BACHELOR OF SCIENCE IN ELECTRONIC AND TELECOMMUNICATION ENGINEERING

An Improved Novel Corona virus (COVID-19) Detection Model Employing Convolutional Neural Networks Based on Lung X-Ray Images

Submitted by

Ahmed Jobayer Chowdhury

T181041

Supervised by

Muhammad Mostafa Amir Faisal

Assistant Professor

Department of ETE, IIUC

Department of Electronic and Telecommunication Engineering

International Islamic University Chittagong

Kumira, Sitakunda, Chittagong – 4314

November-2022

CANDIDATE DECLARATION

It is to hereby declare that the work presented in herein is genuine work done by us and has not concurrently submitted in candidature for any degree. The result of this thesis that we have found totally dependent of our own work.

This thesis was done under the guidance of Muhammad Mostafa Amir Faisal, Assistant Professor of Electronic and Telecommunication Engineering, International Islamic University Chittagong.

…...……

Ahmed Jobayer Chowdhury

ID- T181041

RECOMMENDATION

This is to certify that**,** **Ahmed Jobyer Chowdhury** bearing Metric ID. **T-181041** is the student of International Islamic University Chittagong (IIUC) under the Department of Electronic and Telecommunications Engineering has carried out the thesis on titled **"An Improved Novel Corona virus (COVID-19) Detection Model Employing Convolutional Neural Networks Based on Lung X-Ray Images"** successfully under my supervision and guidance and has been accepted as satisfactory in partial fulfillment of the requirement.

Muhammad Mostafa Amir Faisal
Assistant Professor
Dept. of Electronic and Telecommunication Engineering
International Islamic University Chittagon

ACKNOWLEDGMENT

First of all, we would like to express our gratitude to Almighty Allah (SWT) to enabling us to complete this report on **"An Improved Novel Corona virus (COVID-19) Detection Model Employing Convolutional Neural Networks Based on Lung X-Ray Images"**.

Any kind of thesis needs assistance from a lot of people to be completed successfully. To prepare this thesis paper, we also enlisted the aid of several other people. There is now a small effort being made to express our sincere gratitude to that helpful person.

We convey our sincere gratitude to our honorable Academic Supervisor **Muhammad Mostafa Amir Faisal,** Assistant Professor of Department of Electronic and Telecommunications Engineering, International Islamic University ChittagongThis thesis would not have been as successful without his patient coaching and insightful advice. His oversight and direction shaped this thesis to be completed flawlessly at every stage.This thesis would not have been as successful without his patient coaching and insightful advice. His oversight and direction shaped this thesis to be completed flawlessly at every stage.

Last but not least, we would like to express our sincere gratitude to all of our teachers for their endless support and assistance. We are incredibly lucky to have kind people like them by our side.

DECLARATION

The study in this thesis, submitted for the degree of Bachelor of Science was undertaken at the Department of Electronic and Telecommunication Engineering, International Islamic University Chittagong, under the supervision of Muhammad Mostafa Amir Faisal , Assistant Professor,

Department of ETE, IIUC.

Information quoted from any source has been properly acknowledged by mentioning the author's reference at appropriate place. This thesis work is submitted for partial fulfillment of the requirements of the degree of Bachelor of Science in Electronic and Telecommunication Engineering.

ABSTRACT

The first cases of COVID-19 were discovered in December of 2019. Extremely contagious with no cure in sight, the only option was to seek out and quarantine anyone who had contracted it. As a result, scientists are beginning to use chest X-rays (CXRs), a simple and cheap diagnostic tool. Because of a shortage in diagnostic tools, scientists developed an automated systems to combine chest X-rays for a more accurate diagnosis. The purpose of this study is to employ artificial intelligence to train CNN models using raw chest X-rays and CT scans. Binary or multi-class classifications and transfer learning are used. Various datasets' characteristics are offered, and they are used for training and verifying models. Algorithm performance is measured in terms of accuracy, precision, recall, and F1 score. As compared to conventional methods like polymerase chain reaction (PCR) testing, deep learning provides more accurate predictions of patient severity. Automatic CXR detection tools help radiologists screen for and make an accurate diagnosis of infectious disease in patients. They're widely used because of their low cost, accessibility, and speedy outcomes. With a 99.2% detection rate, 97.5 % accuracy, and 98 % F1-score, the proposed method shows potential for detecting people infected with COVID-19.

Table of Contents

List of Figures

LIST OF TABLE

Chapter 1
Introduction

1 Introduction

The origin of the name coronavirus is Greek (κορώνη), which translates to "crown" or "halo" and relates to the virus's appearance under an electron microscope, which resembles a royal crown. Therefore, coronavirus is also known as the crowned virus. In December 2019, the COVID-19 developed as an epidemic illness in Wuhan, China. Today, this has evolved into a pandemic as a hazardous global public health crisis. The COVID-19 virus is also known as the SARS-COV-2 virus. This virus is a large-family virus that is subdivided into four subtypes: alfa-coronavirus, bita-coronavirus, gamma-coronavirus, and lamda-coronavirus. To present, seven of the forty coronavirus species have been identified as being transmitted to people through common illnesses such as the common cold. Previous research has shown that some viruses, like SARS and MERS, may be transmitted from cats or camels to people. It is believed that the COVID-19 has been transferred to humans for the first time via anteaters and bats.[1] This virus is characterized by dry cough, fever, and shortness of breath. Additionally, muscular soreness, sputum secretion, and sore throats are moderate COVID-19 symptoms. In severe situations, the virus may result in pneumonia, acute respiratory problems, septic shock, multiorgan failure, and death. The virus is mostly transmitted by the coughing droplets of the carrier. The virus takes between 2 and 14 days to grow. This virus's structure comprises of outer and inner layers. COVID-19's internal structure comprises the virus's nucleus, which carries genetic material. The virus's outermost coat is composed of protein crowns. Upon entering the host cell, the viral genome enters the cytoplasm. According to research, males have a greater incidence of coronavirus than women. Additionally, children are less prone than adults to get the virus. Children are projected to have a COVID-19 death rate between 1% and 5% . Recent hospitalization guidelines released by the World Health Organization (WHO) stipulate that the essence of coronavirus must be confirmed by Reverse Transcription Polymerase Chain Reaction (RT-PCR) or a gene sequence as the primary indicator for respiratory or blood specimen symptoms .[2] However, due to sample collection, transportation constraints, and kit findings, the original presentation indicated that the positive RT-PCR rate for throat swab samples was between 30 and 60%. In addition, the lengthy diagnostic period based

on RT-PCR kits leads many individuals to be misdiagnosed as having COVID-19 and to not get adequate therapy. Because the RT-PCR kits for the diagnosis of COVID-19 are restricted. Therefore, the patient may die owing to a lack of hospitalization-required therapy, the extremely infectious nature of the virus, the poor sensitivity of the RT-PCR Kits, and the length of time required to determine the conclusion of the diagnosis. In addition, due to the highly contagious nature of this virus, the patient poses a danger of infecting further individuals.[3]

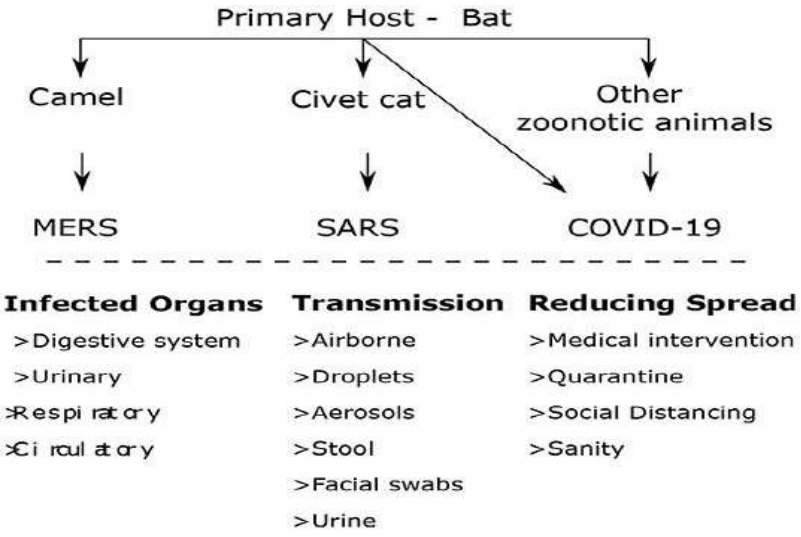


FIGURE 1.1. Transmission of the SARS-COV

The transmission of the severe acute respiratory syndrome coronavirus is shown in Figure 1. (SARS-CoV). The four genera of SARS-CoV-2 strains are alpha, beta, gamma, and delta. However, alpha and beta are derived from bats, and gamma and delta are descended from birds and the peccary. The carriers of SARS and MERS are civet cats and camels, respectively. Along with other major symptoms such as fever, cold, bodily discomfort, sore throat, and epistaxis, Pneumonia is the most prevalent sign of infection with the COVID-19 terrible illness.[2]

The overall number of actual cases in the first wave through June 2020 is 8,708,008, while the estimated number of fatalities is 461,716. (WHO Statistics). However, the severity of COVID-19 indicates the total number of cases in the second wave until June 1, 2021. As a consequence, the total number of confirmed cases is 171,782,908, while the estimated cumulative death toll is 3,698,621. According to the WHO, between January 3, 2020 and October 1, 2021, there were 33,766,707 confirmed cases of COVID-19 in India,

with 448,331 deaths. On September 27, 2021. 870,708,636 doses of vaccination had been distributed. Table 1 displays the global COVID-19 cases and deaths by area, in absolute numbers, as of September 27, 2021. These territories have been modified in accordance with the World Bank. Figure 2 summarizes the global COVID-19 cases and deaths reported by age and gender between 30 December 2019 and 27 September 2021.[1]

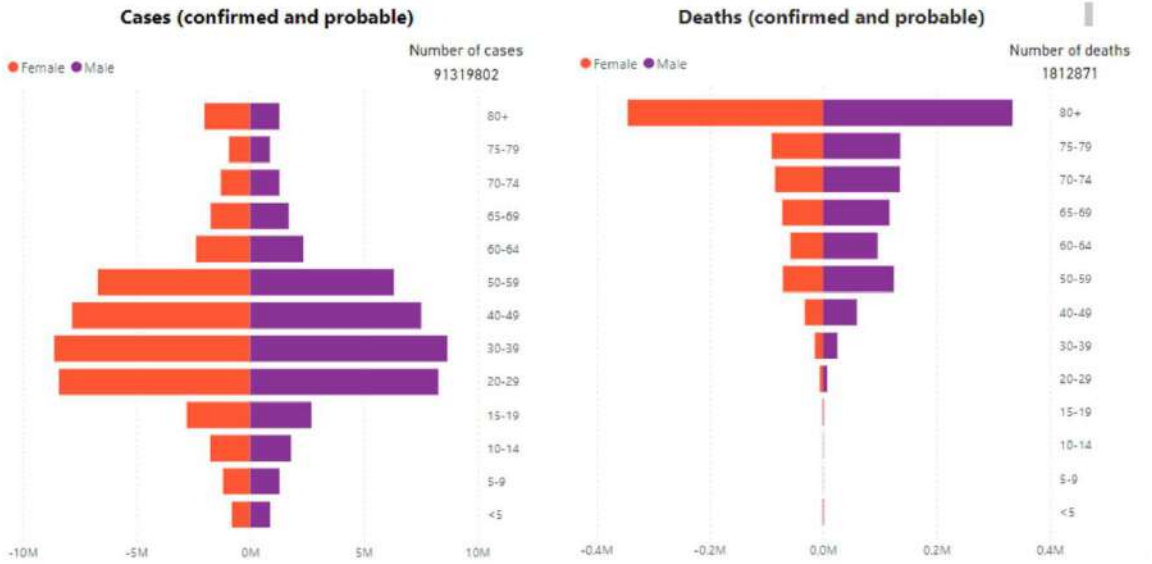


FIGURE 1.2 COVID-19 cases and fatalities were reported by age and gender

The COVID-19 situation has caused social, economic, and health disruptions. Lockdowns have caused individuals to lose their jobs. These factors have exerted enormous strain on people's mental health and had negative bodily effects. Working from home has helped the sector thrive due to the availability of internet jobs. However, the worst-affected sectors entail physical labor such as labor, technicians, etc. The lockdowns have a significant impact on low-income, family-like laborers, such as street workers. The lockdown has significant detrimental effects, especially on the economy. Therefore, a prompt COVID-19 diagnosis is vital for the management and treatment of this condition[4]. In addition, nurses are less likely to be infected with the virus if a chest scan is performed as opposed to RT-PCR kits. Computerized Tomography (CT) and X-ray scans are the most used procedures for imaging the chest. X-rays are used to detect pneumonia, lung infections, fractures, and malignancies inside the body. CT scan is a specialized X-ray device that provides crisper pictures of bones, tissues, and organs. The X-ray approach is simpler, faster, and more cost-effective than the CT method, but it is more hazardous. Doctors may visually identify Viral, Bacterial, and COVID-19 infections,

among others, using chest X-rays. Typically, visual diagnosis is an unpleasant, time-consuming, and inaccurate approach. Because it requires specialized human resources and may lead to inaccuracy. In addition, this form of imaging has some flaws, such as overlapping organs, blurring boundaries, and reduced contrast, which may not lead to an accurate diagnosis of pneumonia. Consequently, based on these facts, the automated identification of virus kinds, including corona virus, on the basis of chest imaging has gained increased attention in recent years. The speedy diagnosis made possible by the automated analysis of the COVID-19 reduces health care staff effort. This study is highly useful for avoiding death and treating patients promptly.Deep learning has altered the way we use and understand data and artificial intelligence. Deep learning derives its name from the networks' several layers and vast number of trainable parameters. In this research, we have used the well-known neural network CNN (Convolution neural network). It is mostly used to handle information that has a complete grid structure, such as a picture. Images are composed of pixels organized in a lattice structure, with each grid having a unique value that represents the hue and brightness of that pixel. As each neuron functions in its receptive area, the human brain is able to extract information from pictures. The brain's neuronal network encompasses the full visual field. CNNs may similarly handle data inside their receptive fields.

1.1 Motivation

To effectively tackle any danger, it is necessary to first comprehend it. The global population has been preoccupied with the corona-virus epidemic. Concerns about the virus and stay-at-home orders have prevented some patients from undergoing normal examinations, screening tests, diagnostic procedures, and other medical treatment.[5]
However, delaying medical care might have major consequences. Numerous physicians and hospitals around the nation have seen a significant reduction in patients seeking anything from routine exams to cancer screenings such as mammograms, colonoscopies, and even pediatric care. Due of the epidemic, a large number of Americans have forgone or delayed medical treatment. Due to the precautions taken, discovery is delayed.

During the Covid-19 pandemic, testing for SARS-CoV-2 to enable early identification and isolation of potentially infectious individuals was a cornerstone of public health efforts to prevent viral transmission. Early reliance on reverse transcription polymerase chain reaction (RT-PCR) to confirm infection in those exhibiting traditional Covid-19

signs and symptoms was quickly supplemented by the use of rapid antigen detection tests, particularly as the use case scenarios for testing shifted from containment to a situation in which testing plays a much larger role in efforts to ease societal restrictions.[6] The need for specialized laboratory facilities and the worldwide demand for reagents impeded the ability of nations to rapidly scale up RT-PCR testing. In contrast, screening Covid-19 using CXRs is less costly, faster, and more accurate. In addition, researchers have discussed the societal effects of Covid-19 by taking the following factors into account: open public health issues and common AI solutions (with multiple case studies, such as TB and SARS: COVID-19), AI in sustainable health care, AI in precision medicine, and data privacy concerns. Public health merits particular attention since it drives the economy and the educational system. Covid chest X-ray detection is quicker than RT-PCR (Reverse Transcription polymerase chain reaction). As we all know, PCR tests for identifying Corona Virus are costly in our country, however this suggested approach is cost-effective. The PCR test is time-consuming, but the suggested method can identify Coronavirus faster. This innovation will transform healthcare. This technique may rapidly identify COVID-19 patients, so aiding in the fight against the current pandemic. Numerous deep learning algorithms may be used to build a robust model that may benefit mankind. This invention will inspire future generations to find a solution to this problem. This proposed strategy could be useful in the medical field . To respond to the COVID-19 pandemic, researchers will use the proposed approach to continue developing superior deep learning technologies for computer-aided design systems.[7]

1.2 Deep Features- Implementation

Deep Learning is a branch of machine learning within Artificial Intelligence (AI) that use algorithms inspired by the biological structure and function of the brain to aid computers in acquiring intelligence. While machine learning is effective for a broad variety of problems, it is ineffective in other areas where people shine.

