy in image classification tasks such as heart disease classification [10], medical image classification and segmentation [7], hemorrhage detection [8], diagnosis of oral cancer [9] and plant disease detection [6].

E. Ayan et al. employed the Xception and VGG-16 models, which were trained using 5856 chest X-ray images. They also applied image augmentation and transfer learning techniques. The Vgg16 model shows poor result in detecting pneumonia, while the Xception model shows poor result in detecting normal cases[11]. In certain studies, focal loss was utilized to prevent bias in an unbalanced dataset [12]. T. Rajasenbagam et al. developed a deep Convolutional Neural Network model for determining Pneumonia via chest X-ray images using VGG19 network. They used 12,000 X-ray images for training the model and had a classification accuracy of 99.34 percent. They evaluated the results of their model to three transfer learning techniques (VGG16Net, InceptionNet, and AlexNet) and found that the model's comparative result was greater than the other models [13].

The authors Khalid El Asnaoui et al. compare 9 Deep Learning architectures for automatic binary pneumonia classification and answered which techniques beats other DCNN approaches by a substantial margin. The accuracy of Inception Resnet V2, MobileNet V2, and Resnet50 is greater than 96 percent, while the other six models are less efficient [14]. Garima Verma et al. presented a research that uses chest X-rays to identify Pneumonia. They employed CNN for the implementation, which had six convolutional layers. For categorization and faster processing, this allows us to use fewer complicated layers in the Deep Learning model [15]. Rajpurkar et al. trained a model using ChXNet, which is a deep CNN with 121 layers. Their model detected fourteen other diseases in addition to pneumonia. They compared the effectiveness of their model to professional radiologists' practice. They stated in their research paper that their model is better than others and they hopes for improvement in the medical field [5].

Transfer learning has also been used by some researchers to determineif a person has pneumonia [16]. Ibrahim et al. [17] proposed Transfer learning approaches for classifying normal and pneumonia (viral, bacterial) patient’s chest X-ray images using AlexNet model. The model attained 94.4 percent accuracy, the results were inadequate and unfit for practical usage.

2.2 Comparative Analysis

The proposed model is compared to the accuracy of several existing methods. Table 2.2.1 shows the accuracy of the existing methods.

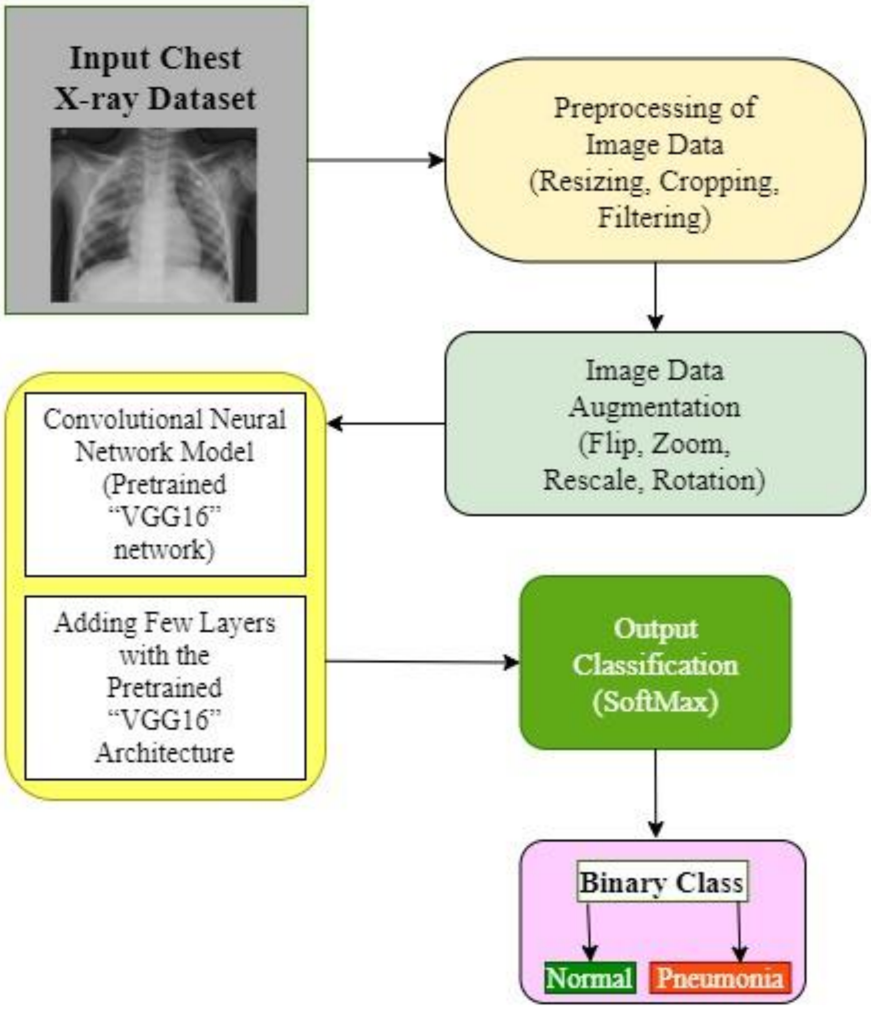
Table: 2.2.1 Comparative Analysis

Author	**Model**	**Accuracy (Percent)**
G. Liang et al. [18]	VGG16	74.2
Hasan et al. [19]	VGG16 model	91.69
Varshni et al. [20]	DenseNet-169 and SVM	80.02
Jain et al. [21]	CNN model (VGG16)	87.17 (Validation)
Garstka et al. [22]	CNN with dropout regularization	90.5 (Binary classification)
Tammina [23]	Pretrained Vgg16 model and image augmentation	86.5(Training) and 95.4 (Validation)
Proposed Model	Modified VGG16	91.67

CHAPTER 3
Methodology

3.1 Methodology

Initially, we collect data of pneumonia patients. The data was then pre-processed and augmented. The VGG16 model is used to identify pneumonia. The model is then trained using the pre-processed data. Finally, we test if our trained model is capable of correctly identifying pneumonia. This section goes over the entire process of the proposed Pneumonia detection model in depth. The study's entire methodology is represented in Figure 3.1.1.



Input Chest X-ray Dataset
Preprocessing of Image Data (Resizing, Cropping, Filtering)
Image Data Augmentation (Flip, Zoom, Rescale, Rotation)
Convolutional Neural Network Model (Pretrained "VGG16" network)
Adding Few Layers with the Pretrained "VGG16" Architecture
Output Classification (SoftMax)
Binary Class
Normal
Pneumonia

Figure: 3.1.1 Proposed Methodology

3.2 Detail Model Schema

In training and test classes, images must be isolated from the chest X-ray picture collection. The scaled and preprocessed photos were then sent through a classifier to determine the disease. After that, each image is passed through a series of layers. It enters the convolutional layer first, followed by the flatten layer, and multiple dense and dropout layer. After that, any optimizer is used to assemble the model. In this situation, we utilized Adam optimizer to compile the code. Predict Normal and Pneumonia disease at the conclusion. In this simple detail schema, the entire operation is depicted.

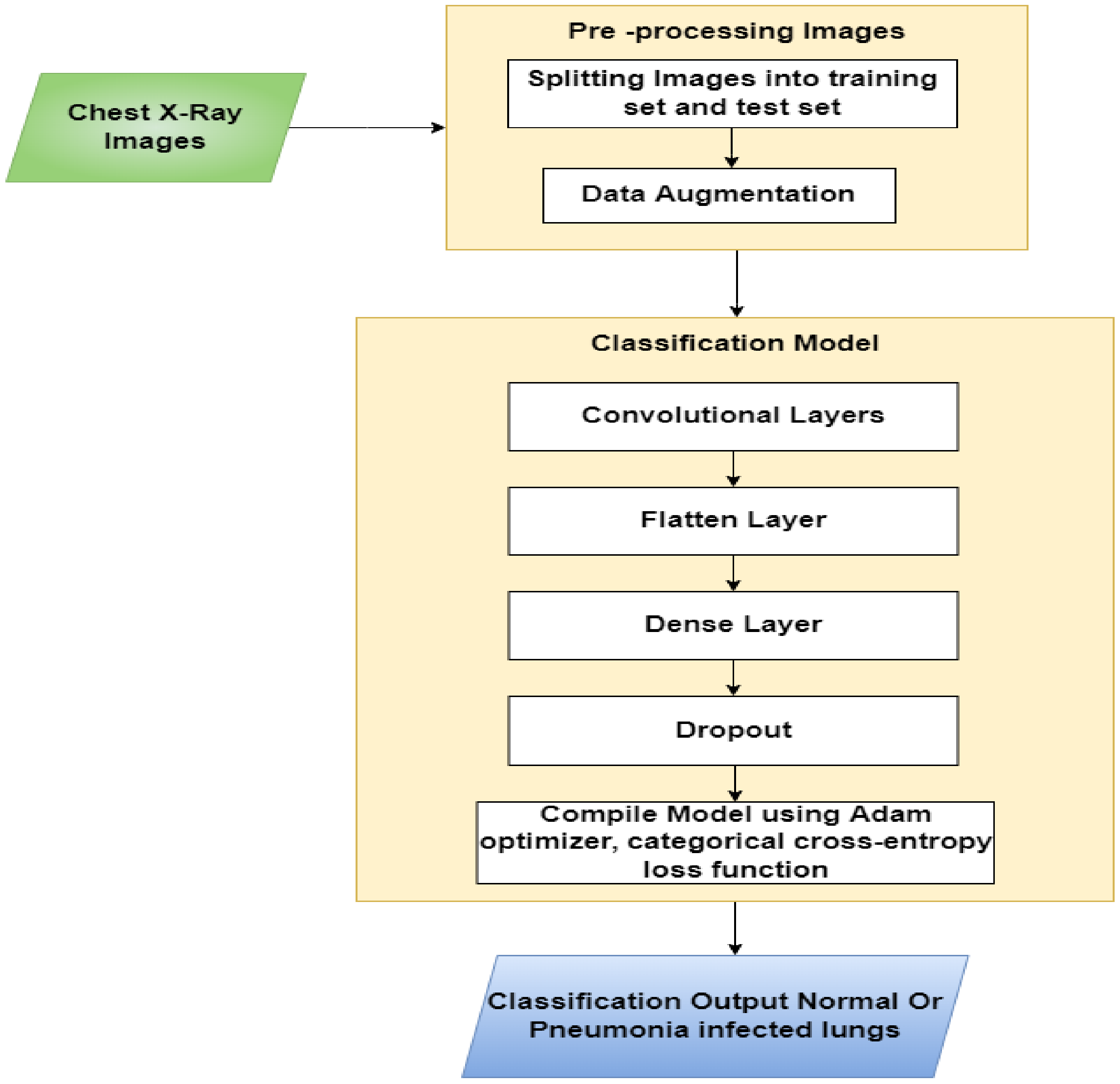

Chest X-Ray Images
Pre -processing Images
Splitting Images into training set and test set
Data Augmentation
Classification Model
Convolutional Layers
Flatten Layer
Dense Layer
Dropout
Compile Model using Adam optimizer, categorical cross-entropy loss function
Classification Output Normal Or Pneumonia infected lungs

Figure: 3.2.1 Detailed schema of the model

3.3 Dataset Collection

We collected 6,000 raw images of normal and pneumonia patient’s chest X-Ray. The entire data was collected from Super Medical Hospital which are located in Bangladesh. We have selected 4151 data from there, where 1362 normal and 2789 pneumonia patients X-Ray images were found. Rest of the data has been reduced in the pre-processing step due to the lack of noise and label problems.

(Normal)

(Pneumonia)

Figure: 3.3.1 Sample of dataset

3.4 Dataset Preprocessing

There was a lot of data in our collected data that was not labeled, and the size and quality of each image was different. The noised X-ray image data was reduced and cleaned from the dataset after collecting images. The pre-trained models on ImageNet will have inputs that are less than or equal to 224*224. The data was organized into three basic folders (training, validation, and test) and two sub-folders (pneumonia and normal) respectively in this study. The training set was employed for model training and includes a total of 3640 (87.69 percent) photos, where, 2480 pneumonia patients' image and 1160 Normal patients' image, respectively. In addition, 494 (11.9 percent) photos were retained separate from the test set, with 300 and 194 images of pneumonia and normal individuals, respectively. Finally, the validation set contains 0.41 percent of the total, with 9 pneumonia and 8 normal patient photos, respectively. However, to increase the dataset's size, data augmentation techniques were applied. The number of images in the dataset is shown in Table 3.4.1.

Table 3.4.1: Number of images in proposed dataset

Dataset		**Number of Images**
Training	Pneumonia	2480
	Normal	1160
Test	Pneumonia	300
	Normal	194
Validation	Pneumonia	9
	Normal	8

3.5 Image Data Augmentation

Rather than collecting new data, professionals might utilize data augmentation to increase the diversity of data samples for training models. To artificially boost the size and quality of the dataset, we used a variety of data augmentation techniques. This procedure assists in the resolving of overfitting issues and improves the model's capacity to generalize during training. The data augmentation was carried out with the help of the Python language, Tensorflow, and the Augmentor library. Figure 3.5.1 shows some samples of augmented images.

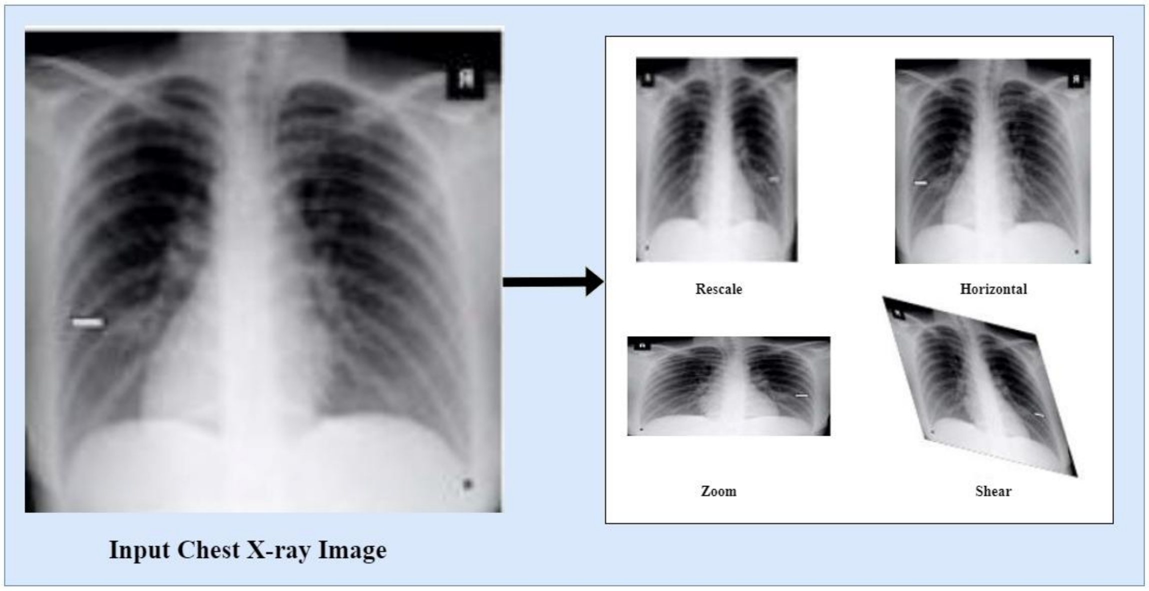


Figure: 3.5.1 Image Augmentation

The rescale operation scale images from integers to floats during the augmentation techniques, the rotation range denotes the range in which the images were randomly rotated during training, which we set to 20 degrees. The width shift range is the 0.2 percent horizontal translation of the images, while the height shift range is the 0.2 percent vertical translation of the images. In addition, a shear range of 0.2 percent shears image controls the angles in a counterclockwise direction, and a zoom range of 0.2 percent randomly zooms in and out the photos. After that, the photos were horizontally flipped during the training process. Finally, set the fill mode to nearest because this is the default setting, which selects the closest pixel value and repeats the process for all empty values.

3.6 Convolutional Neural Network Architectures

The architecture of human brain inspired CNNs. Artificial neurons in CNNs collect inputs, process them, and deliver the result as output, just like a neuron in the brain processes and transmits information throughout the body. This method employs locally trained filters to extract visual information from the input image. A Convolution, Pooling and Fully Linked layer compensate CNN's internal layer structure. This approach generates the most effective results in the image categorization with the fewest errors.

Convolution Layer

This layer is the most basic component of a CNN architecture. The dataset's features are extracted using convolutional layers. Convolutional operations enable a more complex feature representation. The purpose of this layer is to find features in an image's original areas and produce a feature map that decreases their presence in the input data. The fundamental convolution operation can be written using equation 1.

$$X(i,j) = (J * K)(i,j) = \sum_m \sum_n J(i+w, j+h) K(w,h) \quad (1)$$

Here, K (Kernel) representing a 2D filter and w x h representing dimension. The 2D characteristic map produces the value X. Input J is convolved with K to produce X. J * K denotes the convolution operation and the * indicates a discrete convolution procedure.

Pooling Layer

Pooling layers are a vital component of the convolution layer sequences. These layers decrease the spatial size of the neuron bunches outputs at one layer before combining them into a single neuron at the next. The pooling layer determines the highest value, reduces complexity, and hold the most important features. Pooling layers increase feature extraction performance in this approach. These layers allow a 2D filter to be slid over each channel of the feature map. The below formula can be used to calculate output dimensions after a pooling layer operation.

$$(p_h - s + 1)/lx(p_w - s + 1)/lxp_c \quad (2)$$

The feature map height is denoted by ph, width is denoted by pw, and the employed channels number in the feature map is denoted by pc, where s is the filter size and l are the stride length. Some of the pooling layers utilized in CNN are max pooling, global max pooling layers, and average pooling. In general, when used in the input zone, max pooling provides the maximum value.

Fully Connected Layer

This layer is a basic component of CNN which appears after the Pooling layer. Each neuron in the previous layer is connected with each neuron in the next layer, which is a basic component of CNN. The Convolutional and Pooling layers alternately transfer image features, and Fully Connected layer receives the output (image feature) of the last Pooling layer as an input. One or more hidden layers may be applied to classify extracted features in the fully connected layer. The result of the last fully connected layer is then combined with a function called "activation function," which generates output class scores. Sigmoid, Support Vector Machine, SoftMax and other classifiers are used by CNN. In the proposed model, we used the Softmax classifier for classifying the image data of chest X-rays.

3.7 Proposed Model Building & Improvement

The collected dataset of chest x-ray images is used to train and test the VGG16 model. The following is a full description of the model presented in the report:

K. Simonyan and A. Zisserman developed the VGG16 architecture in 2014[24]. On the Imagenet dataset, this architecture achieves Top 5 test accuracy (92.7%). This is a Convolutional Neural Network architecture which is widely used for solving image data classification problems, on a large image dataset. This model uses ImageNet to load pre-trained weights while omitting the fully connected layer head. Instead of having large kernel sized filters and a high number of hyper-parameters, in the convolutional layers, this architecture uses 3X3 size multi kernels and the max pooling layers use multiple 2X2 size kernels.Multiple layers of kernels result enhanced the depth of neural network and enables the neural network to learn and detect more complex features. Multi layers of the kernels enhanced the deepness of the Neural Network (NN) and as a result of this, more complicated features can be trained and recognized by the network.

There are thirteen convolutional, five max-pooling and three fully connected layers in the VGG16 model. The first two convolutional layers have sixty-four kernel filters of 3X3 size. At first, when the input image goes through the first two convolutional layer, its dimension changes to 224 X 224 X 64. The changed result of the previous layers is then passed to the max-pooling layer and output dimensions reduces to 112X112X128. The third and fourth convolutional layer is composed of 128 kernel filters, with a 3X3 filter.