As an example, classifying a picture as a cat or a dog, or recognizing audio recordings as male or female voices, etc. Image, audio, and other forms of unstructured input are difficult for machine learning to process. An epiphany led to the idea of recreating the organic process of learning new information in the human brain, which is composed of billions of interconnected and ordered neurons. In contrast, neural networks had been the

focus of study for a number of years but had made little progress owing to computational and data constraints. When researchers coupled machine learning with neural networks, they created the field of deep learning, which is characterized by the creation of deep neural networks, which are neural networks with many more layers that were created by improvisation.

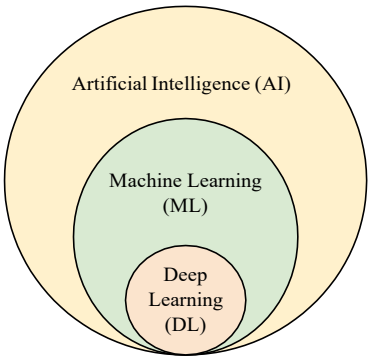


Figure 1.3: Basic view of Deep Neural Networks

1.3 Perceptron

In 1957, Frank Rosenblatt invented Perceptron. He designed a Perceptron learning rule based on the original MCP neuron. A Perceptron is a method for binary classifiers that use supervised learning. This method allows neurons to learn and understand specific training set components.[8]

Perceptrons are divided into two categories: single layer and multilayer.

- Singlelayer - perceptrons can only learn patterns that are linearly separable.
- Multilayer - With two or more layers, multilayer perceptrons or feed-forward neural networks have increased processing power.

To construct a linear decision boundary, the Perceptron algorithm learns the input signal weights.

The significance of the perceptron rule in neural networks is discussed below:

- A Perceptron method is required for supervised binary classification learning.
- Another advantage of perceptron is that this method allows neurons to learn and processes each training set element individually.

- Single-layer perceptrons can only learn linearly separable patterns, allowing them to be used in less sophisticated applications.
- The processing capacity of multilayer perceptrons and feed-forward neural networks with two or more layers is increased, allowing them to be used in complicated algorithms.
- Another significance of perceptron is that the Perceptron algorithm learns the weights for each input signal in order to establish a linear decision boundary.

Example of perceptron rule in neural network are given below:

- A perceptron is a sort of neural network unit that calculates to identify characteristics in input data. It is a function whose input x is multiplied by the learnt weight coefficient to produce its output f(x).

$$f(x) = \begin{matrix} 1 & \text{if } w.x + b > 0 \\ 0 & \text{otherwise.} \end{matrix} \qquad (4.1)$$

In the equation given above: w is the vector of real-valued weights, bias is an element that changes the boundary away from the origin regardless of the input value, and x is the vector of input x values

$$\sum_{i=1}^{m} wixi \qquad (4.2)$$

where m is the number of inputs to the Perceptron. The output can be represented as 1 or 0. It can also be represented as 1 or -1 depending on which activation function is used

- Perceptron is a supervised learning method for a binary linear classifier with a single layer.
- Optimal weight coefficients are automatically learned.The input characteristics are multiplied by the weights, and a judgment is made about whether or not to activate the neuron.
- The activation function employs a step rule to ascertain if the output of the weighting function is greater than zero.
- The creation of a linear decision boundary permits the separation of the two linearly distinct classes +1 and -1. If the sum of the input signals above a certain threshold, a signal is produced; otherwise, no signal is generated.[9]

1.3.1 Neural Network

Neural network refers to a collection of algorithms meant to identify patterns in sensory input that are roughly based after the human brain. Labeling or grouping raw input, these algorithms interpret sensory data. All real-world input, including sights, sounds, texts, and time data, must be transformed into the vector-based numerical patterns they understand. Using neural networks, data may be grouped and categorized. On top of the data you maintain and administer, they serve as a layer for grouping and categorization. When they are trained on a labeled dataset, they help categorize data and organize unlabeled data based on similarities among the example inputs.

Layers”are made up of nodes. A node is only a spot where computation occurs, roughly patterned after a neuron in the human brain that fires in response to sufficient inputs. A node combines data input with a set of coefficients, or weights, that either amplify or dampen that input, so giving significance to inputs in connection to the objective the algorithm is seeking to learn; for instance, which input is most useful for accurately categorizing data? The total of these input-weight products is then transmitted to a node's so-called activation function, which decides whether and how far the signal should go through the network to influence the final result, such as a classification act. The neuron is "activated" when signals pass”through.[10]

1.3.2 Activation Function

An activation function determines the output of a node. The activation function, also known as the transfer function, transforms input signals into output signals. It transforms the output values to a range, such as 0 to 1 or -1 to 1. It is a simplification of the action potential firing rate of the cell. It shows the likelihood that the cell may ignite. At its most fundamental level, the function is binary: yes (the neuron fires) or no (the neuron does not fire). The output might be a single number or a range of numbers (on/of or yes/no).

Depending on the data kinds and class levels, several sorts of outputs are possible if the neural network kernel (activation) function is modified. Occasionally, the model might provide improved performance.[11] For certain datasets, modifying kernel functions might be detrimental, and it is feasible that the model would cease to operate. If we do not know which kernel function is a good match for our data, we must modify it to discover a model that is a good fit. There are several forms of kernel or activation function that may be used in a neural network, including:

- **Sigmoid** - Logistic regression employs the Sigmoid function. It is a continuous development from 0 to It is beneficial in the output layer and is frequently used for linear regression.

The following equation describes how the Sigmoid function predicts outcomes:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{4.3}$$

Similar to the result of the step function, the output of the sigmoid function is between 0 and 1. At z=0, the curve crosses 0.5, allowing us to establish activation function rules such as: If the sigmoid neuron's output is greater than or equal to 0.5, it outputs 1; otherwise, it produces 0.

The curve of the sigmoid function does not have a jerk. It is silky and has a very pleasant and straightforward derivative of

$$\sigma(z) \times (1 - \sigma(z)) \tag{4.4}$$

This is clear throughout the whole arc.

If z is very negative, the output is nearly 0; if z is extremely positive, the output is around 1; however if z is neither too big nor too little, the output is approximately 0 around z=0. Sigmoid Function is graphically shown in Figure 1.4.

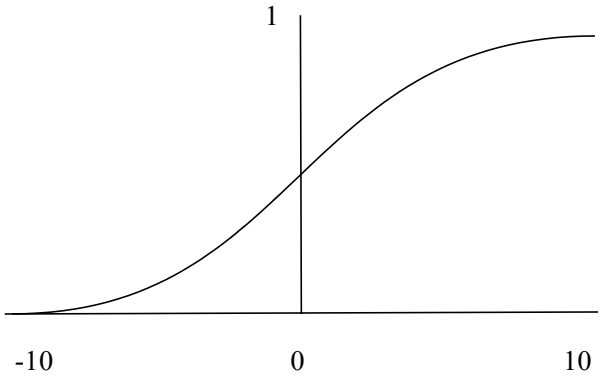


Figure 1.4: Graphical representation of Sigmoid Function

- **Tanh** - hyperbolic function resembles the sigmoid function closely. In contrast to the sigmoid function, which ranges from 0 to 1, the value ranges from -1 to 1. Even though this is not very similar to what occurs in the brain, this function produces better outcomes when training neural networks. Occasionally, neural networks get "stuck" during training using the sigmoid function. This occurs

when there is an abundance of really negative input that maintains the output close to zero, which disrupts the learning process. The Tanh Function is shown graphically in Figure 1.5.

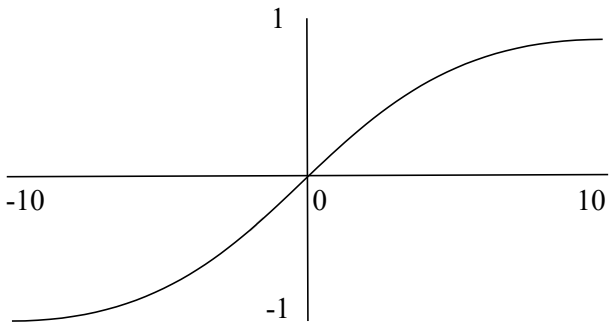


Figure 1.5: Graphical representation of Tanh

- **ReLU** - Rectifier is the most often used activation function. It is the most efficient and realistic solution. This function does not need normalization or other complex computations; its result is either "no" or a percentage of "yes." Figure 1.6 is a graphical illustration of ReLu.

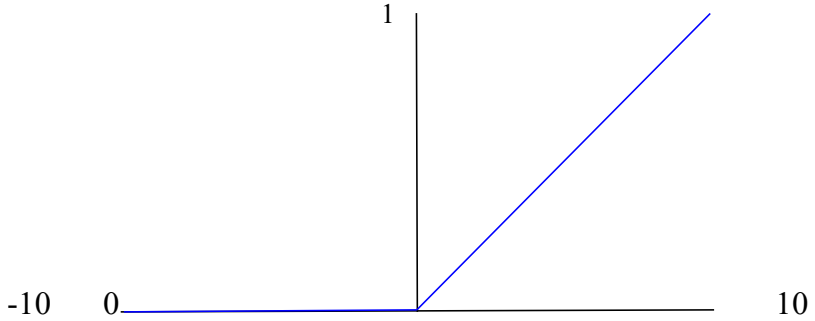


Figure 1.6: Graphical representation of ReLu

- **Threshold** - A threshold function is a Boolean function that assesses whether the value equality of its inputs has surpassed a predetermined threshold.
- **Softmax -** Softmax is an additional activation function used in the output layer of neural network models. Softmax is a mathematical function that turns a vector of integers into a vector of probabilities, proportional to the relative scale

of the vector. When there are more than two class labels, the softmax function is utilized for multi-classification.

1.3.3 Deep Nural Network

A”simplification of a”Deep Neural Network is a hierarchical (layered) design of neurons that are interconnected (similar to the neurons in the brain). These neurons deliver a message or signal to other neurons based on the information they receive, establishing a complex network that learns via a feedback loop. Figure 4.5 illustrates a summary of the deep neural network.

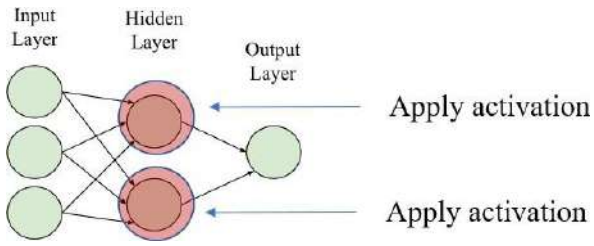


Figure 1.7: Overview of Deep Neural Network

The incoming data is processed by the neurons in the first layer (which are not buried), which then transmit an output to the neurons in the next layer, and so on, until the final output is reached. A conclusion such as Yes or No might be the outcome (represented in probability). Each layer may include one or more neurons, with each neuron calculating a little function known as the activation function. The activation function mimics the signal that will be sent to the next neuron in the circuit. If the value of the incoming neurons above a threshold, the output is transmitted; otherwise, it is ignored. There would be a weight associated with the connection between two neurons from consecutive levels. The weight defines the extent to which the input effects the output of the subsequent neuron and, eventually, the overall final output. All of the initial weights in a neural network are random, but these weights are modified repeatedly throughout model training in order to predict the correct output. Decomposing the network, we can identify a few logical building blocks such as a neuron, layer, weight, input, output, activation function, and finally a learning mechanism (optimizer) that will assist the neural network in incrementally updating the randomly initialized weights to a more appropriate weight that aids in accurate outcome prediction.[12]

1.4 Convolutional Neural Network (CNN)

A convolutional neural network (CNN, or ConvNet) is another class of deep neural networks. CNNs are employed most frequently in computer vision. Given a sequence of photos or videos from the real world, with the application of CNN, the AI system learns to automatically extract the characteristics of these inputs to fulfill a specified goal, e.g., image classification, face identification, and image semantic segmentation. Different from fully connected layers in MLPs, in CNN models, one or multiple convolution layers extract the simple features from input by executing convolution operations. Each layer is comprised of a set of nonlinear functions of weighted sums at different coordinates of spatially adjacent subsets of outputs from the previous layer, allowing the weights to be reused.[13]

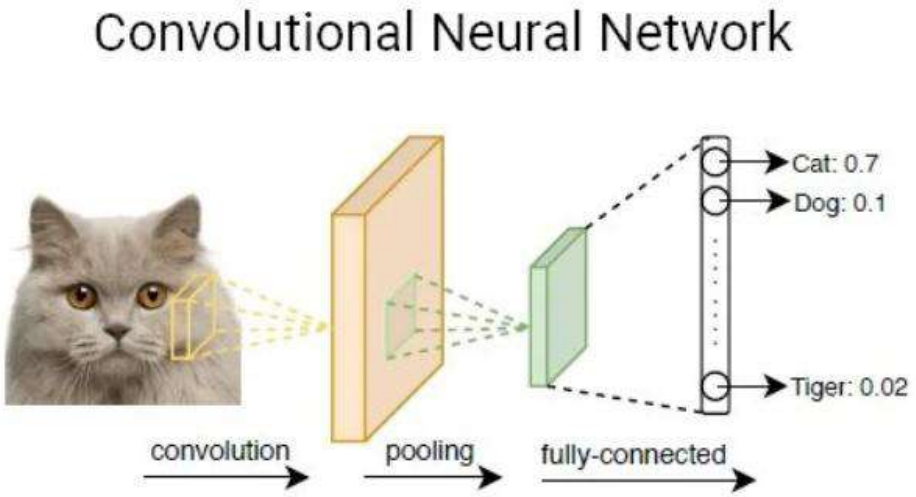


Figure 1.8 -Concept of a Convolution Neural Network (CNN)

Applying multiple convolutional filters, CNN machine learning models can capture the high-level representation of input data, making CNN approaches very useful in computer vision applications. Convolutional neural network example applications include image classification (e.g., AlexNet, VGG network, ResNet, MobileNet) and object detection (e.g., Fast R-CNN, Mask R-CNN, YOLO, SSD) (e.g., Fast R-CNN, Mask R-CNN, YOLO, SSD)

- **AlexNet-** AlexNet, the first CNN neural network to win the ImageNet Challenge in 2012, consists of five convolution layers and three fully connected layers for image classification. AlexNet thus requires 61 million weights and 724 million MACs (multiply-add computation) to classify the 227×227 pixel image.[14]
- **VGG-16-** VGG-16 is trained to a deeper structure of 16 layers consisting of 13 convolution layers and three fully connected layers, needing 138 million weights and 15.5G MACs to identify an image of size 224×224 in order to attain greater accuracy.[15]
- **GoogleNet-**GoogleNet introduces an inception module comprised of filters of varying sizes to improve accuracy while reducing the computation required for DNN inference. As a consequence, GoogleNet achieves a higher accuracy performance than VGG-16 while utilizing just seven million weights and 1.43G MACs to analyse a picture of the same size.[16]
- **ResNet50-** ResNet50, the state-of-the-art system, employs the "shortcut" structure to achieve human-level accuracy with a top-5 error rate of less than 5%. In addition, the “shortcut” module is used to solve the gradient vanishing problem during the training process, making it possible to train a DNN model with a deeper structure.[17]

1.4.1 Deep Residual Learning for Image Recognition

Due to the invention of new technology, the discipline of computer vision has seen significant modifications in recent years. As a direct result of these advancements, it is now possible for computer vision models to outperform humans in solving various image recognition, object detection, face recognition, image classification, etc.-related problems. while it gives us the option of adding more layers to the CNNs to solve more complicated tasks in computer vision, it comes with its own set of issues. It has been discovered that training neural networks gets more harder as the number of additional layers increases, and in certain circumstances, the accuracy also decreases. Here, the application of ResNet becomes significant. Training neural networks with greater depth is more difficult. Resnet makes it feasible to overcome the challenges of training very deep neural networks.[18]

1.4.2 ResNet

ResNet was designed specifically to address this issue. Deep residual nets make use of residual blocks to increase the accuracy of the models. The idea of "skip connections," which is fundamental to residual blocks, is the strength of this form of neural network.Residual Network (ResNet) was presented for the first time in the publication "Deep Residual Learning for Image Recognition". ResNet's primary objective is to generate a "identity shortcut connection" that bypasses one or more layers.