Following these two layers is a max-pooling layer (stride 2), which decreases the dimensions to 56X56X128. Next three (5, 6, 7) convolutional layers are comprise with 3X3 size kernels, 256 filters and output dimensions are 56X56X256. These layers are then followed by a max-pooling layer (stride 2). Convolutional layers with 512 kernel filters, 3X3 size kernels and output dimensions of 28X28X512 compensate the eighth, ninth, and tenth layers. Next with a stride of 1, max-pooling layer follows the previous layers. The eleventh, twelfth, and thirteenth convolutional layers make up with kernel sizes of 3X3 and 512 kernel filters, with output dimensions of 14X14X512. Again, a max-pooling layer (stride 1) follows the previous layers. The fully connected (FC) layers are the fourteenth and fifteenth hidden layers (Dense) of 4096 units. At last, the sixteenth layer is a Thousand-unit (Softmax) output layer. The proposed VGG16 architecture and VGG16 architecture are compared in Figure 3.7.1

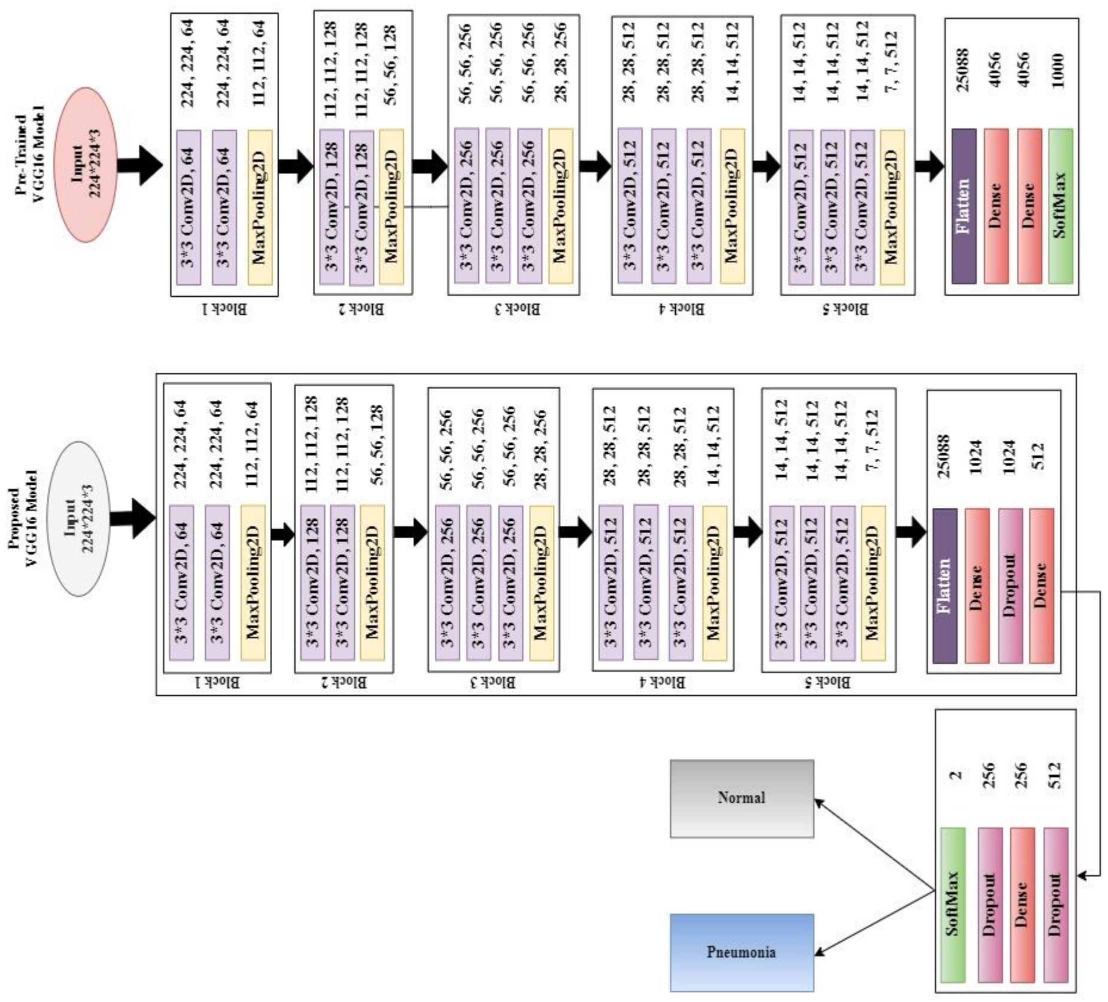


Figure 3.7.1 Proposed vs Main VGG16 model

We further enhanced the basic architecture of the VGG16 by adding some extra layers to fit the Chest X-ray dataset in the network in order to make the proposed model more effective, and reliable. In the proposed method, after the thirteenth convolutional layer, we removed all of VGG16's layers and replace them with a fully connected block. The replacement process begins with a flatten layer, then some dense layers and after all dense layer we added dropout layers which showed in Fig. 3.7.1.

Flattening is used in this study to transform the data from an array of two-dimensional matrices into the appropriate format and pass them as inputs to the dense layer. The previous convolutional layer's output is flattened to create a single feature vector. To allow for quick feedforward execution, the proposed model was flattened. An example of a flattening operation is shown in Figure 3.7.2. Dense layers were used to create the final product, which was optimized. As a result, in the proposed model, we included some Dense layers.

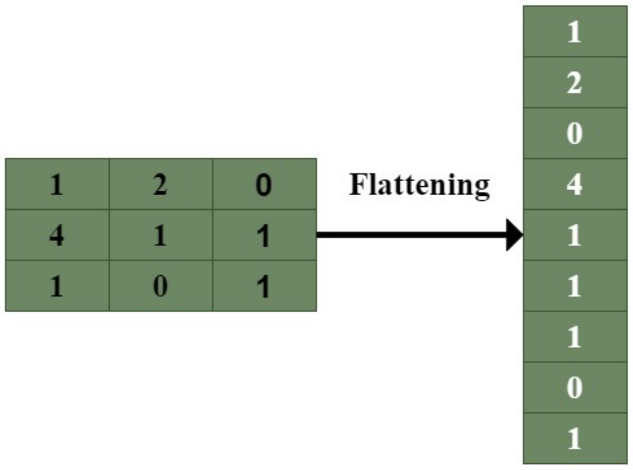


Figure 3.7.2 Falttening Operation

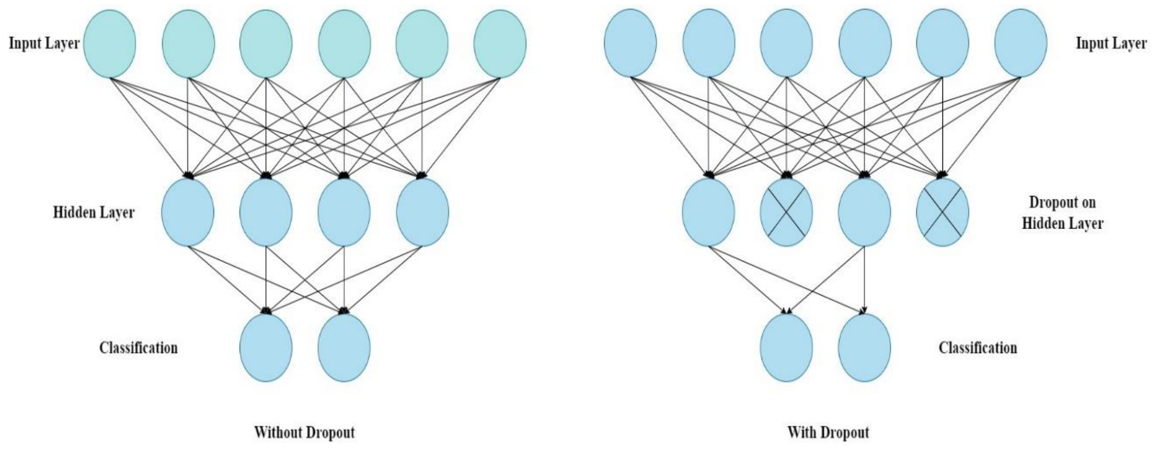


Figure 3.7.3 Dropout Layer basic operation

The dense layer is connected deeply and each neuron in this layer receives output from every neuron of its preceding layer, making them densely connected and neurons of the dense layer execute matrix-vector multiplication. A densely connected layer learns features from all the features of the previous layer. The dense layer implements matrix-vector multiplication, and the parameters can be trained and updated using backpropagation. The dense layer is used to change the dimensions of the vector as well as other operations such as rotation, scaling, and translation. In the proposed model, the dropout layers were implemented after each Dense layer.

One of the most widely used regularization approaches in deep learning is dropout. This layer acts as a mask, prohibiting some neurons from contributing to the following layer while allowing the rest to remain unconverted. The inclusion of the dropout layers reduces the deep learning model's performance because of overfitting and also reduces the model's complexity. The dropout layer's basic operation is shown in figure 3.7.3. Based on the targeted class, we then remove the top layer and replace it with a SoftMax layer (output layer). The SoftMax layer in the proposed model classifies the images into two classes. In this model, we used a (224x224) image size and ImageNet weights. Table 3.7.4 summarizes the created architecture's output shape as well as a brief model summary.

Table 3.7.4 Parameter of proposed VGG16 architecture

Layer		Output	Number of Parameters
Block 1	Conv2D-1	224, 224, 64	1,792
	Conv2D-2	224, 224, 64	36,928
	MaxPooling2D	112, 112, 64	0
Block 2	Conv2D-1	112, 112, 128	73,856
	Conv2D-2	112, 112, 128	1,47,584
	MaxPooling2D	56, 56, 128	0
Block 3	Conv2D-1	56, 56, 256	2,95,168
	Conv2D-2	56, 56, 256	5,90,080
	Conv2D-3	56, 56, 256	5,90,080
	MaxPooling2D	28, 28, 256	0
Block 4	Conv2D-1	28, 28, 512	11,80,160
	Conv2D-2	28, 28, 512	23,59,808
	Conv2D-3	28, 28, 512	23,59,808
	MaxPooling2D	14, 14, 512	0
Block 5	Conv2D-1	14, 14, 512	23,59,808
	Conv2D-2	14, 14, 512	23,59,808
	Conv2D-3	14, 14, 512	23,59,808
	MaxPooling2D	7, 7, 512	0
Block 6	Flatten	25088	0
	Dense-1	1024	2,56,91,136
	Dropout-1	1024	0
	Dense-2	512	5,24,800
	Dropout-2	512	0
	Dense-3	256	1,31,328
	Dropout-3	256	0
	Dense-4(Softmax)	2	514

We used Google Colab for coding and compiled the model by Adam optimizer and batch size was 32. The model has a total of 41,062,466 parameters, with 26,347,778 trainable and 14,714,688 non-trainable parameters.

3.8 Implementation Requirements

The proposed VGG16 was written in Keras, with Tensor Flow GPU support, and Python as the programming language. Many features of Google Collaboratory enable us to operate more quickly and efficiently. We can use it to write and execute Python code, share notebooks with a Google link, import notebooks from GitHub, and import data from Google Drive, among other things. The entire experiment, as well as the training and testing, was completed in the Google Colaboratory environment.

- Image Processing
- Convolutional Neural Network
- Google Colaboratory
- Machine Learning
- Artificial Intelligence

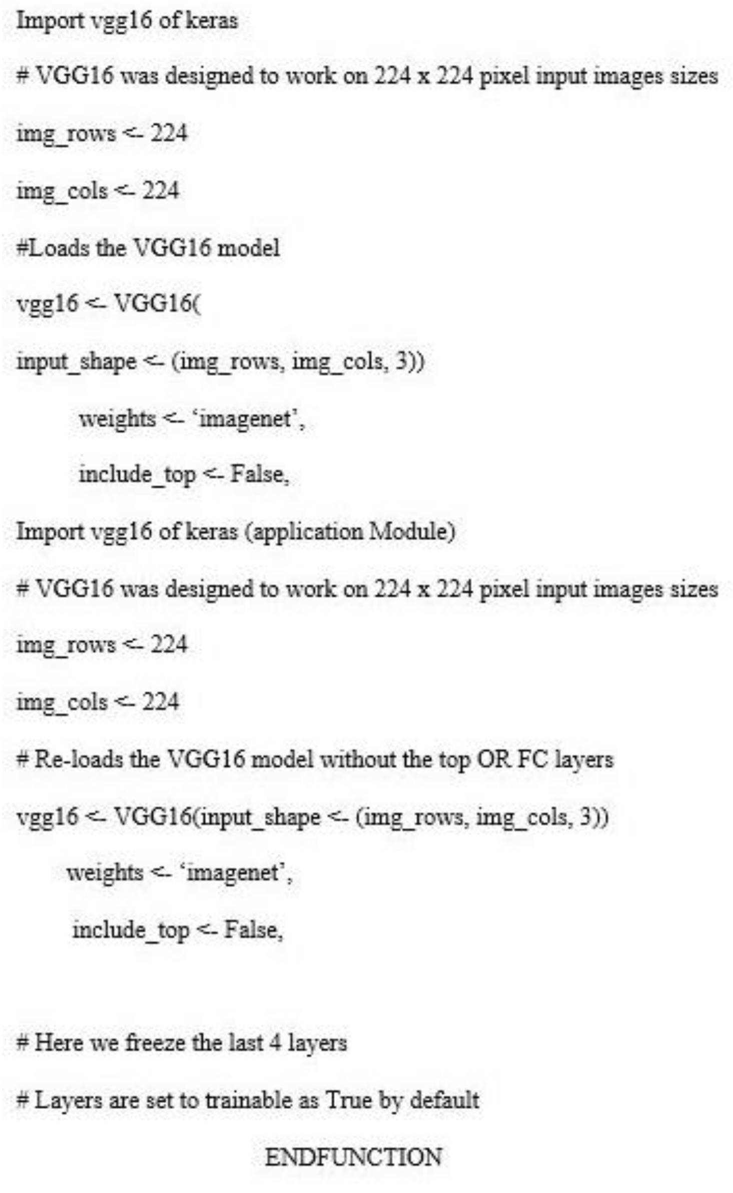
Chapter 04

Algorithm

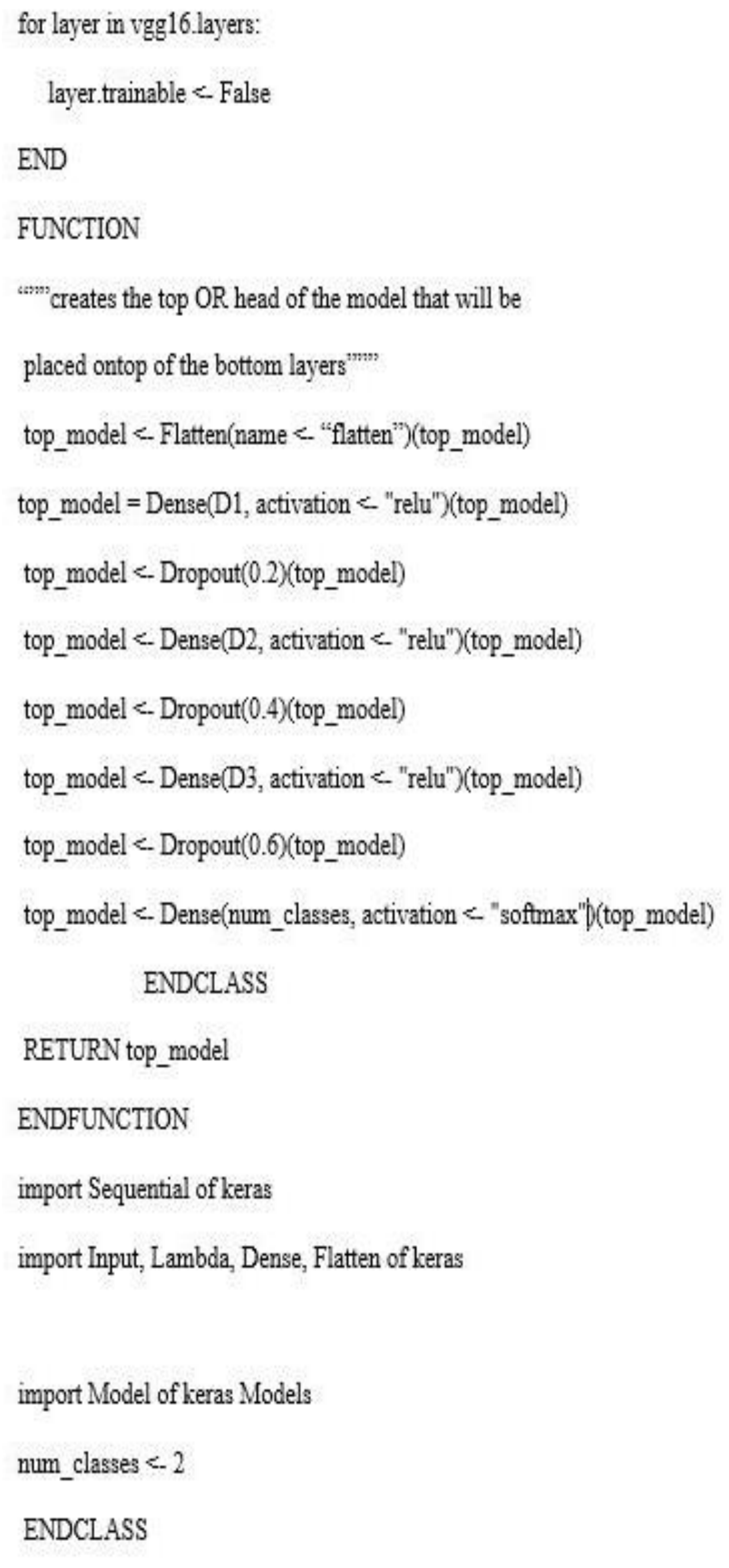
4.1 Algorithm

4.1.1 Training

In most cases, a neural network algorithms are trained in 2 steps: During this phase, the input is routed entirely along through network. In this stage, gradients is backpropagated & weights is changed. For big data sets, training the model takes longer time. To formulate & train the model, we just simply use Python library. While training VGG16 neural network model, we changed some layers.

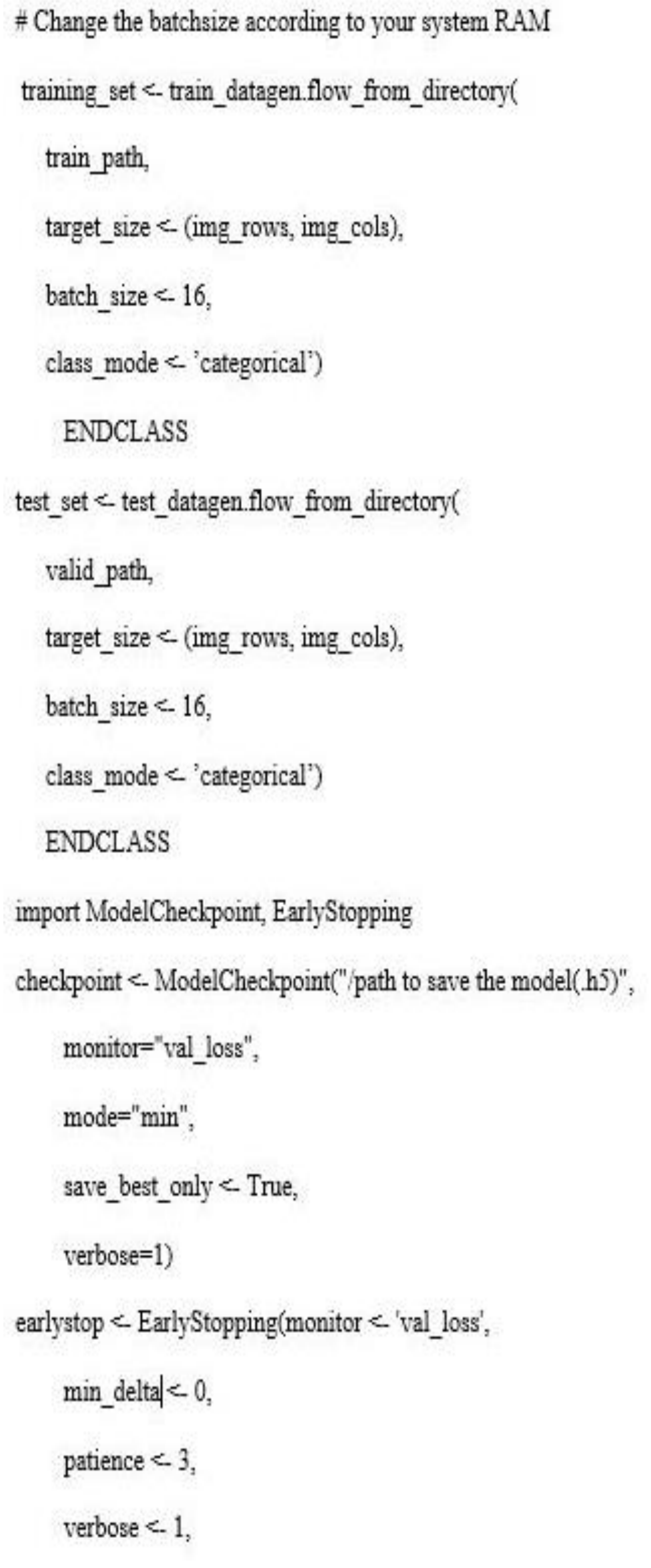


```
Import vgg16 of keras

# VGG16 was designed to work on 224 x 224 pixel input images sizes

img_rows <- 224

img_cols <- 224

#Loads the VGG16 model

vgg16 <- VGG16(

input_shape <- (img_rows, img_cols, 3))

        weights <- ‘imagenet’,

        include_top <- False,

Import vgg16 of keras (application Module)

# VGG16 was designed to work on 224 x 224 pixel input images sizes

img_rows <- 224

img_cols <- 224

# Re-loads the VGG16 model without the top OR FC layers

vgg16 <- VGG16(input_shape <- (img_rows, img_cols, 3))

       weights <- ‘imagenet’,

       include_top <- False,


# Here we freeze the last 4 layers

# Layers are set to trainable as True by default

                                ENDFUNCTION
```