The introduction of ResNet or residual networks, which are made up of Residual Blocks, has reduced the issue of training extremely deep networks.

The first thing we see is a direct link that avoids many tiers. The crucial component of residual blocks is a connection known as the'skip connection.'

The layer's output has changed as a result of this skip connection. Without this skip link, the input 'x' is multiplied by the layer's weights, followed by the addition of a bias term.

This term is then subjected to the activation function f(), and the result is H. (x). The output has changed since the skip connection was implemented.[17]

When the dimensions of the input and output mismatch, as may happen with convolutional and pooling layers, there seems to be a little difficulty with this technique. When the dimensions of f(x) vary from those of x, we have two options:

- The skip connection is padded with additional zero entries to increase its size.
- To match the dimension, the projection method is used, which adds 1×1 layers to the input.

1.4.3 Importance of ResNet

ResNet's skip connections prevent gradients from vanishing in deep neural networks by routing the gradient via an extra shortcut channel. These connections also aid the model by allowing it to learn the identity functions, ensuring that the higher layer performs as well as or better than the lower layer. ResNet deep learning quickly became one of the most prominent solutions in many computer vision problems because of its impressive outcomes. This model was incredibly successful, as can be observed from the fact that its ensemble earned the top place at the ILSVRC 2015 classification competition with an

error of just 3.57%. Additionally, it won the 2015 ILSVRC & COCO contests for ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation.[17]

1.4.4 Proposed Model

We present an ensemble deep learning strategy that, compared to previous methods, will increase the deep learning prediction accuracy of COVID-19 and reduce the error rate of misclassification. InceptionResNetV2 is a CNN architecture that has been trained on over one million pictures from the ImageNet database.

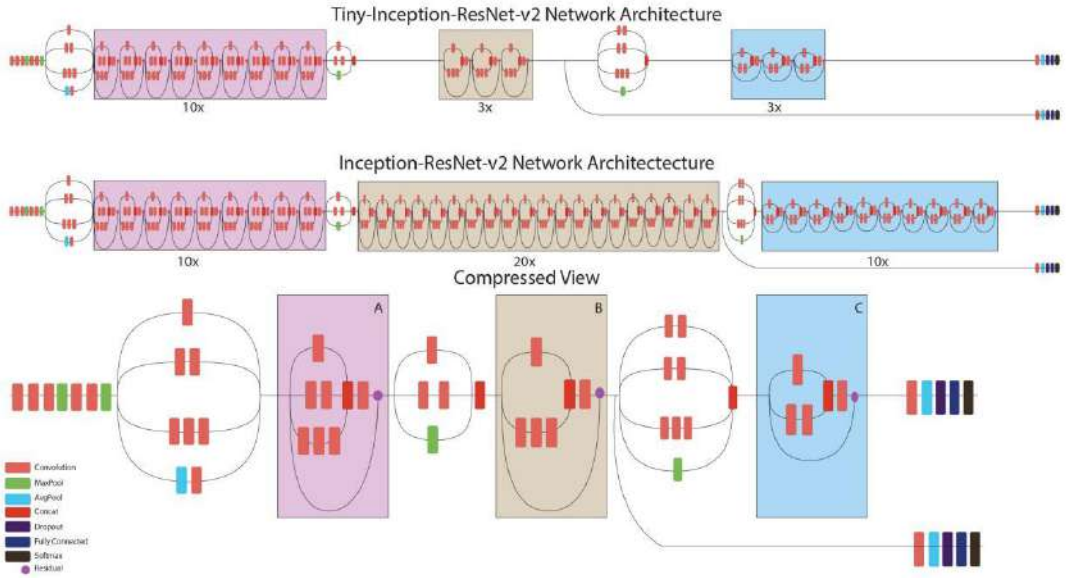


Figure 1.9- Basic Architecture of Inception-ResnetV2 Network

It provides excellent performance at a relatively modest computational cost. Inception variations are nonresidual, while InceptionResNetV2 variants are residual. This distinction implies that the batch-normalization idea is applied only on top of the conventional layer and not on top of the residual summations [19]. InceptionResNetV2 has a natural depth of 164 layers, and with the addition of three levels in our method, it has 167 layers. The model has a total of 55,919,843 parameters, of which 1,580,035 are trainable and 54,347,808 are not.

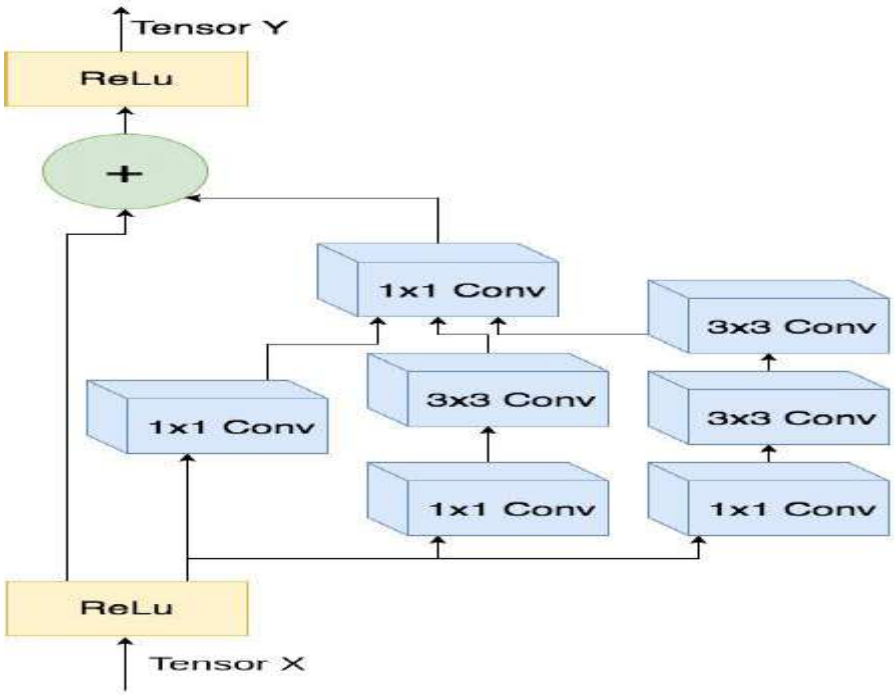


Figure 1.10 -Structure of Inception-ResnetV2

1.5 Popular Models

Because the usage of deep learning has increased at an alarming pace, the ecosystem's maturity has also increased considerably. These deep learning frameworks provide us with reusable code blocks that abstract the logical blocks, in addition to a variety of additional helpful modules for constructing a deep learning model.

We may categorize the available alternatives into two categories: low-level deep learning frameworks and advanced deep learning frameworks. Despite the fact that this terminology is not industry-standard, we may utilize it to intuitively grasp the frameworks. Low-level frameworks provide a more basic level of abstraction while yet allowing a great lot of customization and flexibility. High-level frameworks further simplify our duties by combining abstractions, but restricting the extent of customization and flexibility. A low-level framework functions as the backend for high-level frameworks, which generally function by converting the source code into the required low-level framework for execution. Here are a few of the most prevalent deep learning frameworks.

1.5.1 DenseNet201

Recent research has shown that convolutional networks may be much deeper, more accurate, and easier to train if they have shorter connections between input and output layers.DenseNet, proposed by Gao Huang et al [], is 201 layers deep, with each layer inheriting additional inputs from all preceding layers and passing on its own feature-maps to all succeeding layers. Due to the possibility of feature reuse by various layers, it has two characteristics: simplicity in the training process and exceptionally, parametrically efficient models. This increases the likelihood of variation in the subsequent layer inputs. Each layer of a DenseNet design is linked to every other layer, thus the term Densely Connected Convolutional Network. There are L(L+1)/2 direct connections for L layers. For each layer, the feature maps of all previous layers are utilized as inputs, and each following layer uses its own feature maps as inputs. DenseNets literally connect every layer to every other layer, as straightforward as this may appear. This is the central, incredibly potent notion. The input of a layer in DenseNet is the concatenation of preceding layers' feature maps.[20]

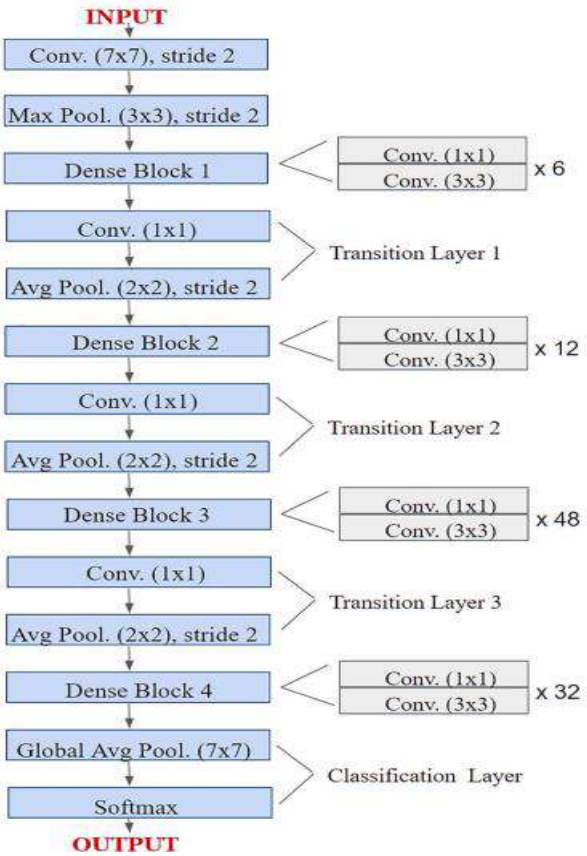


Figure 1.11- Layered Architecture of Densenet201

1.5.2 NasNetLarge

The Google ML team developed the Neural Architectural Search (NAS) Network, often known as NasNet. Its structure is based on reinforcement learning. NAS-NetLarge has been trained on more than one million photos from the Imagenet database and is capable of classifying images into one thousand class groups.

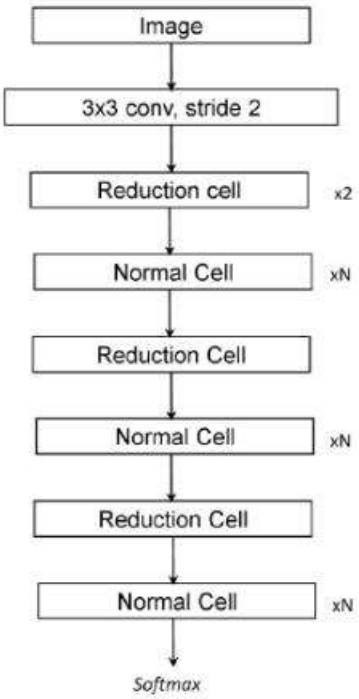


Figure 1.12-architecture of NasNetLarge

NASNet-Large has 89,065,813 parameters, of which 4,140,931 are trainable and 84,924,882 are not. It is a CNN architecture with a 331 by 331 input picture size. The architecture's components consist of a Controller Recurrent Neural Network (CRNN) and a CNN block.[21]

1.5.3 Xecption

The Xception is a 71-layer CNN architecture that extends the Inception model established by Francois Chollet in [22]. On the ImageNet dataset, Xception is known to outperform Inception v3 using Xception. This architecture reestablishes the inception module with depthwise separable convolutions operations, where the convolutions are performed both depthwise and pointwise.

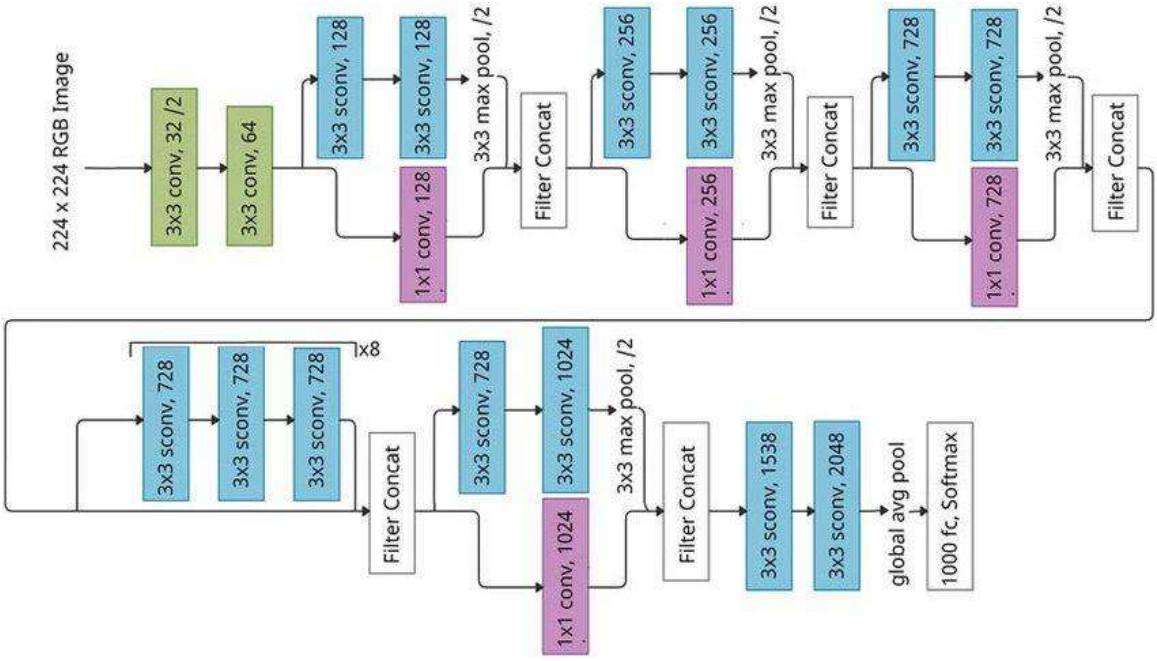


Figure 1.13- Architecture of Xecption

It contains a total of 22,970,923 parameters, of which 2,105,347 are trainable, and comprises of depth-wise independent convolution layers as opposed to the standard convolution layers. It takes into consideration the mapping of spatial correlations and cross-channel correlations, which CNN feature maps may dissociate in their entirety. A second way to employ a pre-trained model is to train not just a new classifier but also the important feature extraction-reliant upper convolutional layers of the pre-trained model.

1.5.4 ResNet50

ResNet-50 is a 50-layer deep convolutional neural network. ResNet, which is an abbreviation for Residual Networks, is a traditional neural network that serves as the foundation for several computer vision applications. The revolutionary aspect of ResNet was that it enabled us to train neural networks with 150+ layers or more. In their 2015 computer vision research article titled 'Deep Residual Learning for Image Recognition,' Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun proposed a novel neural network.[17]

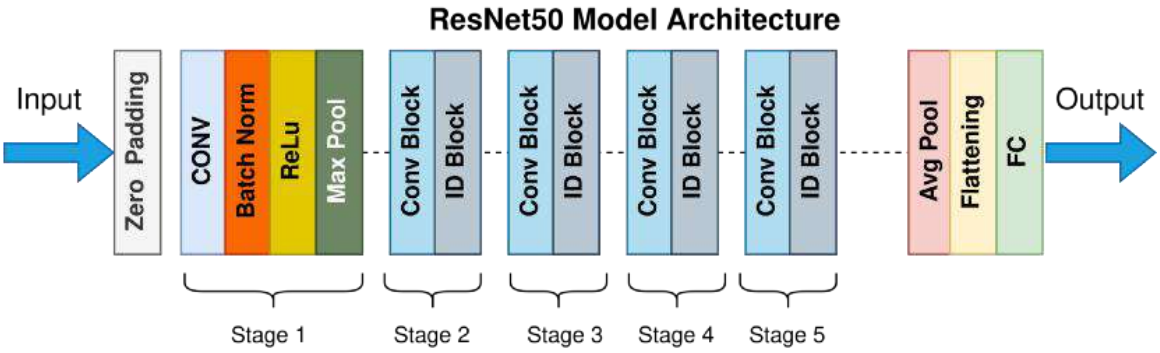


Figure 1.14- ResNet50 Model Architecture

the 50-layer ResNet design incorporates the following elements:

- A 77 kernel convolution with 64 additional kernels with a 2-sized stride.
- A maximum pooling layer with a stride of two.
- Additional 9 layers consist of 33,64 kernel convolution, 11,64 kernel convolution, and 11,256 kernel convolution. Three times, these three levels are repeated.
- Iterate 12 further layers with 11,128 kernels, 33,128 kernels, and 11,512 kernels four times.
- 18 further layers comprised of 11,256 cores, 2 cores 33,256, and 11,1024, iterated six times.
- 9 further layers consisting of 11,512 cores, 33,512 cores, and 11,2048 cores repeated three times.(as of now, the network has fifty layers)
- Average pooling followed by a 1000-node fully connected layer with softmax activation function.

Convolutional Neural Networks have a significant drawback known as the "Vanishing Gradient Problem." During backpropagation, the gradient value reduces dramatically, resulting in little weight changes. To circumvent this, ResNet is used. It utilizes "SKIP CONNECTION."

1.6 Goal

Overall, the purpose of this thesis is to screen for Covid-19 utilizing a less costly, rapid, and accurate method in order to protect individuals from Covid-19 infection. Covid-19 might be transmitted by an infected person two days before difficulties appear or a positive test result is acquired, therefore only early identification can keep individuals

healthy. Patients with Covid-19 may not usually have clinical symptoms. Even if a person was wearing a mask near a COVID-19 carrier, they are still considered a direct contact. Finally, the effects of input image modification and the methodologies used to analyze the outcomes of each implementation, as well as the findings of the study of all the final implementations, will be addressed.

1.7 Procedure of the proposed method

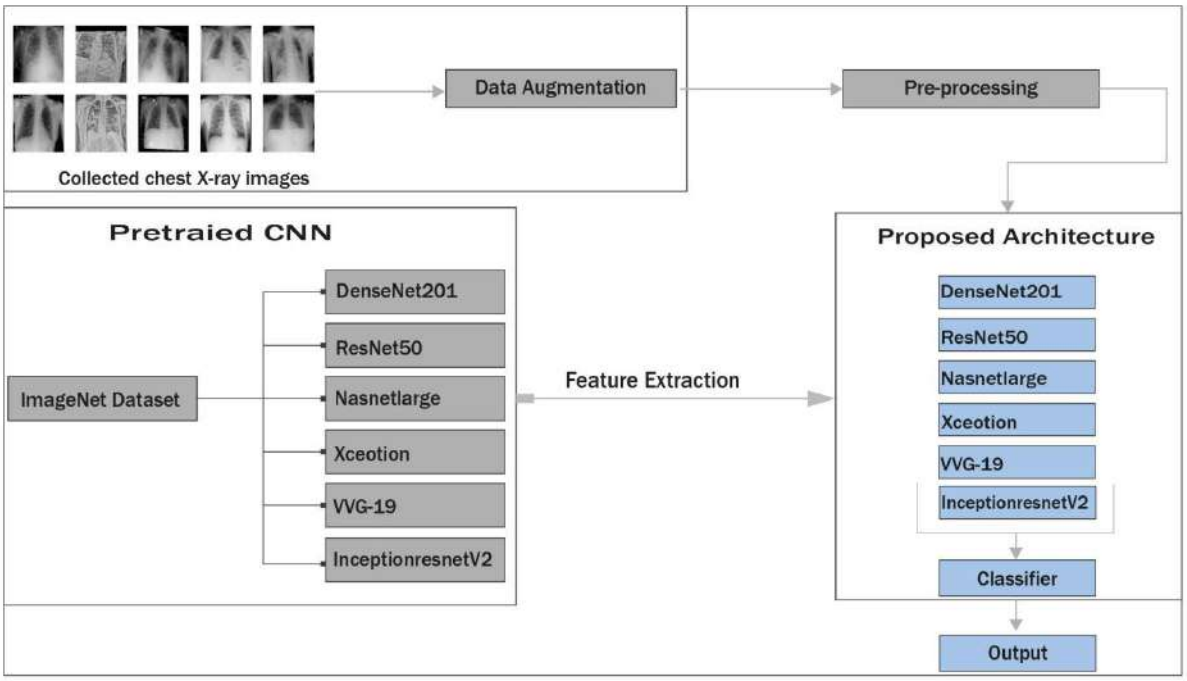


Figure 1.15- Procedure of the proposed work

In the above figure, we try to show the procedure of the proposed work. Collect dataset by data argumentation then prepossessing by proposed architecture. Then we will find output.

1.8 Methodology of the work

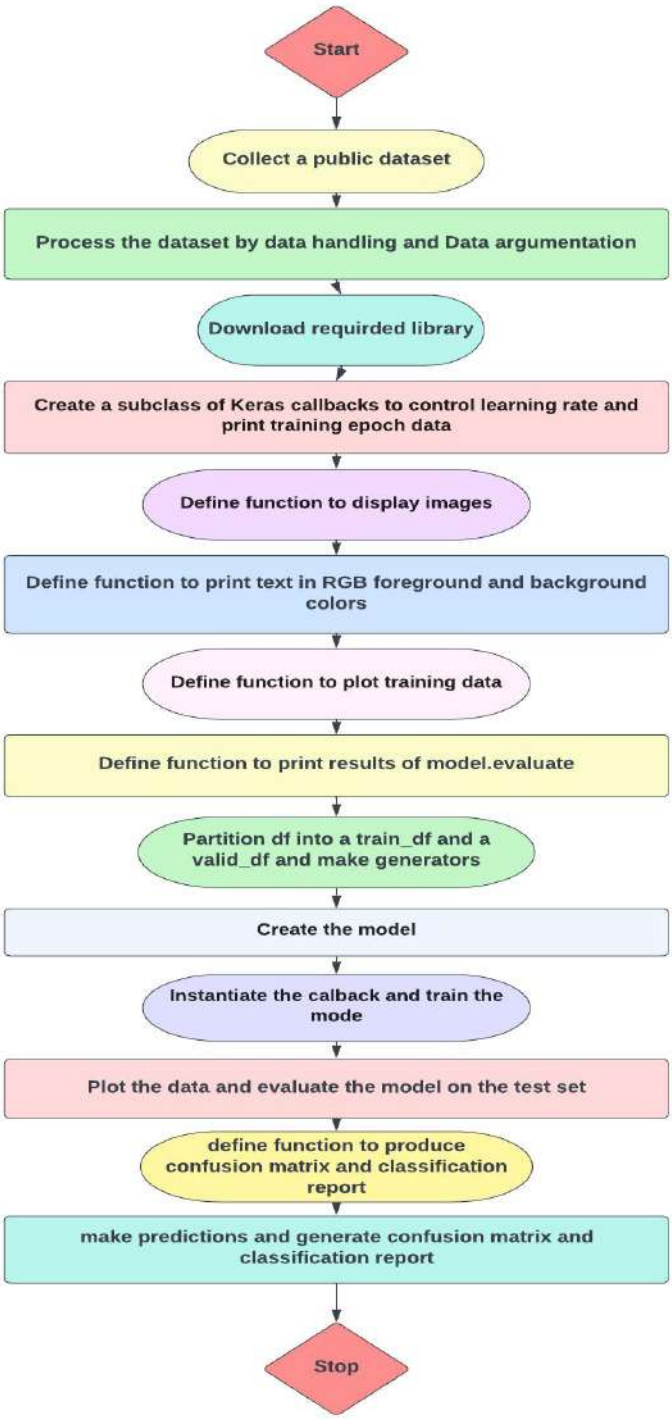


Figure- proposed method the work

1.9 Thesis Outline

The remaining thesis sections are structured as follows. We summarize the literature review in Chapter 2. In Chapter 3, we discuss the experimental datasets we employ. Then, in Chapter 4, we describe Experimental setup. Chapter 5 we discuss about result.

Comparative analysis is presented in Chapter 6. In Chapter 7, closes the thesis and discusses future work.

1.10 Main contribution summary

- The proposed model integrated various convolution models with efective activation layers to support better learning for extracting elite patterns to be classifed.
- To determine the feasibility of the proposed scheme, detailed comparative analyses are conducted using various measurement criteria such as accuracy, precision, recall, F1-score,
- The data is also tested on diferent pre-trained models and hyperparameter tuning has been carried out to achieve optimal accuracy
- The main difference between our work and the previous studies is that this study incorporates a large number of Dataset can train by the model we excute and Data limitation will not be any issue and accuracy loss.
- The proposed models obtain the test accuracy of 97.5% the 29994 CXR image dataset.

1.11 Summary

So, this chapter provides an introduction to the thesis topic and its motivations and aims, a brief discussion of the methods to be used throughout the body of the work, and an overview of the overall thesis structure. An overview of deep neural networks is provided in this chapter . In the overview, you'll find information on the following: perceptron, neural network, activation function, deep neural network, recommended Model, and common models. In this analysis, we used the Inception-ResnetV2, DenseNet201, ResNet50, Vgg-119, and NasNetLarge models.

Chapter 2
LITERATURE REVIEW

In this section the works by other researchers that are related to this thesis “An Enhanced Novel Coronavirus (COVID-19) Detection Model Based on Lung X-Ray Image using Convolutional Neural Network” will be reviewed, which is vital element to successful research to detecting Covid-19 using Convolutional Neural Network with the existing research. Hence, for the detection of Covid-19 from lung x-ray images with better performance and easy fabrication of a covid detecting model.

2.1 Paper review

1. Research paper on, “CAE-COVIDX: automatic covid-19 disease detection based on x-ray images using enhanced deep convolutional and autoencoder.” In this research, the author introduced a novel technique called CAE-COVIDX, which combines deep CNN with Autoencoder (AE) to extract picture characteristics and categorize COVID-19. 400 typical images of diseases that weren't COVID-19 positive and 400 of those that were COVID-19 positive were evaluated and compared to the conventional CNN and the existing framework VGG16. Accuracy, confusion analysis, and loss evaluation were used to evaluate performance. The CAE-COVIDX framework performs better than earlier approaches in a number of testing scenarios, according to experiment results. The capacity of this framework to identify Covid-19 in a variety of non-standard picture X-rays may be useful to medical professionals in diagnosing Covid-19 patients. To acquire final results based on machine learning, CAE-COVIDX includes the following these crucial steps: Acquisition of COVID-19 image lung data, data image preprocessing, suggested CNN and AE hybridization model, proposed training procedure, and evaluation of the training procedure. After all the testing the total accuracy found 98%.[23]

2. Research paper on, “ULNet for the detection of coronavirus (COVID-19) from chest X-ray images.In this paper, On the foundation of U-net, the suggested architecture is built by including a new down-sampling side, skip connections, and completely connected

layers. The network is known as ULNet because of its UL-like form. A publicly accessible Kaggle dataset (comprising a combination of 219 COVID-19, 1314 normal, and 1345 viral pneumonia chest X-ray images) was used to train and test this model. Binary classification (COVID-19 vs. Normal) and multiclass classification (COVID-19 vs. Normal vs. Viral Pneumonia) were performed. Identification of COVID-19 patients is important for the management of the COVID-19 pandemic. This suggested model is 99.53% and 95.35% accurate at detecting COVID-19 in binary class and multiclass tasks, respectively. This strategy is predicted to aid medical professionals in the diagnosis and detection of COVID-19 in light of these encouraging outcomes. Overall, our ULNet offers a simple way to detect COVID-19 patients, which is helpful for managing the COVID-19 pandemic.[24]

3. Research paper on, “Classification of COVID-19 chest X-Ray and CT images using a type of dynamic CNN modification method”. In this study, a sort of modified MobileNet architecture for COVID-19 CXR image classification and a modified ResNet design for CT image classification were developed. The gradient vanishing problem is specifically addressed by a modification method of convolutional neural networks (CNN) that dynamically combines features from several CNN layers to enhance classification performance. With the exception of COVID-19, viral pneumonia, bacterial pneumonia, and normal controls are all classified using CXR images utilizing the modified MobileNet. Additionally, the COVID-19, non-COVID-19 infections, and normal controls are classified using CT scans using the suggested updated ResNet. The findings demonstrate that the suggested approaches achieve 99.6% test accuracy on the five-category CXR image dataset and 99.3% test accuracy on the CT image dataset. Comparative investigations employ two distinct COVID-19 detection methods, COVID-Net and COVIDNet-CT, along with six sophisticated CNN architectures. To assess the effectiveness of the aforementioned models, two benchmark datasets and a CXR picture dataset made up of eight independent CXR image sources are used. The findings indicate the suggested methods' potential for computer-aided diagnosis in healthcare applications by showing that they perform better than comparative models in classification accuracy, sensitivity, and precision. In this study, COVID-19, bacterial pneumonia, viral pneumonia (apart from COVID-19), and tuberculosis are all classifications of lung disorders that are taken into consideration. Healthy cases are added as a fifth class as well. This study's dataset was constructed by combining information from four sources that are all freely

accessible. The distribution of samples over 7 data sources is 1770, 1436, 1341, and 1700, respectively, for COVID-19, tuberculosis, normal, and bacterial pneumonia.[25]

4. Research paper on, “A multiple input deep convolutional attention network for Covid-19 diagnosis based on chest CT and chest X-ray”. This paper presents Convolutional neural network-based deep learning models that were specifically designed and trained were used to find pneumonia brought on by COVID-19 respiratory problems. 368 confirmed COVID-19 patients' chest X-ray photos were gathered locally. Additionally, information from three publicly accessible datasets was utilized. Four criteria were used to evaluate the performance. First, training and testing were conducted using the public dataset. Second, to train and test the models, data from both local and public sources were merged. Third, only the local data were utilized for testing and the public dataset was used to train the model. By testing the detection models' capacity to generalize to new data without overfitting the model to particular samples, this method gives detection models more validity. The local dataset was used for testing, while the combined data were used for training. The combined dataset yields a high detection accuracy of 98.7%, and most models handled fresh data with only a small loss in accuracy.[26]

5. Research paper on, “An Efficient CNN Model for COVID-19 Disease Detection Based on X-Ray Image Classification”. This research proposes a deep CNN architecture for the classification of chest X-ray images in the diagnosis of COVID-19. An efficient and precise CNN classification was difficult due to the lack of a chest X-ray image collection that was large enough and of high enough quality. The dataset has been preprocessed in different phases using different techniques to create an effective training dataset for the proposed CNN model to achieve its best performance. This was done to deal with these complexities, such as the availability of a very-small-sized, imbalanced dataset with image-quality issues. Preprocessing of the datasets used in this investigation involved dataset balance, image interpretation by medical professionals, and data augmentation. Experimental results showed an overall accuracy of 99.5%, demonstrating the suggested CNN model's strong suit in the relevant application domain. 450 COVID-19 pictures and 450 normal images make up this data collection. Two scenarios were used to evaluate the CNN model. Using the 100 X-ray pictures from the original processed dataset, the model was evaluated in the first scenario and achieved 100% accuracy. The model has been

tested using an independent dataset of COVID-19 X-ray images in the second scenario, while performance in this test scenario reached 99.5%.[27]

6. Research paper on, “COVID-19 detection from lung CT-Scans using a fuzzy integral-based CNN ensemble”. In this research, the author presents a method for classifying chest CT-scan pictures into COVID and Non-COVID categories using a Sugeno fuzzy integral ensemble comprising four pre-trained deep learning models, namely, VGG-11, GoogLeNet, SqueezeNet v1.1, and Wide ResNet-50-2. A publicly accessible dataset was used to test the suggested framework, and it was shown to have 98.93% accuracy and 98.93% sensitivity on the same dataset. On the same dataset, the model performs better than cutting-edge techniques, demonstrating its accuracy as a COVID-19 detector.[28]

7. Research paper on, “MFBCNNC: Momentum factor biogeography convolutional neural network for COVID-19 detection via chest X-ray images”. In order to automatically optimize the hyperparameters value of the models in accordance with various detection targets, this study provides an improved intelligent global optimization technique that is based on biogeography-based optimization. We suggested including a comparison operation to compare their values in the optimization process after choosing the immigration of a suitable index vector and the emigration of a suitable index vector. According to the different numerical relationships between them, the corresponding operations are performed to improve the migration operation of biogeography-based optimization. The enhanced algorithm can carry out the automatic optimization operation more effectively. Additionally, our group put forth two frameworks: the momentum factor biogeography convolutional neural network and the framework for biogeography. Additionally, two COVID-19 detection techniques based on the suggested frameworks. This technique uses three convolutional neural networks (LeNet-5, VGG-16, and ResNet-18) as the fundamental classification models for detecting COVID-19, Normal, and Pneumonia in chest X-ray pictures. LeNet-5, VGG-16, and ResNet-18 accuracy are enhanced by 1.56%, 1.48%, and 0.73%, respectively, after the hyperparameters of the models were optimized using a biogeography-based approach. After the hyperparameters of the models were optimized using the momentum factor biogeography-based optimization, the accuracy of LeNet-5, VGG-16, and ResNet-18 increased by 2.87%, 6.31%, and 1.46%, respectively.[29]