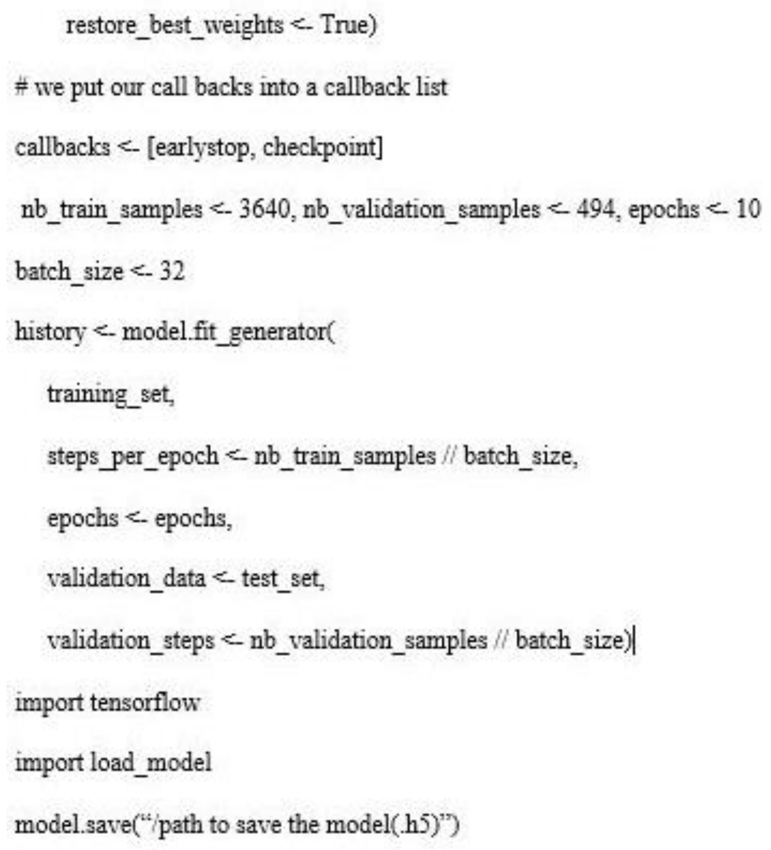


```
for layer in vgg16.layers:
    layer.trainable <- False
END
FUNCTION
“”””creates the top OR head of the model that will be
 placed ontop of the bottom layers””””
 top_model <- Flatten(name <- “flatten”)(top_model)
top_model = Dense(D1, activation <- "relu")(top_model)
 top_model <- Dropout(0.2)(top_model)
 top_model <- Dense(D2, activation <- "relu")(top_model)
 top_model <- Dropout(0.4)(top_model)
 top_model <- Dense(D3, activation <- "relu")(top_model)
 top_model <- Dropout(0.6)(top_model)
 top_model <- Dense(num_classes, activation <- "softmax")(top_model)
            ENDCLASS
 RETURN top_model
ENDFUNCTION
import Sequential of keras
import Input, Lambda, Dense, Flatten of keras


import Model of keras Models
num_classes <- 2
 ENDCLASS
```

```
prediction <- addTopModel(vgg16, num_classes)
 ENDCLASS
model <- Model(inputs=vgg16.input, outputs=FC_Head)
OUTPUT model.summary()


# Note we use a very small learning rate
model.compile(loss <- 'categorical_crossentropy',
 optimizer <- 'adam',
 metrics <- ['accuracy'])


from keras.preprocessing.image import ImageDataGenerator
train_data_dir <- '/train Folder Location'
validation_data_dir <- '/test folder Location'
train_datagen <- ImageDataGenerator(
    rescale <- 1./255,
    shear_range <- 0.2,
    rotation_range <- 20,
    width_shift_range <- 0.2,
    height_shift_range <- 0.2,
    zoom_range <- 0.2,
    horizontal_flip <- True,
    fill_mode <- 'nearest')
test_datagen <- ImageDataGenerator(rescale=1./255)
```



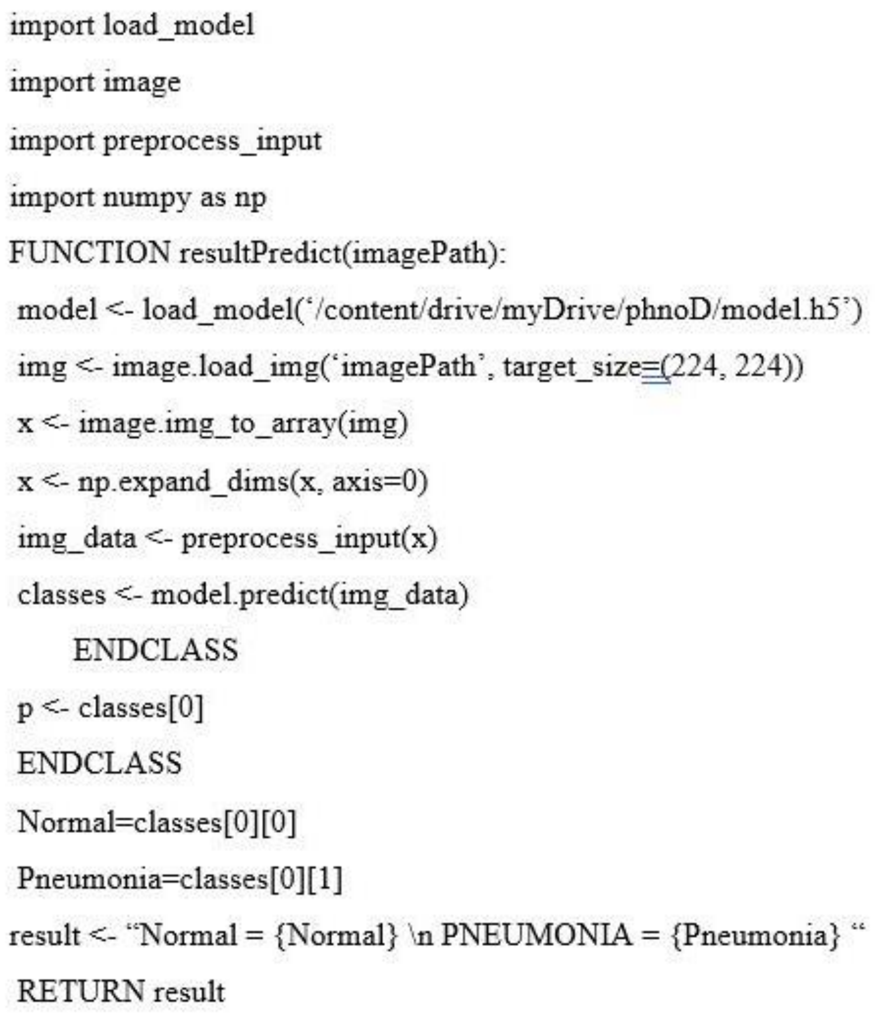
```
# Change the batchsize according to your system RAM
 training_set <- train_datagen.flow_from_directory(
    train_path,
    target_size <- (img_rows, img_cols),
    batch_size <- 16,
    class_mode <- 'categorical')
      ENDCLASS
test_set <- test_datagen.flow_from_directory(
    valid_path,
    target_size <- (img_rows, img_cols),
    batch_size <- 16,
    class_mode <- 'categorical')
    ENDCLASS
import ModelCheckpoint, EarlyStopping
checkpoint <- ModelCheckpoint("/path to save the model(.h5)",
      monitor="val_loss",
      mode="min",
      save_best_only <- True,
      verbose=1)
earlystop <- EarlyStopping(monitor <- 'val_loss',
      min_delta <- 0,
      patience <- 3,
      verbose <- 1,
```



```
       restore_best_weights <- True)
# we put our call backs into a callback list
callbacks <- [earlystop, checkpoint]
 nb_train_samples <- 3640, nb_validation_samples <- 494, epochs <- 10
batch_size <- 32
history <- model.fit_generator(
    training_set,
    steps_per_epoch <- nb_train_samples // batch_size,
    epochs <- epochs,
    validation_data <- test_set,
    validation_steps <- nb_validation_samples // batch_size)
import tensorflow
import load_model
model.save("/path to save the model(.h5)")
```

4.1.2 Predict

We tested this section at the completion of the training the model to determine if our model is capable identify pneumonia. We've passed along our trained model, along with some web pictures which were kept for validation.



```
import load_model
import image
import preprocess_input
import numpy as np
FUNCTION resultPredict(imagePath):
 model <- load_model('/content/drive/myDrive/phnoD/model.h5')
 img <- image.load_img('imagePath', target_size=(224, 224))
 x <- image.img_to_array(img)
 x <- np.expand_dims(x, axis=0)
 img_data <- preprocess_input(x)
 classes <- model.predict(img_data)
       ENDCLASS
 p <- classes[0]
 ENDCLASS
 Normal=classes[0][0]
 Pneumonia=classes[0][1]
result <- "Normal = {Normal} \n PNEUMONIA = {Pneumonia} "
 RETURN result
```

4.2 Algorithm Explanation

To begin, we'll need to import keras as well as all of the libraries we'll need to build our model.

Following that, we import VGG16 via keras using pre-trained weights from imagenet. The top parameter is being turned into False. This indicates that just pre-trained weights from convolution layers would be imported, however no dense layer weights.

Afterward, we will do not choose to train the weights of the first 19 layers & instead using as it is. As an outcome, for the very first 19 layers, the trainable parameter is turned into false. This model's last dense layer must be a 2 unit softmax dense layer as our issue is to identify Pneumonia, that has 2 classes. Then we added some additional dense and dropout layer which aids our model's training. After that, we run vggmodel.summary() on the entire VGG model, and the results are displayed below. We can see that the last 1000-unit softmax dense layer has been replaced with a 2-layer softmax dense unit when we print the model's summary. Then, from keras.preprocessing, we import ImageDataGenerator. The purpose of ImageDataGenerator is to make importing data with labels into the model as simple as possible. It's an extremely helpful class because it includes numerous functions such as resizing, rotating, and flipping. The best thing about this class is that it has no impact on disk data. As data is supplied to the model, this class modifies it. Then, in Keras, we'll import all of the Pneumonia and Normal chest X-ray pictures into the model using the ImageDataGenerator method. The data will be labeled and all of the labels will be mapped to the data using ImageDataGenerator.

Then, for both training and testing data, we created an ImageDataGenerator object, passing the picture size and folder location to the object training set and the object test set, respectively. In addition, we set the train batch size to 16 and the validation batch size to 16. This means that 16 Pneumonia photos will be passed to the network at the same time in a batch. After the model is generated, we import the ModelCheckpoint and EarlyStopping methods from keras. We create a combined object and pass it to the fit generator as callback functions.