8. Research paper on, “Coronavirus disease (COVID-19) detection in Chest X-Ray images using majority voting-based classifier ensemble”. In this paper, the author introduces an automated COVID screening (ACoS) method that identifies healthy, suspicious, and nCOVID-19-infected patients using radiomic texture descriptors taken from CXR images. The proposed system employs a majority vote-based classifier ensemble comprising five benchmark supervised classification algorithms in a two-phase classification approach (normal vs. abnormal and nCOVID-19 vs. pneumonia). Using 2088 (696 normal, 696 pneumonias, and 696 nCOVID-19) and 258 (86 images of each category) CXR images, respectively, the training-testing and validation of the ACoS system are carried out. The obtained validation findings for phases I and II (accuracy (ACC) = 98.062% and area under curve (AUC) = 0.956 and 0.831, respectively) demonstrate the proposed system's promising performance.[30]

9. Research paper on, “CoroDet: A deep learning based classification for COVID-19 detection using chest X-ray images”. This study proposes a novel CNN model dubbed CoroDet for automatically detecting COVID-19 utilizing unprocessed chest X-ray and CT scan pictures. With the development of CoroDet, an accurate diagnostic tool for COVID and Normal, COVID, Normal, and Non-COVID Pneumonia, and 4 Class Classification will be available (COVID, Normal, non-COVID viral pneumonia, and non-COVID bacterial pneumonia). The performance of the proposed model was compared with 10 existing COVID detection techniques in terms of accuracy. Classification accuracies of 99.1% for 2-class classification, 94.2% for 3-class classification, and 91.2% for 4-class classification were produced by the proposed model.[31]

10. Research paper on, “Automatic detection of COVID-19 from chest radiographs using deep learning”. Switching to a rigorous computational model powered by deep neural networks, as suggested in this paper, is one option to accelerate the scale of testing. Chest radiographs, one of the most used imaging techniques for clinical diagnosis due to quick imaging and low cost, are employed in the proposed model to determine whether a subject is infected or not. The dataset used in this study includes 1428 chest radiographs of healthy subjects and confirmed COVID-19 positive cases with common bacterial pneumonia (no infection). In this, the pre-trained VGG-16 model was investigated for classification tasks. In this study, transfer learning with fine-tuning was employed to efficiently train the network using relatively tiny chest radiographs. Initial testing showed

that the model produced positive outcomes and may be effectively used to speed up COVID-19 detection. In scenarios involving two and three output classes, the experiment demonstrated accuracy of 96% and 92.5%, respectively.[32]

11. Research paper on, “Automated medical diagnosis of COVID-19 through EfficientNet convolutional neural network”. The primary goal of this paper is to suggest a convolutional neural network-based medical decision assistance system (CNN). The EfficientNet architecture was used in the creation of this CNN.The authors are unaware of any other study that suggests an automated method for COVID-19 diagnosis using EfficientNet. The findings of a CNN created with EfficientNet and 10-fold stratified cross-validation are thus the main contribution. Two key experiments are presented in this paper. The findings of the binary classification utilizing photos of COVID-19 patients and healthy patients are first displayed. Second, a discussion of the multi-class results utilizing photos from COVID-19, pneumonia, and healthy patients. In this dataset 404 images for training of Normal, pneumonia & covid-19 consists. The findings indicate that the average accuracy for binary and multiclass is 99.62%, and for binary it is 96.70%. The suggested CNN model with EfficientNet displays an average recall value for binary and multi-class data of 99.63% and 96.69%, respectively. On the other hand, binary classification has an average precision value of 99.64%, while multi-class reports a value of 97.54%. The average F1 score for binary classification is 99.62%, and 97.11% for multi-class classification. In conclusion, the suggested architecture can offer a system for automated medical diagnosis that will aid medical professionals in making better decisions in this pandemic situation.[33]

12. Research paper on, “COVID-19 Detection System Using Chest CT Images and Multiple Kernels-Extreme Learning Machine Based on Deep Neural Network”. Using chest CT scan pictures, an unique Multiple Kernels-ELM-based Deep Neural Network (MKs-ELM-DNN) method is suggested in this research for the identification of novel coronavirus disease, also known as COVID-19. Deep features are recovered from CT scan images using a Convolutional Neural Network in the proposed model (CNN). Pre-trained CNN-based DenseNet201 architecture, based on the transfer learning method, is employed for this purpose. The performance of the architecture is estimated using an Extreme Learning Machine (ELM) classifier based on several activation techniques. Last but not least, the class label for the final architecture is chosen by majority voting

for each architecture based on ReLU-ELM, PReLU-ELM, and TanhReLU-ELM.The dataset consists of two classes, non-COVID-19 (no-findings) and COVID-19, and a total of 746 images; with 349 images of COVID-19 and 397 images of no-findings cases. Using the MKs-ELM-DNN model, the accuracy score was 98.36%. The findings show that, when compared to other models, the MKs-ELM-DNN model is more effective than cutting-edge algorithms and earlier research.[34]

13. Research paper on, "COVID-19 detection in X-ray images using convolutional neural networks". The methodology used in this paper processes these photos and categorizes them as positive or negative for COVID-19 using two current deep learning models (VGG19 and U-Net). The proposed system begins with lung segmentation to remove any background data that is irrelevant to the task at hand and could lead to biased results. Next, a classification model trained using transfer learning follows, and finally, results analysis and interpretation using heat maps is carried out. The positive cases dataset contains 6475 images for the train set, 3454 images for the test set, and 2873 images for the validation set for the COVID-19 classification model. The BIMCVCOVID dataset is split into 2342, 1228, and 1040 images for train, test, and validation for the negative cases datasets. There were 1645 pictures, 895, and 700 for the train, test, and validation sets after the BIMCV-COVID-dataset was selected. The Pre-COVID period dataset was then separated into the train, test, and validation sets as follows: 2803 pictures, 1401, and 1265, The most accurate models were able to detect COVID-19 with a 97% accuracy rate.[35]

14. Research paper on, "Lightweight Model for the Prediction of COVID-19. Through the Detection and Segmentation of Lesions in Chest CT Scans"This study suggested a simple model that predicts COVID-19 from chest CT images by segmenting regions based on Ground Glass Opacity and Consolidation. The model leverages Mask R-CNN features for lesion segmentation along with shortened versions of ResNet18 and ResNet34 as a backbone net. COVID-CT-Mask-Net classification model with 6.12M total and 600K trainable parameters achieves 91.35% COVID-19 sensitivity, 91.63% Common Pneumonia sensitivity, 96.98% true negative rate, and 93.95% overall accuracy on COVIDx-CT dataset without any class balance or data manipulation. The same datasets for the train, validation, and test the train/validation split for the segmentation problem is 650, and the test split is 100. The classification model's train/validation/test divides were adapted from COVIDx-CT. All COVID-CT-Mask-Net classifiers were trained using a

random sample of 3000 images (1000 each class) from the train split (more than 60000 images). Splits for validation and testing were fully utilized (21036 and 21192 images resp.). The test results are generally balanced. 35% Common Pneumonia, 25% COVID-19, and 45% Normal/Control. For the training and evaluation of the segmentation model, we only used one positive class, "Lesion," which was created by merging the classification model's masks.[36]

15. Research paper on, “Deep-learning based detection and analysis of COVID-19 on chest X-ray images”. In this study, both patients with covid-19 disease and healthy individuals' chest x-ray scans were viewed from the PA perspective. They employed deep learning-based CNN models and compared their performance after cleaning the images and adding data augmentation. We compared the accuracy of the Inception V3, Xception, and ResNeXt models. 6432 chest x-ray scan samples from the Kaggle repository were gathered in order to analyze the model's performance, of which 5467 were used for training and 965 for validation. When compared to other models, the Xception model has the highest accuracy (i.e., 97.97%) for detecting chest X-ray pictures. This research makes no claims to medical accuracy and merely considers potential classification schemes for covid-19-infected people.[37]

16. Research paper on, “A deep learning approach to detect Covid-19 coronavirus with X-Ray images”. This work proposes an alternative diagnostic method to detect COVID-19 patients using available resources and cutting-edge deep learning algorithms. The proposed approach is put into practice in four stages: data augmentation, preprocessing, creating stage-I and stage-II deep network models. This work uses web resources with 1215 photos that are further strengthened by using data augmentation approaches to improve model generalization and reduce model overfitting by lengthening the dataset overall to 1832 images. Deep network implementation in two stages is intended to distinguish healthy cases, bacterial, and other virus- and bacteria-induced pneumonia on chest X-ray pictures. The effectiveness of the suggested strategy has been thoroughly assessed using both training-validation-testing and (ii) five-fold cross validation procedures. The effectiveness of the suggested strategy is demonstrated by the high classification accuracy of 97.77%, recall of 97.14%, and precision of 97.14% in the case of COVID-19 detection.[38]

17. Research paper on, “Automatic detection of coronavirus disease (COVID-19) using X-ray images and deep convolutional neural networks” .In this study, five models (ResNet50, ResNet101, ResNet152, InceptionV3 and Inception-ResNetV2) based on pre-trained convolutional neural networks have been suggested for the detection of coronavirus pneumonia-infected patients utilizing chest X-ray radiographs. Utilizing five-fold cross-validation, they constructed three distinct binary classifications with four classes (COVID-19, normal (healthy), viral pneumonia, and bacterial pneumonia). The first data set includes 341 covid-19 and 2800 photos of a normal chest x-ray, while the second data set includes 341 covid-19 and 1493 images of viral pneumonia. However, dataset-3 only contains 341 chest x-ray images of Covid-19. According to the performance findings, the pre-trained ResNet50 model outperforms the other four models in terms of classification performance (96.1% accuracy for Dataset-1, 99.5% accuracy for Dataset-2, and 99.7% accuracy for Dataset-3).[39]

18. Research paper on, “Diagnosis and detection of infected tissue of COVID-19 patients based on lung x-ray image using convolutional neural network approaches” .In this study, the authors employed three deep learning-based techniques to identify and classify COVID-19 patients from lung X-ray pictures. They introduced two algorithms for the diagnosis of the illness, including convolutional neural network (CNN) techniques that directly employ lung images and deep neural network (DNN) approaches based on the fractal feature of images. MRI pictures totaling 682 are included in the data set. Photos of patients and average people were separated into two categories: COVID-19 patients and non-COVID-19 images of patients.Results classification demonstrates that the provided CNN architecture outperforms the DNN technique, which has accuracy and sensitivity ratings of 83.4% and 86%, respectively.[40]

19. Research paper on, “Cov XNet: A multi-dilation convolutional neural network for automatic COVID-19 and other pneumonia detection from chest X-ray images withtransferable multi-receptive feature optimization.In order to effectively extract a variety of features from chest X-rays, a deep convolutional neural network (CNN) based architecture called CovXNet is presented. Since the chest X-ray images corresponding to COVID-19 induced pneumonia and other common pneumonias share a lot of similarities, the proposed CovXNet was initially trained using a lot of chest X-ray images relating to both normal and (viral/bacterial) pneumonia patients. Additional fine-tuning layers that

are further trained using less chest X-rays belonging to COVID-19 and other pneumonia patients are used to transmit the learning from this initial training phase. The 5856 images in the datasets used in this investigation include 1583 normal X-rays, 1493 X-rays of viral pneumonia unrelated to COVID, and 2780 X-rays of bacterial pneumonia. In the suggested approach, numerous CovXNets models are created and trained using X-ray images of varying resolutions. A stacking algorithm is then used to further optimize the predictions made by the models. To differentiate the aberrant regions of X-ray pictures related to various types of pneumonia, a gradient-based discriminative localization is integrated at the end.With an accuracy of 97.4% for COVID/Normal, 96.9% for COVID/Viral pneumonia, 94.7% for COVID/Bacterial-pneumonia and 90.2% for multiclass COVID/normal/Viral/Bacterial pneumonias, extensive experiments using two separate datasets produce highly good detection performance.[41]

20. Research paper on, “Deep Learning Frameworks for COVID-19 Detection”In order to analyze chest X-ray images to detect the virus and identify high-risk patients for containing the spread, this paper illustrates how deep learning techniques are highly helpful. Additionally, it demonstrates how deep learning's Convolutional Neural Network (CNN) technology aids in swift virus detection. A CNN is a particular kind of artificial neural network that helps in detection and is used for pre-processing images. Using a dataset of 2159 images, a sequential CNN model is proposed with various kernel sizes, filters, and parameters. The results demonstrate that a model with a sufficient number of filters, max-pooling layers, dropout layers, and dense layers provides the maximum accuracy of 99.53% in coronavirus detection[42]

2.2 Summary

From this chapter, we can learn about the literature review for this thesis's topic and get an overview of the works that are related to the thesis.

2.3 Problem statement

Hemdan et al. [43] COVIDX-Net, a deep learning system, was presented to categorize CXR pictures as positive or negative COVID-19 situations. They find favorable accuracy for normal and positive COVID-19, respectively, despite using seven deep convolutional

neural network models. However, their findings were based on just 50 CXR pictures, which is a relatively tiny dataset for developing a trustworthy deep learning system.

Rajaraman et al.[44] classified CXRs into normal, COVID-19, or bacterial pneumonia with optimal accuracy using iteratively pruned deep learning ensembles. To increase classifification accuracy, many models were evaluated and the best results were pooled using different ensemble procedures. However, since the computational cost of many model computations is substantial, such approaches are best suited for limited numbers of COVID-19 pictures, and there is no assurance that their accuracy will be maintained with huge datasets.[45]

No one could get greater accuracy with additional data sets and they only utilized a maximum of 1,000 to 1,500 data sets for the evaluation, which we found out after doing our literature study. Our goal is to increase the accuracy of big amount of data set by our model and comparing it to other models. We will work to reduce the false positives and errors.

It's also important to note that the findings reported don't always imply the same performance across all data sets. Primary data sets, for example, originate from European patients; other globe patients may have modest data collection differences or diseases, implying that a better categorization utilizing global data sets is required.

Chapter 3
DATA Set Collection

The acquisition of Covid-19 datasets is not simple. On the other side, we compile a variety of publicly accessible CXR benchmark sets.

3.1 Dataset Collection

Released COVIDx CXR-3, a cleaned version of the dataset in which several hundred bad training images have been removed. The new dataset contains 29,986 images from 16,648 patients (variant A)

As there exists no single dataset available for both Covid-19 and non-Covid-19 cases, we considered three different sources to collect Covid-19 dataset and another source to collect healthy and pneumonia dataset. Table 3.1 represents the summary of the dataset.

Table 3.1: Dataset summary

Cases	Dataset (size)	Avg Image size
Covid-19 Positive	16,490	0.5MB
Covid-19 Negative	13,992	0.5MB
Total	30,482	

Table 3.2:Chest radiography images distribution

Type	COVID-19 Negative	COVID-19 Positive	Total
train	13992	15994	29986
test	200	200	400

Table 3.3:Patients distribution

Type	COVID-19 Negative	COVID-19 Positive	Total
train	13850	2808	16648
test	200	178	378

3.2 Data-set Description

Data is the fundamental component of deep learning. For this project, we compiled radiography photos from several public archives and categorized them.We gathered 29986 COVID-19 images from the Kaggle dataset (COVID-19 Radiography Database).

To generate a dataset that is ready for training, images from The dataset must be downloaded first. Next, the desired images are selected and saved to the correctly labeled folder. Images are then loaded and prepossessed by being transformed into numpy arrays with desired size in order to be ready to feed into the training process. As COVID-19 is a relatively new disease, despite past incidences of coronavirus diseases such as SARS in 2002-2003 and MERS in 2012, hospital datasets are extremely difficult to access. Consequently, we depended only on publicly accessible data-sets for our experiment. Other diseases and multiclass labels, such as photos containing both pneumonia and another disease, were omitted from the data-set such that only the aforementioned classes were considered. Every image was read as RGB. Posteroanterior seeing images were only selected for uniformity purposes. Following the compilation and development of our data-set, we randomly divided 80% of the data-set for training and testing and 20% for validation. The resulting data-set was subsequently divided into train and test sets with the same 80:20 ratio. The training set consisted of 13992 Covid negative photos and 15994 covid positive images, whereas the testing set consisted of 200 images of each class.

3.3 Data Augmentation

We utilized ImageDataGenerator from Tensor-flow, which enables us to augment images as the data is fed into the models at each epoch. Images were scaled to 224×224 pixels and enhanced using a variety of settings. First, each image is normalized, followed by the application of a random combination and range of enhancements. This process occurs every epoch and generates diverse training data with random enhancement each time. The fundamental purpose of augmenting our data-set is to enhance its size, reduce over-fitting, and provide diversity.For each training epoch, a fresh image was supplemented. For example, each image was rotated many times.

3.4 Summary

From chapter 3, we are capable of understanding the origin of the data collection and the description of the obtained data-set used in our studies.

Chapter 4
EXPERIMENTAL SETUP

4.1 Experimental setup

The experiments used the Python programming language to train the suggested framework, and Google Colab was used as an editor to run the codes. TensorFlow (1.14), a deep learning framework, and the Keras package are used to construct the background operating environment. The CPU Intel Core i5 9300K - (8 GB/2 TB HDD/512GB SSD/Windows 11 Home/4 GB Graphics) and GPU NVIDIA GeForce GTX 1050Ti were used for all studies. The SGD was used to train the framework using random initialization weights using the SGD Optimizer for (Stochastic Gradient Descent). To prevent over-fitting for all trials, the batch size, learning rate, and number of epochs are experimentally set to 10, 0.0001, and 20 respectively.

4.2 Performance metrics

The following metrics have been incorporated in this study to display the categorized or misclassified examples and measure the classification performance of pre-trained models. The True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) data are used to determine the performance measures.

4.3 Accuracy

It calculates the proportion of correctly classified cases to the entire dataset. The models perform better if the accuracy is higher. A percentage of the projected or classified value to its actual value is the accuracy. It displayed the following:

$$\text{Acc} = \frac{TN+TP}{TN+TP+FN+FP}$$

4.4 precision

It measures the percentage of correctly classified as positive out of all positive cases. It is defined as follows:

$$\text{Pre} = \frac{TP}{TP+FP}$$

4.5 Recall

The recall is computed as the ratio of positives that were correctly predicted as true positives divided by the number of actual positives. It is calculated as follows:

$$\text{Rec} = \frac{TP}{TP+FN}$$

4.6 F1 score

Based on the precision and recall scores, the F1 Score is determined. It offers the model's categorization functionality. Accuracy of the exam is gauged by the F1 score. If the F1 score exhibits the greatest value, then faultless recall and precision are implied. The formula is as follows:

$$\text{F1-S} = 2\times \frac{(Precision\times Recall)}{(Precisoon+Recall)}$$

According to the proposed model, TN represents the proportion of negative cases that are correctly classified as negative, FN represents the proportion of positive cases that are incorrectly classified as negative, and FP represents the proportion of negative cases that are misclassified as positive.

Chapter 5
RESULT

5.1 Setting up google colaboratory

Before obtaining the data, it is required to establish the project's working environment. Google Collaboratory (colab) is a free cloud-based Jupyter notebook environment that requires no configuration. Colaboratory enables browser-based access to powerful computer resources and the ability to create and run programs. Moreover, colab generously supplies GPU, which considerably accelerates the computation-intensive training procedure. For these reasons, colab has proven immensely popular among Deep Learning and Data Science enthusiasts who may not necessarily possess a PC with costly GPUs.

Then install open dataset library

Figure5.1-install library

Then we import the opendataset from kaggle. Because, privet dataset is too cofidential to find. In bangladesh no hospital want to share their dataset for confidential issue. that’s why we use public dataset.

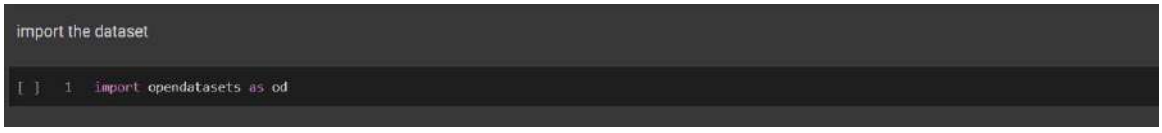


Figure5.2- import dataset

We call a open-source public dataset and download it. After the download we give them a directory.

call the dataset

```
[ ]  1  data_url="https://www.kaggle.com/datasets/andyczhao/covidx-cxr2"
```

download the dataset

```
[ ]  1  od.download(data_url)
```

```
Please provide your Kaggle credentials to download this dataset. Learn more: http://bit.ly/kaggle-creds
Your Kaggle username: ahmedjobyer
Your Kaggle Key: ··········
Downloading covidx-cxr2.zip to ./covidx-cxr2
100%|██████████| 12.9G/12.9G [01:28<00:00, 157MB/s]
```

make data directory

```
[ ]  1  data_dir="./covidx-cxr2"
```

Figure 5.3-dataset download

5.2 Import library

In the backend we use tensorflow and keras.then we import dense,activation,dropout, Conv2D, MaxPooling2D ,BatchNormalization, Flatten layer.Then we import adam and adamax as optimizer and import necessary library

import library

```
1  import os
2  import tensorflow as tf
3  from tensorflow import keras
4  from tensorflow.keras import backend as K
5  from tensorflow.keras.layers import Dense, Activation,Dropout,Conv2D, MaxPooling2D,BatchNormalization, Flatten
6  from tensorflow.keras.optimizers import Adam, Adamax
7  from tensorflow.keras.metrics import categorical_crossentropy
8  from tensorflow.keras import regularizers
9  from tensorflow.keras.preprocessing.image import ImageDataGenerator
10 from tensorflow.keras.models import Model, load_model, Sequential
11 import numpy as np
12 import pandas as pd
13 import shutil
14 import time
15 import cv2 as cv2
16 from tqdm import tqdm
17 from sklearn.model_selection import train_test_split
18 import matplotlib.pyplot as plt
19 from matplotlib.pyplot import imshow
20 import os
21 import seaborn as sns
22 sns.set_style('darkgrid')
23 from PIL import Image
24 from sklearn.metrics import confusion_matrix, classification_report
25 from IPython.core.display import display, HTML
```

Figure 5.4-Import Library

5.3 Create a subclass of Keras callbacks

The callback initially monitors training accuracy and will adjust the learning rate based on that until the accuracy reaches a user specified threshold level. Once that level of training accuracy is achieved the callback switches to monitoring validation loss and adjusts the learning rate based on that.

the callback is of the form:

callbacks=[LRA(model, patience, stop_patience, threshold,factor, dwell, model_name, freeze, batches,initial_epoch, epochs, ask_epoch)]

```
class LRA(keras.callbacks.Callback):
    reset=False
    count=0
    stop_count=0

    def __init__(self,model, patience,stop_patience, threshold, factor, dwell, model_name, freeze,batches, initial_epoch,epochs, ask_epoch):
        super(LRA, self).__init__()
        self.epochs=epochs
        self.ask_epoch=ask_epoch
        self.model=model
        self.patience=patience # specifies how many epochs without improvement before learning rate is adjusted
        self.stop_patience=stop_patience
        self.threshold=threshold # specifies training accuracy threshold when lr will be adjusted based on validation loss
        self.factor=factor # factor by which to reduce the learning rate
        self.dwell=dwell
        self.lr=float(tf.keras.backend.get_value(model.optimizer.lr)) # get the initiallearning rate and save it in self.lr
        self.highest_tracc=0.0 # set highest training accuracy to 0
        self.lowest_vloss=np.inf # set lowest validation loss to infinity
        #self.count=0 # initialize counter that counts epochs with no improvement
       # self.stop_count=0 # initialize counter that counts how manytimes lr has been adjustd with no improvement
        self.initial_epoch=initial_epoch
        self.batches=batches
        #self.epochs=epochs
        best_weights=self.model.get_weights() # set a class vaiable so weights can be loaded after training is completed
        msg=' '
        if freeze==True:
            msgs=f' Starting training using  base model { model_name} with weights frozen to imagenet weights initializing LRA callback'
        else:
            msgs=f' Starting training using base model { model_name} training all layers '
        print_in_color (msgs, (244, 252, 3), (55,65,80))
```

Figure5.5- keras callbacks

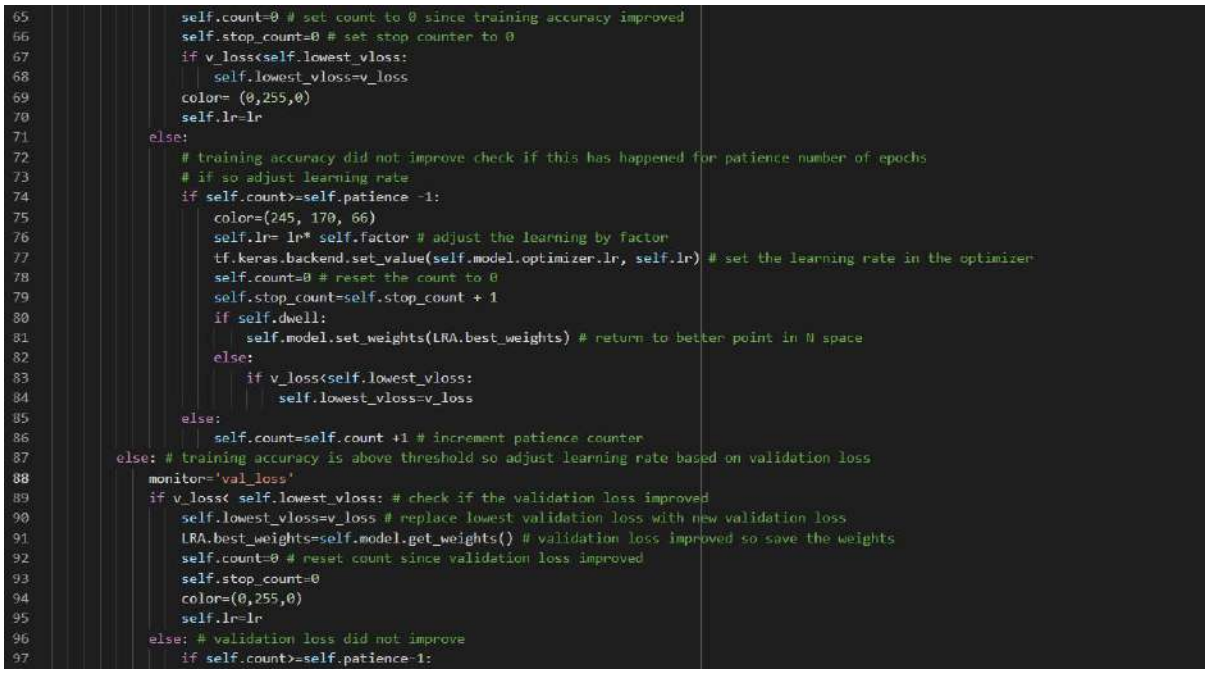
Early halting of training is supported by Keras through a callback named EarlyStopping. This callback enables us to define the performance metric to monitor and the corresponding trigger, which, when activated, will halt the training process.

The EarlyStopping callback is set by parameters when it is constructed.

```
    def on_train_begin(self, logs=None):
        msg='{0:^8s}{1:^10s}{2:^9s}{3:^9s}{4:^9s}{5:^9s}{6:^9s}{7:^10s}{8:^8s}'.format('Epoch', 'Loss', 'Accuracy',
                                                                                       'V_loss','V_acc', 'LR', 'Next LR', 'Monitor', 'Duration')
        print_in_color(msg, (244,252,3), (55,65,80))
    def on_train_end(self, logs=None):
        model.set_weights(LRA.best_weights)
        msg='Training is completed - model is set with weights for the epoch with the lowest loss'
        print_in_color(msg, (0,255,0), (55,65,80))

    def on_train_batch_end(self, batch, logs=None):
        acc=logs.get('accuracy')* 100  # get training accuracy
        loss=logs.get('loss')
        msg='{0:20s}processing batch {1:4s} of {2:5s} accuracy= {3:8.3f}  loss: {4:8.5f}'.format(' ', str(batch), str(self.batches), acc, loss)
        print(msg, '\r', end='') # prints over on the same line to show running batch count


    def on_epoch_begin(self,epoch, logs=None):
        self.now= time.time()

    def on_epoch_end(self, epoch, logs=None):  # method runs on the end of each epoch
        later=time.time()
        duration=later-self.now
        lr=float(tf.keras.backend.get_value(self.model.optimizer.lr)) # get the current learning rate
        current_lr=lr
        v_loss=logs.get('val_loss')  # get the validation loss for this epoch
        acc=logs.get('accuracy')  # get training accuracy
        v_acc=logs.get('val_accuracy')
        loss=logs.get('loss')
        #print ( '\n',v_loss, self.lowest_vloss, acc, self.highest_tracc)
        if acc < self.threshold: # if training accuracy is below threshold adjust lr based on training accuracy
            monitor='accuracy'
            if acc>self.highest_tracc: # training accuracy improved in the epoch
                self.highest_tracc=acc # set new highest training accuracy
                LRA.best_weights=self.model.get_weights() # traing accuracy improved so save the weights
```

Figure5.5- keras callbacks

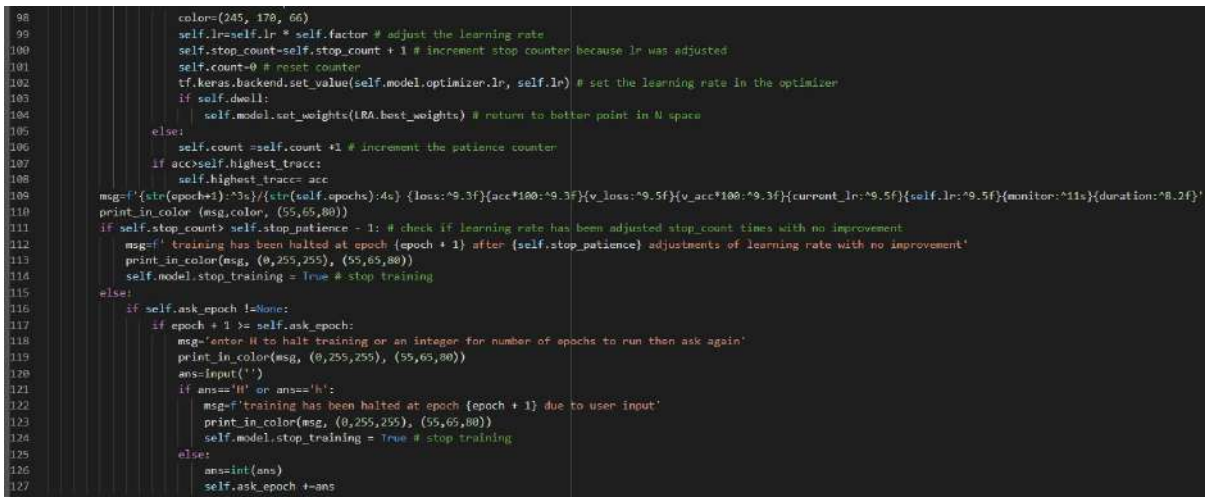
The "monitor" allows us to select the performance metric to be monitored in order to conclude training. Remember that the computation of measures on the validation dataset will have the 'val_' prefix, such as 'val loss' for the validation dataset's loss.



```
                self.count=0 # set count to 0 since training accuracy improved
                self.stop_count=0 # set stop counter to 0
                if v_loss<self.lowest_vloss:
                    self.lowest_vloss=v_loss
                color= (0,255,0)
                self.lr=lr
            else:
                # training accuracy did not improve check if this has happened for patience number of epochs
                # if so adjust learning rate
                if self.count>=self.patience -1:
                    color=(245, 170, 66)
                    self.lr= lr* self.factor # adjust the learning by factor
                    tf.keras.backend.set_value(self.model.optimizer.lr, self.lr) # set the learning rate in the optimizer
                    self.count=0 # reset the count to 0
                    self.stop_count=self.stop_count + 1
                    if self.dwell:
                        self.model.set_weights(LRA.best_weights) # return to better point in N space
                    else:
                        if v_loss<self.lowest_vloss:
                            self.lowest_vloss=v_loss
                else:
                    self.count=self.count +1 # increment patience counter
        else: # training accuracy is above threshold so adjust learning rate based on validation loss
            monitor='val_loss'
            if v_loss< self.lowest_vloss: # check if the validation loss improved
                self.lowest_vloss=v_loss # replace lowest validation loss with new validation loss
                LRA.best_weights=self.model.get_weights() # validation loss improved so save the weights
                self.count=0 # reset count since validation loss improved
                self.stop_count=0
                color=(0,255,0)
                self.lr=lr
            else: # validation loss did not improve
                if self.count>=self.patience-1:
```