By keeping track of a specific parameter, ModelCheckpoint assists us in saving the model. If the current epoch's validation loss is less than the previous epoch's validation loss, the model will be saved to drive. If the parameter we've set to monitor in EarlyStopping does not increase, we can use EarlyStopping to halt the model's training early. We can maintain track of validation loss in our circumstance by giving val loss to EarlyStopping. We've set patience to 3, which means that if validation loss doesn't improve after three epochs, the model will stop training.We set the loss value to 'categorical crossentropy' because we have a 2 unit dense layer at the end. The Adam optimizer was chosen, and the value 'accuracy' was added to the metrics list.

Then the data from the training and test sets will be fed into the fit generator. The model will begin to train after it has been executed, and we will begin to see training and validation accuracy and loss. The model took about 4 hours to train. We were able to obtain 92 percent accuracy on our training data and 91 percent accuracy on validation data at epoch six. We then visualize both training and validation accuracy using pyplot. We also show the loss of training and validation. The model is then saved after importing tensorflow. We load the best saved model to make predictions on the trained model.

4.3 Libraries of Python

Keras Library: In simplest form, Keras is just a Application Programming Interface(high-level) which allows you to create Deep Learning models without having to understand how the ML algorithms perform inwardly. As a result, a neural network may be defined and trained within only a few lines of code. This is implemented in Python which has a backend that employs deep learning structures like Tensorflow. Tensoflow package is used to create Deep Neural Networks and machine learning models. Keras is a NN framework that supports almost all NN models and works on both Central processing unit & Graphics processing unit.

Model: There are three approaches to develop Keras models which are subclassing of model, functional APIs, and sequential models.

- **Subclassing of models:** Model subclassing is completely customisable, allowing users to construct their own custom forward-pass. In this case, the operator must build everything from the ground up. This model is utilized in difficult situations. Deep learning practician can use the model subclassing API to obtain additional control over their training processes.

- **Sequential Model:** For most issues, the sequential API allows users to build models layer by layer. It has limitations in that it does not allow users to design models with several layers or inputs and outputs. For a simple stack of layers with one input tensor and one output tensor for each layer, a sequential approach is appropriate.
- **Functional API:** This API claims to support arbitrary model designs and is easy for using. Almost majority of the users and in the majority of use cases employ functional APIs. The layers inside sequential models which are ordered in a specified order are referred to it as the sequential API. Most Artificial Neural Networks have layers that are built in a precise order because as data is transmitted among them in a predictable order till it reach output layer.

ImageDataGenerator: Keras ImageDataGenerator api was utilized to create an augmented picture generator speedily. ImageDataGenerator creates image information in batches and augments it in actual time. By using this class users can easily enhance pictures. This class also ensures that the model is feeding with new kinds of images on every epoch. It's doesn't, however, often include processed photos towards the main corpus and only sends them. It requires smaller capacity. The flow from directory approach allows users to view image files as from directories. Fitting a model is done with the fit generator method, which takes training set, steps per epoch, epochs, callbacks, validation data, and validation steps as arguments.

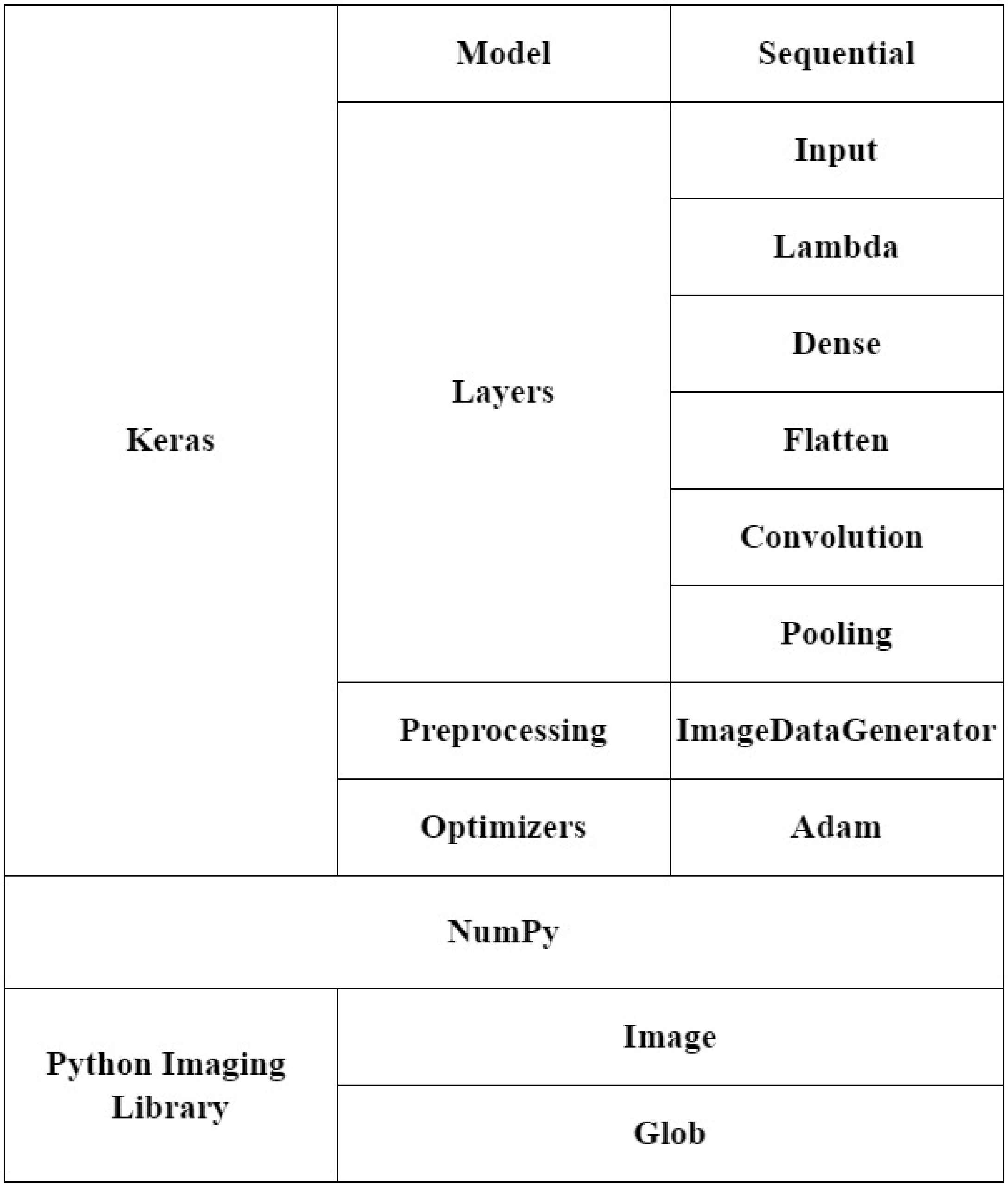
Optimizer: Optimizers are methods for minimizing error when mapping inputs to outputs or for increasing production efficiency. Optimizers are mathematical functions that are based on the learnable parameters of a model. The accuracy of the deep learning model is hugely affected by these optimization techniques. In our project, we employed Adam optimizer.

- **Adam:** Adam is a computationally efficient optimization solver for the Neural Network technique that takes little memory and is ideally suited for problems with a lot of data, parameters, or both. It is a deep learning model training algorithm that replaces stochastic gradient descent. It is highly scalable, simple to apply, and appropriate for problems involving very chaotic gradients. It's especially very well with the cases involving a bunch of data and parameters but little memory.

NumPy: Numpy is a commonly used Python library for numerical computation. It profound and complex arrays and matrices, but also a lot of high numerical methods that can be applied to them. It can restructure data contained in multidimensional arrays using Fourier Transforms. It adds robust data architectures to Python, allowing for quick arithmetic calculations with arrays and matrices. For dealing with massive volumes of data, NumPy is a handy and convenient method. It comes in handy for multiplication of matrix and data reorganization.

Glob: Glob is a Python tool that can be used to locate all of the path names that have been trying to look for files that match a particular pattern. The rules of the Unix shell are used to describe the supplied pattern for file similarity. The outcome achieved by applying those conditions for a specific pattern file similarity is handed back in different order in the program's output. We have to meet specified glob tool criteria when utilizing the file similarity pattern since the glob tool can browse in via a series of files at several point on our local disk. This tool may simply doing the process without accessing the system's output in a various sub-shell. Table: 4.3.1 summarizes the libraries of Python that we have used in this project.

Table: 4.3.1 Libraries



Keras	Model	Sequential
	Layers	Input
		Lambda
		Dense
		Flatten
		Convolution
		Pooling
	Preprocessing	ImageDataGenerator
	Optimizers	Adam
NumPy		
Python Imaging Library	Image	
	Glob	

Chapter 05

Results & Discussion

5.1 Model Performance

The results of the transfer learning model for the chest X-Ray are presented in this section. There are different ways of evaluate the results but we are evaluating the results by analyzing training accuracy and loss, validation accuracy and loss parameters. In addition, numerous tests on the chest X-Ray dataset are performed to test the efficiency of this proposed model. We set twenty epochs in the proposed model, but it stopped after thirteen epochs because validation loss did not improve. The model's training and validation performance for each epoch is shown in Table 5.1.1.

Table: 5.1.1 Training model Performance

Number of Epochs	**Training Accuracy**	**Training Loss**	**Validation Accuracy**	**Validation Loss**
1	0.8850	0.2767	0.8980	0.2437
2	0.9329	0.1753	0.9260	0.1839
3	0.9427	0.1536	0.9128	0.2231
4	0.9513	0.1262	0.9441	0.1634
5	0.9521	0.1288	0.9243	0.2159
6	0.9563	0.1176	0.9490	0.1374
7	0.9608	0.1135	0.9441	0.1473
8	0.9619	0.1088	0.9572	0.1349
9	0.9598	0.1074	0.9457	0.1493
10	0.9610	0.1042	0.9457	0.1370
11	0.9633	0.0972	0.9309	0.2187
12	0.9604	0.1043	0.9391	0.1607
13	0.9640	0.0919	0.9359	0.1566

Table 5.1.1 shows that the suggested model starts running epoch after epoch after training all images from the training dataset. At epoch 1, the training accuracy was 0.8850, the training loss was 0.2767, and the validation accuracy and loss were 0.8980 and 0.2437, respectively. Then after epochs the model trained well and also overcome the underfitting problem. At epoch eight, we achieved the highest validation accuracy of 0.9572, as well as the lowest validation loss of 0.1349.

5.2 VGG16 Model Training & Validation Accuracy vs Loss

Figures 5.2.1 and 5.2.2 demonstrate the progress of the training and validation accuracy and loss curves over time. The train curve is derived from the training dataset and indicates how well the model is learning, whereas the validation curve, also known as the test curve, is derived from a hold-out validation dataset and indicates how well the model is generalizing.