Figure5.5- keras callbacks

Based on the choice of performance measure, the “mode” argument will need to be specified as whether the objective of the chosen metric is to increase (maximize or ‘max‘) or to decrease (minimize or ‘min‘).



```
                color=(245, 170, 66)
                self.lr=self.lr * self.factor # adjust the learning rate
                self.stop_count=self.stop_count + 1 # increment stop counter because lr was adjusted
                self.count=0 # reset counter
                tf.keras.backend.set_value(self.model.optimizer.lr, self.lr) # set the learning rate in the optimizer
                if self.dwell:
                    self.model.set_weights(LRA.best_weights) # return to better point in N space
            else:
                self.count =self.count +1 # increment the patience counter
            if acc>self.highest_tracc:
                self.highest_tracc= acc
        msg=f'{str(epoch+1):^3s}/{str(self.epochs):4s} {loss:^9.3f}{acc*100:^9.3f}{v_loss:^9.5f}{v_acc*100:^9.3f}{current_lr:^9.5f}{self.lr:^9.5f}{monitor:^11s}{duration:^8.2f}'
        print_in_color (msg,color, (55,65,80))
        if self.stop_count> self.stop_patience - 1: # check if learning rate has been adjusted stop_count times with no improvement
            msg=f' training has been halted at epoch {epoch + 1} after {self.stop_patience} adjustments of learning rate with no improvement'
            print_in_color(msg, (0,255,255), (55,65,80))
            self.model.stop_training = True # stop training
        else:
            if self.ask_epoch !=None:
                if epoch + 1 >= self.ask_epoch:
                    msg='enter H to halt training or an integer for number of epochs to run then ask again'
                    print_in_color(msg, (0,255,255), (55,65,80))
                    ans=input('')
                    if ans=='H' or ans=='h':
                        msg=f'training has been halted at epoch {epoch + 1} due to user input'
                        print_in_color(msg, (0,255,255), (55,65,80))
                        self.model.stop_training = True # stop training
                    else:
                        ans=int(ans)
                        self.ask_epoch +=ans
```

Figure5.5- keras callbacks

A callbacks function was built in the area just above these numbers. The automatic execution of code and engagement with the process of training models are both made possible by callbacks. It is possible to provide callbacks by using the "callbacks" option

of the fit() method. In order to use callbacks, you must first instantiate them. EarlyStopping is the name of the callback that Keras use to facilitate early termination of training. You will be able to specify the performance measure to monitor as well as the associated trigger with the help of this callback. When the trigger is engaged, the training session will be terminated. When it is formed, the EarlyStopping callback has its settings determined by the arguments. You are possible to establish the performance indicator that has to be monitored in order to complete training by using the "monitor." It is important to keep in mind that the computation of measures on the validation dataset will have the 'val_' prefix, such as 'val loss' for the validation dataset's loss. The "mode" option must specify whether the purpose of the chosen metric is to increase (maximize or'max') or decrease (minimize or'min') based on the performance measure that has been selected.

5.4 Getting COVID-19 chest X-ray images

In this section we partition the data frame into train df and test df. After that numpy array of size into 224×244 and split size is .9. then we put the batch size value 64. After the splitting shuffle the dataset,

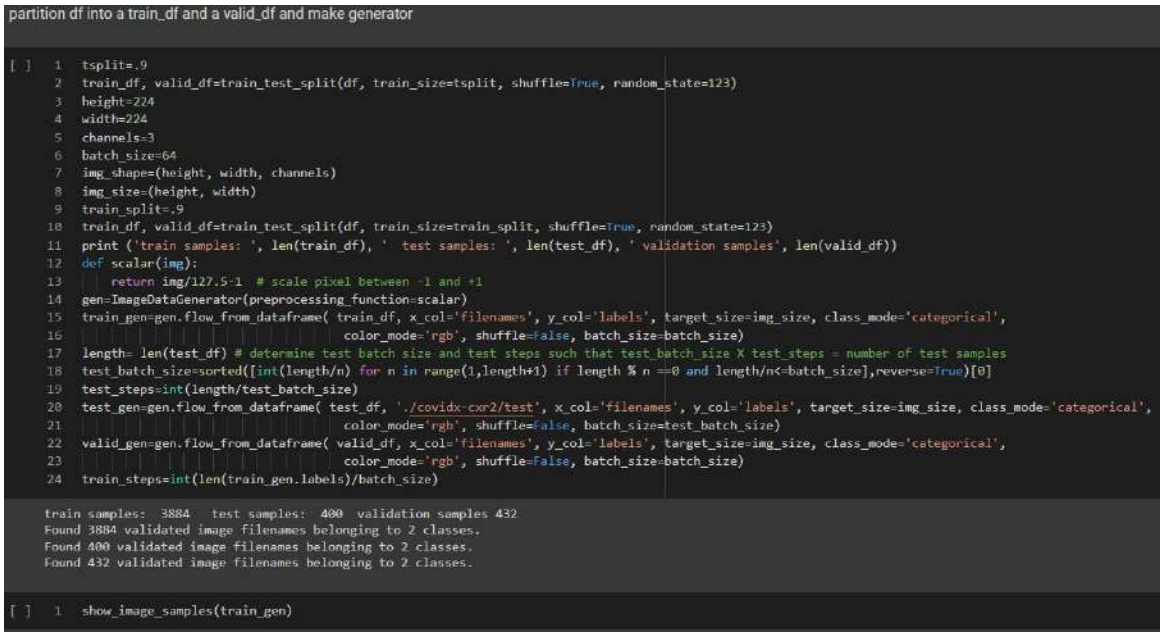


Figure 5.6-dataset prepossessing

Now, the data is ready to be split. Obviously, the train test split function of the sklearn module can accept both Pandas Dataframes and arrays. Therefore, we can simply invoke the relevant function by passing the dataset and the following parameters:

test size:

This option specifies how much of the dataset should be included in the test split. The default value for this option is 0.25, which means that if test size is not specified, the split will consist of 80% train data and 20% test data.

random state:

This option determines the randomization performed on the data prior to the split. By establishing the random state, it is possible to replicate the same data split over numerous function calls.

shuffle:

This option specifies whether the data should be randomized before being divided. Since our dataset is arranged by genre, it must be randomized. If not, the train and test sets would not include the same musical genres.

Then we show the dataset image with level

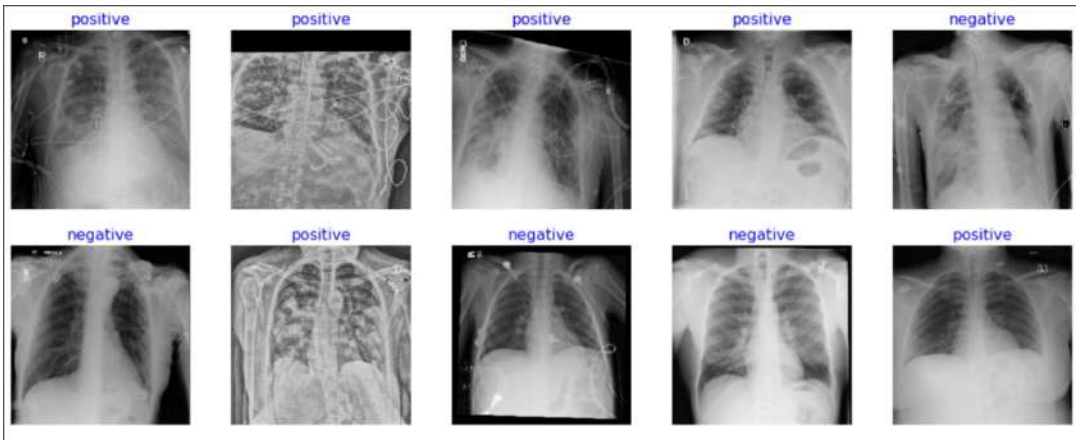


Figure 5.7- Sample Showing

5.5 Prepossessing Data

Before feeding the input into neural networks, the data must be preprocessed. This comprises reading the photos as numpy arrays, scaling the images to the appropriate size, normalizing the images (making the pixel value between 0 and 1), encoding the labels, separating the dataset into training and validation data, and augmentation of the data.

For this study, 30,000 x-ray images have been gathered for each class, with the quantity of accessible photographs of COVID-19 patients being the bottleneck.

Data augmentation is a strategy that might be used to combat overfitting. During the training phase, the generator will essentially randomly change the original picture by rotating, shearing, shifting, and flipping. As a consequence, the model will not be given the same picture again, preventing it from simply remembering all of the training set

photos. The model trained with more data is more robust and generalizable.With 'ImageDatagenerator', implementing it in keras is simple task.

5.6 Create and Training Model

The standard method for picture categorization is a convolutional neural network.We present an ensemble deep learning strategy that, compared to previous methods, will increase the deep learning prediction accuracy of COVID-19 and reduce the error rate of misclassification. InceptionResNetV2 is a CNN architecture trained on more than a million images from the ImageNet database. Since its inception, ImageNet has supplied researchers with a common image database for assessing their models and algorithms. In turn, this has stimulated research in machine learning and deep neural networks, which has simplified the categorization of images and other computer vision-related tasks.

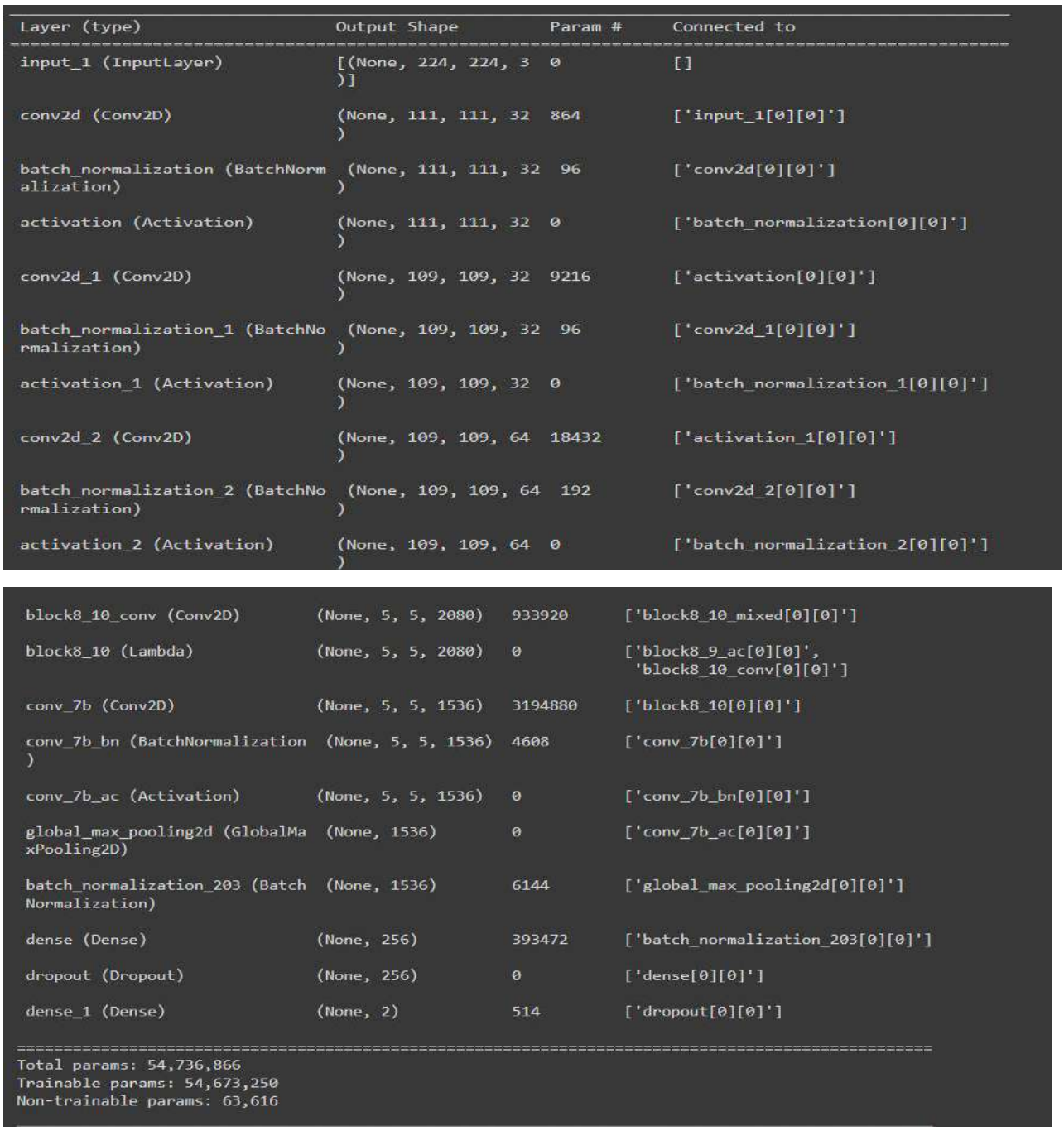
convolution layers with the ReLU activation function are followed by two 2D max-pooling layers and fully linked layers. Max-pooling layers are used to decrease the size of the representation, hence accelerating computing. Dropout is a regularization technique that reduces overfitting by randomly disregarding a percentage of each layer's outputs (50 percent in this case), hence increasing the model's robustness. The model concludes with a single sigmoid-activated unit, which is the default for binary classification tasks.

It offers outstanding performance at a comparatively low computational cost. Inception variants are not residual, but InceptionResNetV2 variants are. This difference indicates that the batch-normalization concept is only implemented on top of the conventional layer and not the residual summations [].

create the model

```
class_dict=train_gen.class_indices
classes=list( class_dict.keys())
class_count=len(classes)
model_name='InceptionResNetV2'
base_model=tf.keras.applications.InceptionResNetV2(include_top=False, weights="imagenet",input_shape=img_shape, pooling='max')
x=base_model.output
x=keras.layers.BatchNormalization(axis=-1, momentum=0.99, epsilon=0.001 )(x)
x = Dense(256, kernel_regularizer = regularizers.l2(l = 0.016),activity_regularizer=regularizers.l1(0.006),
                    bias_regularizer=regularizers.l1(0.006) ,activation='relu')(x)
x=Dropout(rate=.3, seed=123)(x)
output=Dense(class_count, activation='softmax')(x)
model=Model(inputs=base_model.input, outputs=output)
model.compile(Adamax(lr=.005), loss='categorical_crossentropy', metrics=['accuracy'])
model.summary()
```

Figure 5.8- Creating a CNN based Model



```
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            [(None, 224, 224, 3  0           []
                                )]

conv2d (Conv2D)                 (None, 111, 111, 32  864         ['input_1[0][0]']
                                )

batch_normalization (BatchNorm  (None, 111, 111, 32  96          ['conv2d[0][0]']
alization)                      )

activation (Activation)         (None, 111, 111, 32  0           ['batch_normalization[0][0]']
                                )

conv2d_1 (Conv2D)               (None, 109, 109, 32  9216        ['activation[0][0]']
                                )

batch_normalization_1 (BatchNo  (None, 109, 109, 32  96          ['conv2d_1[0][0]']
rmalization)                    )

activation_1 (Activation)       (None, 109, 109, 32  0           ['batch_normalization_1[0][0]']
                                )

conv2d_2 (Conv2D)               (None, 109, 109, 64  18432       ['activation_1[0][0]']
                                )

batch_normalization_2 (BatchNo  (None, 109, 109, 64  192         ['conv2d_2[0][0]']
rmalization)                    )

activation_2 (Activation)       (None, 109, 109, 64  0           ['batch_normalization_2[0][0]']
                                )

block8_10_conv (Conv2D)         (None, 5, 5, 2080)   933920      ['block8_10_mixed[0][0]']

block8_10 (Lambda)              (None, 5, 5, 2080)   0           ['block8_9_ac[0][0]',
                                                                 'block8_10_conv[0][0]']

conv_7b (Conv2D)                (None, 5, 5, 1536)   3194880     ['block8_10[0][0]']

conv_7b_bn (BatchNormalization  (None, 5, 5, 1536)  4608        ['conv_7b[0][0]']
)

conv_7b_ac (Activation)         (None, 5, 5, 1536)   0           ['conv_7b_bn[0][0]']

global_max_pooling2d (GlobalMa  (None, 1536)         0           ['conv_7b_ac[0][0]']
xPooling2D)

batch_normalization_203 (Batch  (None, 1536)         6144        ['global_max_pooling2d[0][0]']
Normalization)

dense (Dense)                   (None, 256)          393472      ['batch_normalization_203[0][0]']

dropout (Dropout)               (None, 256)          0           ['dense[0][0]']

dense_1 (Dense)                 (None, 2)            514         ['dropout[0][0]']

==================================================================================================
Total params: 54,736,866
Trainable params: 54,673,250
Non-trainable params: 63,616
```

Figure 5.9- Showing Parameter

Figure shows the summary of the architecture of our simple Inception-ResNetV2. InceptionResNetV2 has a natural depth of 164 levels. There are a total of 54,736,866 parameters in the model, of which 54,673,250 are trainable and 63,616 are not.