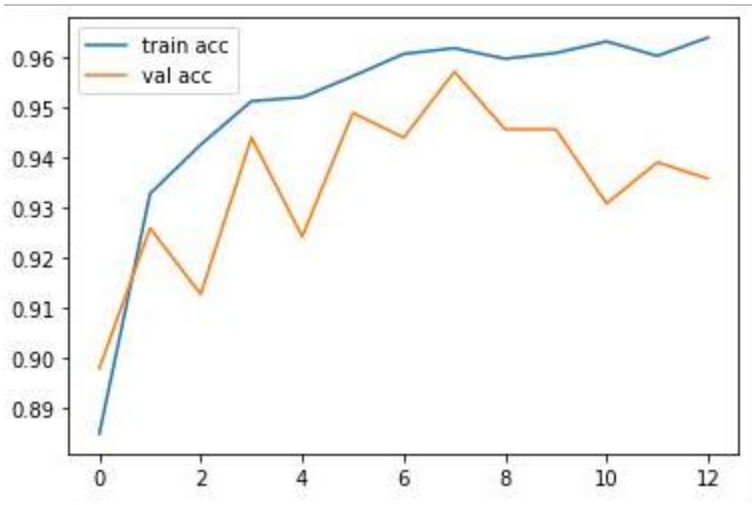


Figure 5.2.1 Training Accuracy vs Validation Accuracy

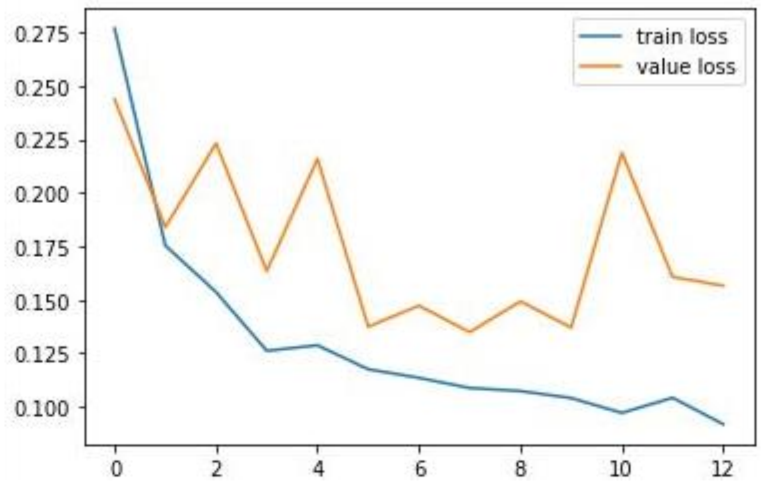
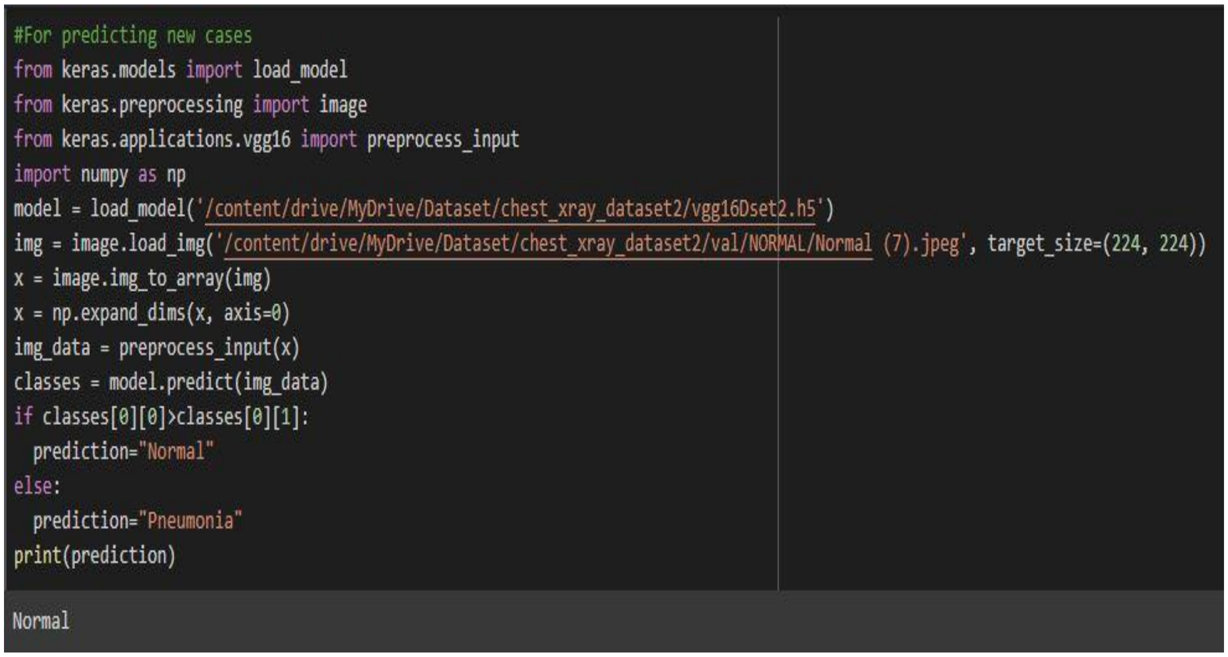


Figure 5.2.2 Training Loss vs Validation Loss

As can be seen from the graphs, both training and validation accuracy improved over time. The number of epochs is represented on the x-axis, and the loss or accuracy is represented on the y-axis. The training and validation accuracies were both low, and the loss was high, when the epoch was one. However, when the number of epochs increased, both training accuracy and validation accuracy increased, and training and validation loss decreased. At epoch eight, the training and validation accuracies was both above 95 percent. The training loss was 10 percent and validation loss 13 percent. After eight epochs training accuracy increased but validation loss did not improved. We used early stopping class of keras and set patience value 5. For this reason, when 5 epochs with no improvement of validation loss the training will be stopped early after completing thirteen epochs.

5.3 Predicting Pneumonia

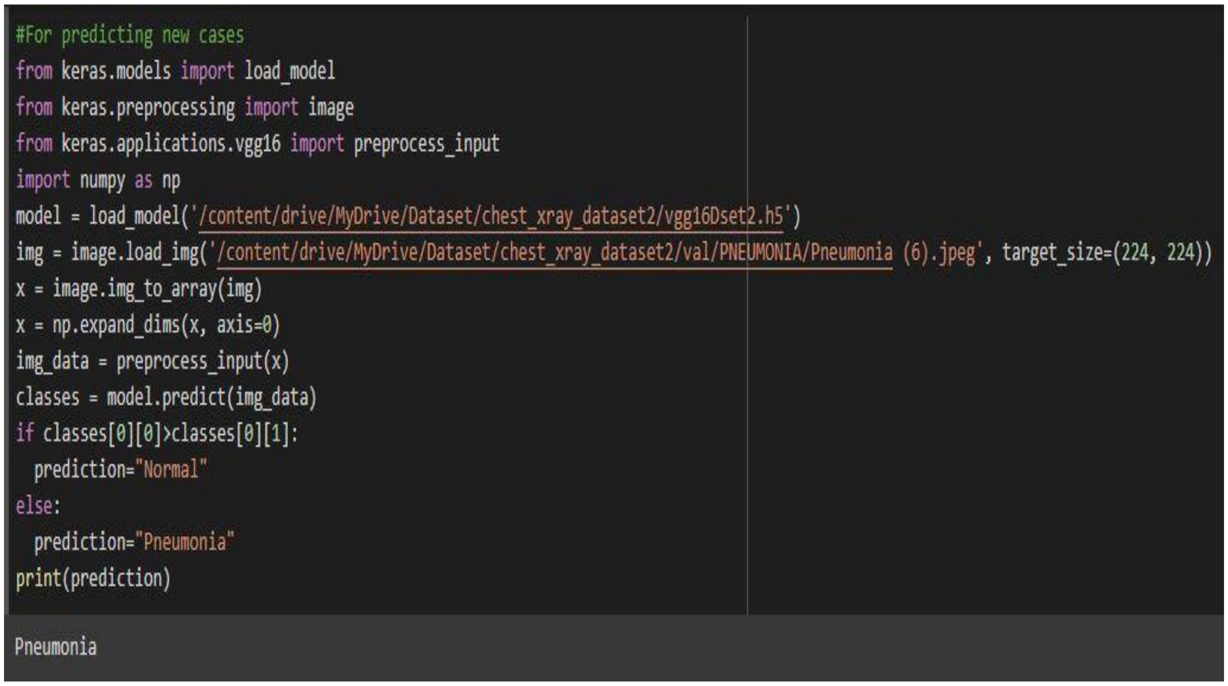
To verify if our model could correctly detect Pneumonia, we tested our model by validation images and other images from the Internet. Figure 5.3.1, 5.3.2, 5.3.3 shows that the proposed model is able to accurately predict Pneumonia.



```
#For predicting new cases
from keras.models import load_model
from keras.preprocessing import image
from keras.applications.vgg16 import preprocess_input
import numpy as np
model = load_model('/content/drive/MyDrive/Dataset/chest_xray_dataset2/vgg16Dset2.h5')
img = image.load_img('/content/drive/MyDrive/Dataset/chest_xray_dataset2/val/NORMAL/Normal (7).jpeg', target_size=(224, 224))
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
img_data = preprocess_input(x)
classes = model.predict(img_data)
if classes[0][0]>classes[0][1]:
  prediction="Normal"
else:
  prediction="Pneumonia"
print(prediction)
```

```
Normal
```

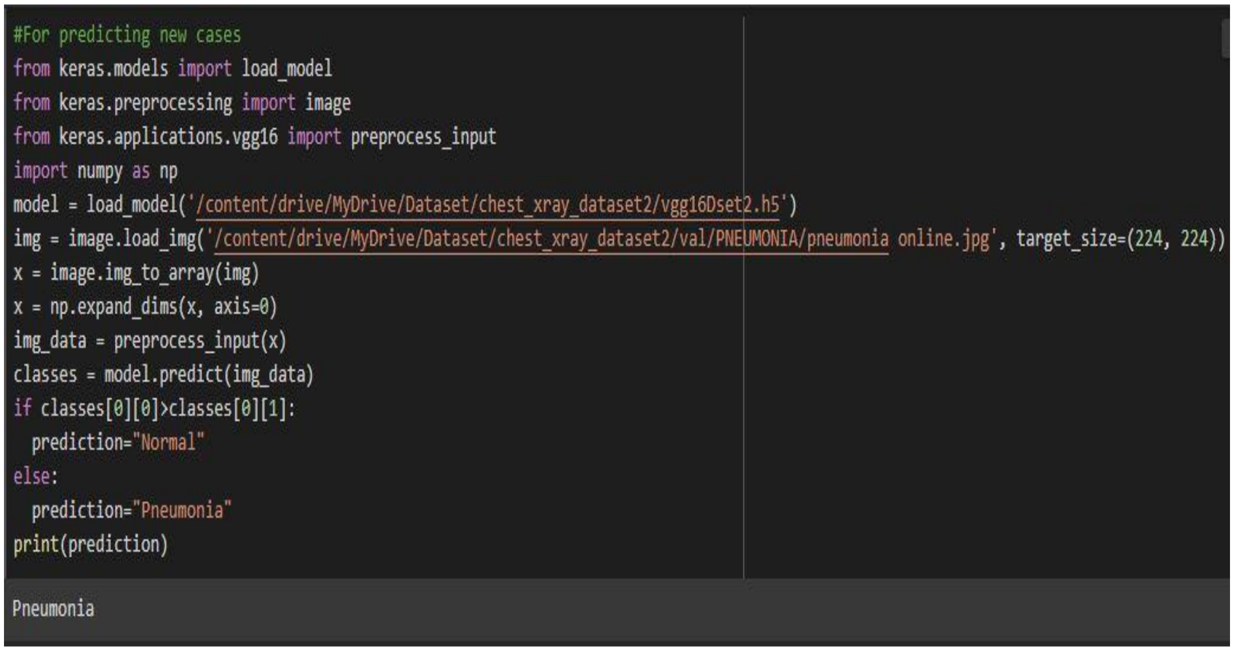
Figure: 5.3.1 Normal Test



```
#For predicting new cases
from keras.models import load_model
from keras.preprocessing import image
from keras.applications.vgg16 import preprocess_input
import numpy as np
model = load_model('/content/drive/MyDrive/Dataset/chest_xray_dataset2/vgg16Dset2.h5')
img = image.load_img('/content/drive/MyDrive/Dataset/chest_xray_dataset2/val/PNEUMONIA/Pneumonia (6).jpeg', target_size=(224, 224))
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
img_data = preprocess_input(x)
classes = model.predict(img_data)
if classes[0][0]>classes[0][1]:
  prediction="Normal"
else:
  prediction="Pneumonia"
print(prediction)
```

```
Pneumonia
```

Figure: 5.3.2 Pneumonia Test



```
#For predicting new cases
from keras.models import load_model
from keras.preprocessing import image
from keras.applications.vgg16 import preprocess_input
import numpy as np
model = load_model('/content/drive/MyDrive/Dataset/chest_xray_dataset2/vgg16Dset2.h5')
img = image.load_img('/content/drive/MyDrive/Dataset/chest_xray_dataset2/val/PNEUMONIA/pneumonia online.jpg', target_size=(224, 224))
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
img_data = preprocess_input(x)
classes = model.predict(img_data)
if classes[0][0]>classes[0][1]:
  prediction="Normal"
else:
  prediction="Pneumonia"
print(prediction)
```

```
Pneumonia
```

Figure: 5.3.3 Online Data Test

Figure 5.3.1 shows that, we tested the proposed model with an chest X-ray of a Normal patient and figure 5.3.2 shows that, we tested the proposed model with an chest X-ray of a Pneumona patient from the dataset's validation folder. Figure 5.3.3 shows that, we tested the proposed model with chest X-ray of a patient which was collected from internet. In each case, the proposed model is able to successfully predict the outcome.

CHAPTER 6

6.1 Conclusion

The detection of normal and pneumonia classes was the main emphasis of this work. In this paper, it is proposed to use deep learning algorithms to automatically determine pneumonia using chest X-ray images. For different image classification problems, there are many types of algorithms in Deep Learning but CNNs are the most popular and widely used. Deep learning models, on the other hand, work best when they've been trained on thousands of images. Large amounts of labeled data are difficult to obtain since image classification requires experienced doctors, which is a costly and time-consuming process.

Overfitting can occur when deep neural networks are trained with limited data. A challenging issue in analyzing the proposed model is the lack of large datasets. The proposed methodology's performance would only improve as more data are available. Because the training data we had was insufficient, we used data augmentation techniques to reduce overfitting.

6.2 Future Works

This research will be expanded in the future to detect bacterial and viral pneumonia. We'll also build a web application to automate the procedure, saving both time and money. Doctors can use this tool to review the chest x-ray to see if the conclusion they made was correct. It can only assist doctors in their decision-making, but only an expert can make the final decision. Patients can also check whether they have pneumonia by uploading an x-ray image into the application. They only need to comply to the doctor's recommendations and after the application diagnoses pneumonia, take proper medications. This approach will assist in lowering the cost of pneumonia treatment, allowing underprivileged people to receive proper care at a lower cost. It will relieve some of the load on the doctors. Doctors will be able to treat more patients for less money, and everyone will be benefitted from this system.

REFERENCES

[1] Hansen, L.S., Lykkegaard, J., Thomsen, J.L. and Hansen, M.P., 2020. Acute lower respiratory tract infections: Symptoms, findings and management in Danish general practice. European Journal of General Practice, 26(1), pp.14-20.

[2] Hashmi, M.F., Katiyar, S., Keskar, A.G., Bokde, N.D. and Geem, Z.W., 2020. Efficient pneumonia detection in chest xray images using deep transfer learning. Diagnostics, 10(6), p.417.

[3] Cherian, T.; Mulholland, E.K.; Carlin, J.B.; Ostensen, H.; Amin, R.; Campo, M.D.; Greenberg, D.; Lagos, R.;
Lucero, M.; Madhi, S.A.; et al. Standardized interpretation of paediatric chest radiographs for the diagnosis
of pneumonia in epidemiological studies. Bull. World Health Organ. 2005, 83, 353–359.

[4] Hosny, A., Parmar, C., Quackenbush, J. and Schwartz, L.H., 2018. Aerts HJWL. Artificial intelligence in radiology. Nat Rev Cancer, 18(8), pp.500-510.

[5] Rajpurkar, P., Irvin, J., Zhu, K., Yang, B., Mehta, H., Duan, T., Ding, D., Bagul, A., Langlotz, C., Shpanskaya, K. and Lungren, M.P., 2017. Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning. arXiv preprint arXiv:1711.05225.

[6] Mohanty, S.P., Hughes, D.P. and Salathé, M., 2016. Using deep learning for image-based plant disease detection. Frontiers in plant science, 7, p.1419.

[7] Cai, L., Gao, J. and Zhao, D., 2020. A review of the application of deep learning in medical image classification and segmentation. Annals of translational medicine, 8(11).

[8] Grewal, M., Srivastava, M.M., Kumar, P. and Varadarajan, S., 2018, April. Radnet: Radiologist level accuracy using deep learning for hemorrhage detection in ct scans. In 2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018) (pp. 281-284). IEEE.

[9] Jeyaraj, P.R. and Nadar, E.R.S., 2019. Computer-assisted medical image classification for early diagnosis of oral cancer employing deep learning algorithm. Journal of cancer research and clinical oncology, 145(4), pp.829-837.

[10] Baccouche, A., Garcia-Zapirain, B., Castillo Olea, C. and Elmaghraby, A., 2020. Ensemble deep learning models for heart disease classification: A case study from Mexico. Information, 11(4), p.207.

[11] Ayan, E. and Ünver, H.M., 2019, April. Diagnosis of Pneumonia from chest X-ray images using deep learning. In 2019 Scientific Meeting on Electrical-Electronics & Biomedical Engineering and Computer Science (EBBT) (pp. 1-5). Ieee.

[12] Lin, T.Y., Goyal, P., Girshick, R., He, K. and Dollár, P., 2017. Focal loss for dense object detection. In Proceedings of the IEEE international conference on computer vision (pp. 2980-2988).

[13] Rajasenbagam, T., Jeyanthi, S. and Pandian, J.A., 2021. Detection of pneumonia infection in lungs from chest X-ray images using deep convolutional neural network and content-based image retrieval techniques. Journal of Ambient Intelligence and Humanized Computing, pp.1-8.

[14] El Asnaoui, K., Chawki, Y. and Idri, A., 2021. Automated methods for detection and classification pneumonia based on x-ray images using deep learning. In Artificial intelligence and blockchain for future cybersecurity applications (pp. 257-284). Springer, Cham.

[15] Verma, G. and Prakash, S., 2020. Pneumonia classification using deep learning in healthcare. International Journal of Innovative Technology and Exploring Engineering (IJITEE), 9(4), pp.1715-1723.

[16] Kermany, D.S., Goldbaum, M., Cai, W., Valentim, C.C., Liang, H., Baxter, S.L., McKeown, A., Yang, G., Wu, X., Yan, F. and Dong, J., 2018. Identifying medical diagnoses and treatable diseases by image-based deep learning. Cell, 172(5), pp.1122-1131.

[17] Ibrahim, A.U., Ozsoz, M., Serte, S., Al-Turjman, F. and Yakoi, P.S., 2021. Pneumonia classification using deep learning from chest X-ray images during COVID-19. Cognitive Computation, pp.1-13.
[18] Liang, G. and Zheng, L., 2020. A transfer learning method with deep residual network for pediatric pneumonia diagnosis. Computer methods and programs in biomedicine, 187, p.104964.
[19] Hasan, M.D., Ahmed, S., Abdullah, Z.M., Monirujjaman Khan, M., Anand, D., Singh, A., AlZain, M. and Masud, M., 2021. Deep learning approaches for detecting pneumonia in COVID-19 patients by analyzing chest X-ray images. Mathematical Problems in Engineering, 2021.
[20] Varshni, D., Thakral, K., Agarwal, L., Nijhawan, R., & Mittal, A. (2019, February). Pneumonia Detection Using CNN based Feature Extraction. In 2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT) (pp. 1-7). IEEE.
[21] Jain, R., Nagrath, P., Kataria, G., Kaushik, V.S. and Hemanth, D.J., 2020. Pneumonia detection in chest X-ray images using convolutional neural networks and transfer learning. Measurement, 165, p.108046.
[22] Garstka, J. and Strzelecki, M., 2020, September. Pneumonia detection in x-ray chest images based on convolutional neural networks and data augmentation methods. In 2020 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA) (pp. 18-23). IEEE.
[23] Tammina, S., 2019. Transfer learning using vgg-16 with deep convolutional neural network for classifying images. International Journal of Scientific and Research Publications (IJSRP), 9(10), pp.143-150.
[24] Simonyan, K. and Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.