The Model is now ready to train. We use colab GPU instade of TPU for faster acceleration. We set epoch 25 and threshold value was .9

instantiate the calback and train the model

```
epochs =25
patience= 1 # number of epochs to wait to adjust lr if monitored value does not improve
stop_patience =3 # number of epochs to wait before stopping training if monitored value does not improve
threshold=.9 # if train accuracy is < threshhold adjust monitor accuracy, else monitor validation loss
factor=.5 # factor to reduce lr by
dwell=True # experimental, if True and monitored metric does not improve on current epoch set  modelweights back to weights of previous epoch
freeze=False # if true free weights of  the base model
ask_epoch=24 # number of epochs to run before asking if you want to halt training
batches=train_steps
callbacks=[LRA(model=model,patience=patience,stop_patience=stop_patience, threshold=threshold,
                   factor=factor,dwell=dwell, model_name=model_name, freeze=freeze, batches=batches,initial_epoch=0,epochs=epochs, ask_epoch=ask_epoch )]

history=model.fit(x=train_gen,  epochs=epochs, verbose=0, callbacks=callbacks,  validation_data=valid_gen,
                  validation_steps=None,  shuffle=False,  initial_epoch=0)
```

Figure 5.10- Preparing for train the model

After the run, first epoch takes a little time for execution all layer.

```
Starting training using base model InceptionResNetV2 training all layers
```

Epoch	Loss	Accuracy	V_loss	V_acc	LR	Next LR	Monitor	Duration
1 /25	4.589	87.230	14685.01367	59.954	0.00500	0.00500	accuracy	120.28
2 /25	1.611	95.314	15.75892	68.056	0.00500	0.00500	val_loss	73.63
3 /25	0.739	97.554	1.24595	82.176	0.00500	0.00500	val_loss	73.52
4 /25	0.408	98.481	0.90977	89.583	0.00500	0.00500	val_loss	74.45
5 /25	0.270	98.970	0.86141	89.352	0.00500	0.00500	val_loss	73.82
6 /25	0.187	99.279	0.44000	95.370	0.00500	0.00500	val_loss	73.49
7 /25	0.149	99.202	0.41872	93.981	0.00500	0.00500	val_loss	73.65
8 /25	0.135	99.228	0.29565	95.833	0.00500	0.00500	val_loss	73.60
9 /25	0.114	99.150	0.32210	96.065	0.00500	0.00250	val_loss	73.77
10 /25	0.116	99.022	0.25338	94.213	0.00250	0.00250	val_loss	74.50
11 /25	0.104	99.125	0.19798	95.370	0.00250	0.00250	val_loss	73.49
12 /25	0.096	99.202	0.24447	95.602	0.00250	0.00125	val_loss	73.68
13 /25	0.096	99.228	0.17122	96.296	0.00125	0.00125	val_loss	73.71
14 /25	0.091	99.279	0.15998	96.528	0.00125	0.00125	val_loss	73.49
15 /25	0.089	99.253	0.17672	95.833	0.00125	0.00062	val_loss	74.57

Figure 5.11-Epoch running of the model

After the 23 epoch, tranning is complete because of low learning rate and detection accuracy was 99.2%.

```
16 /25    0.089   99.202   0.15706  96.296   0.00062  0.00062  val_loss   73.95
17 /25    0.086   99.279   0.14936  96.296   0.00062  0.00062  val_loss   73.49
18 /25    0.086   99.253   0.15644  96.065   0.00062  0.00031  val_loss   73.87
19 /25    0.086   99.279   0.15114  96.065   0.00031  0.00016  val_loss   73.47
20 /25    0.085   99.253   0.14879  96.528   0.00016  0.00016  val_loss   73.79
21 /25    0.085   99.279   0.14900  96.065   0.00016  0.00008  val_loss   74.48
22 /25    0.084   99.279   0.14887  96.296   0.00008  0.00004  val_loss   73.68
23 /25    0.086   99.279   0.14963  96.296   0.00004  0.00002  val_loss   73.44
 training has been halted at epoch 23 after 3 adjustments of learning rate with no improvement
Training is completed - model is set with weights for the epoch with the lowest loss
```

Figure 5.12-Detection result

Then we put the result in a plot and we see our tranning and validation loss was nearly 0 and epoch is stop because of early stopping.

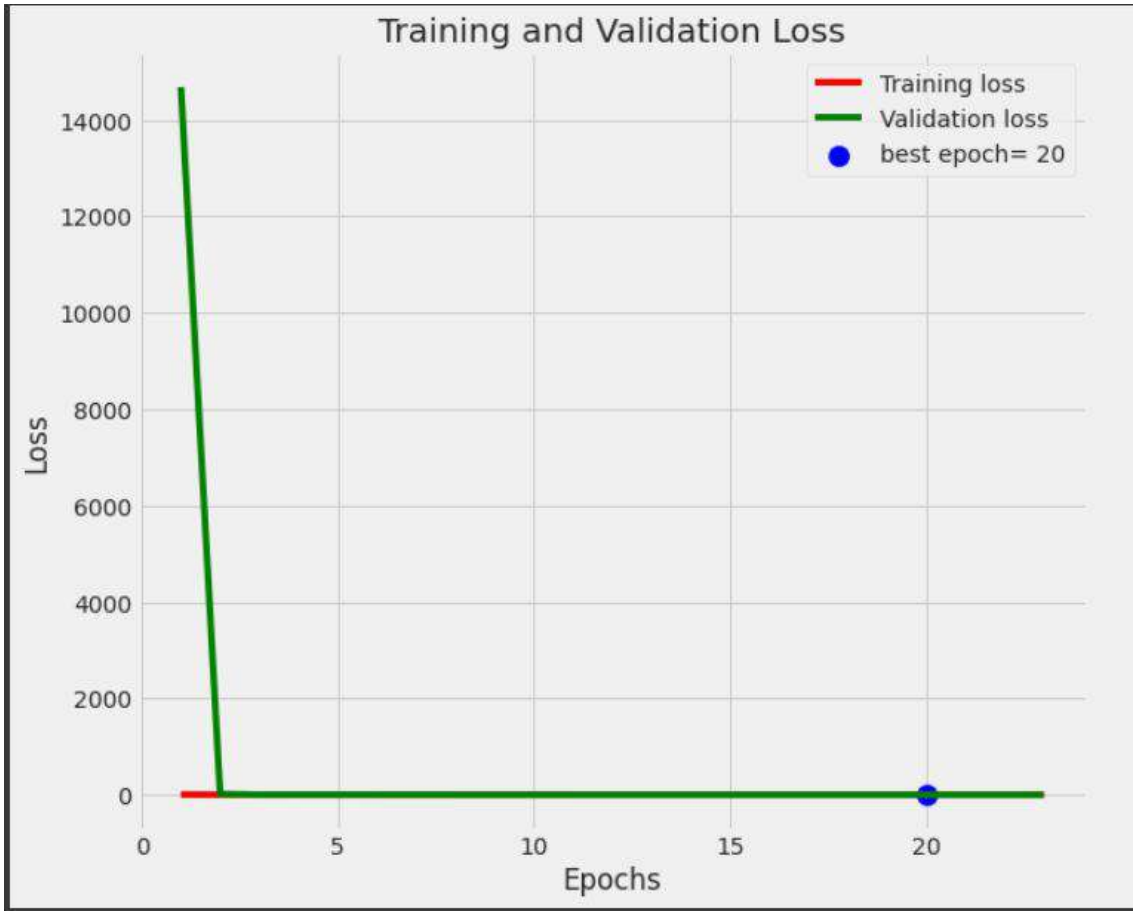


Figure 5.13- Loss of Training and validation

In below figure training accuracy was 99% and validation accuracy was 97.5%.

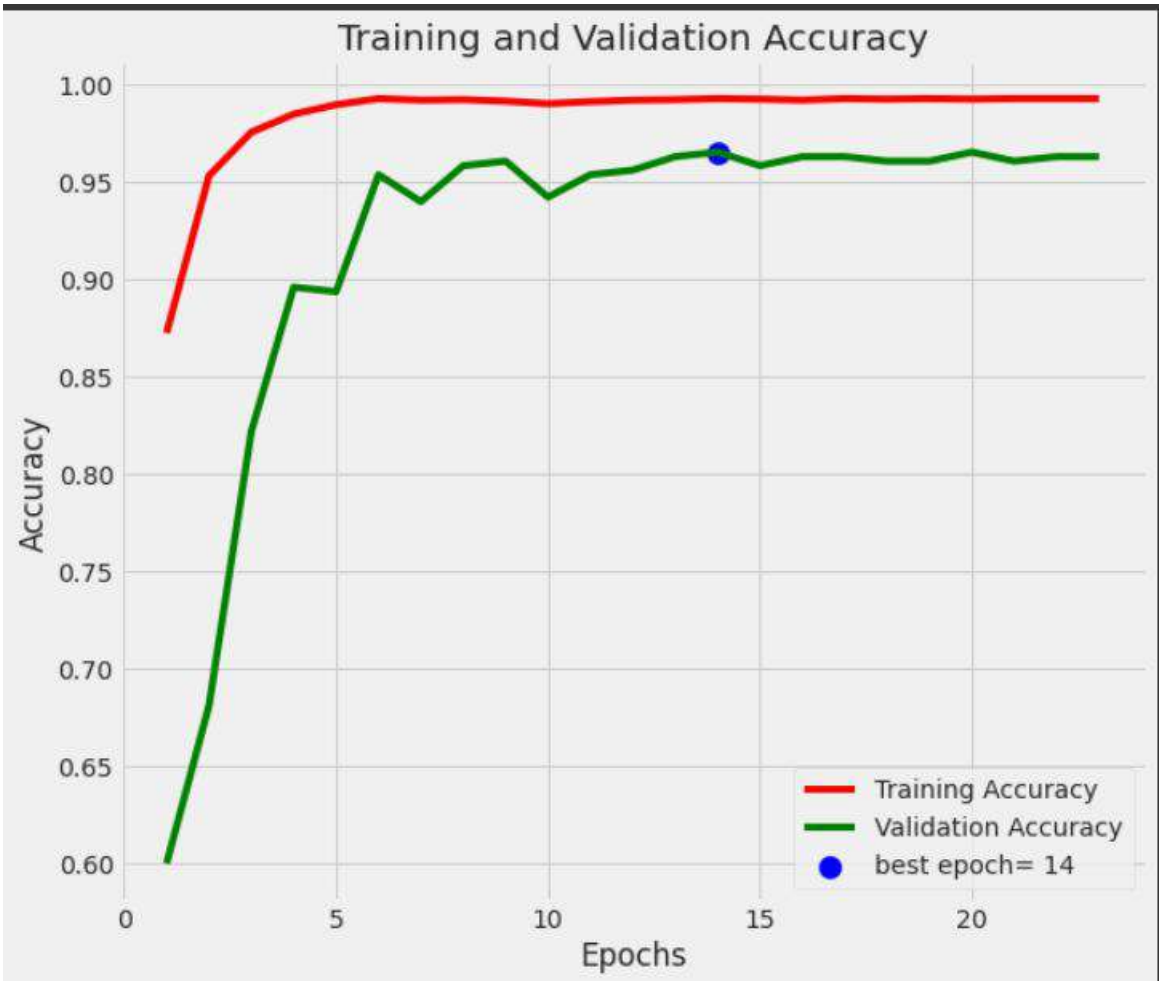


Figure 5.14- Accuracy of training and validation

5.7 Final Result

After running 23 epoch, the LR rate became the lowest and training has been stopped Because of early stopping. Finally our model accuracy was 97.5%

```
[ ]  1  accu=model.evaluate(test_gen,batch_size=test_batch_size, steps=test_steps, verbose=1)[1]*100
     2  msg='Model accuracy on test set: ' + str(accu)
     3  print_in_color(msg, (0,255,0), (55,65,80))

8/8 [==============================] - 21s 2s/step - loss: 0.1336 - accuracy: 0.9750
Model accuracy on test set: 97.50000238418579
```

Figure 5.15- Result of final accuracy model

Predictions performed on the evaluation dataset will be used to compute the evaluation indices once the model has been evaluated. By using sklearn's method [], a report detailing the categorization process is produced.

```
Classification Report:
----------------------
              precision    recall  f1-score   support

    negative       0.95      1.00      0.98       200
    positive       1.00      0.95      0.97       200

    accuracy                           0.97       400
   macro avg       0.98      0.97      0.97       400
weighted avg       0.98      0.97      0.97       400
```

Figure 5.16- Classification report

And the confusion matrices are given below. In the confusion metrics we see our testing dataset was given 200 positive and 200 negative. 200 is detected successfully which was true negativeor covid negative and 190 of 200 is detected as positive is true positive or covid positive.

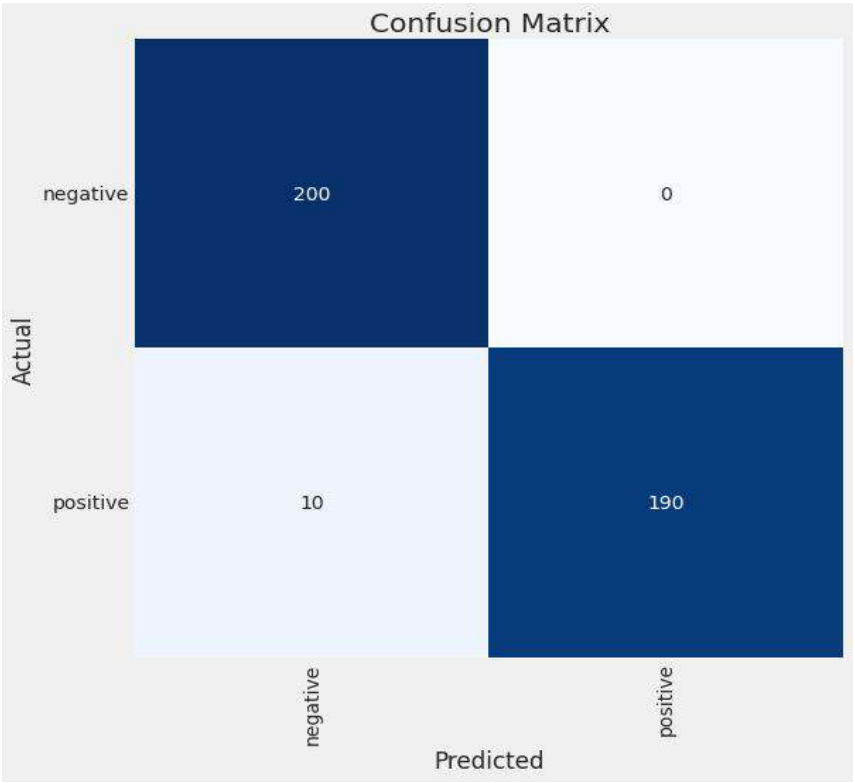


Figure 5.17- Confusion metrics

Chapter 6

Result Analysis

Since the outbreak of Covid-19, various investigations using CXRs to identify Covid-19-positive patients have been proposed/reported []. Deep learning models need a somewhat big dataset, and they may not function as intended in the start of 2020, when we do not have that much clinically annotated data.

6.1 Comparison with other existing model

Table 6.1- comparison with other model

Epoch	Architecture	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
25	DenseNet201	93.08	95.27	94.24	95.26
25	VGG19	89.51	90.42	92.62	91.49
25	ResNet-50	87.58	96.9	79.01	87.97
25	Xception	92.23	90.56	91.43	90.33
25	NasnetLarge	94.25	87.65	90.87	91.58
25	**InceptionResNet-V2**	**97.50**	**95.24**	**1.00**	**97.50**

In the above table we compare our model with other existing model. Our model gives the best result with vast amount of data set.

6.2 DenseNet201-

The table of the result is given below.

Epoch		precision	recall	f1-score	accuracy
	COVID-19	0.95	0.94	0.95	93.08%
25	Normal	0.88	0.95	0.91	

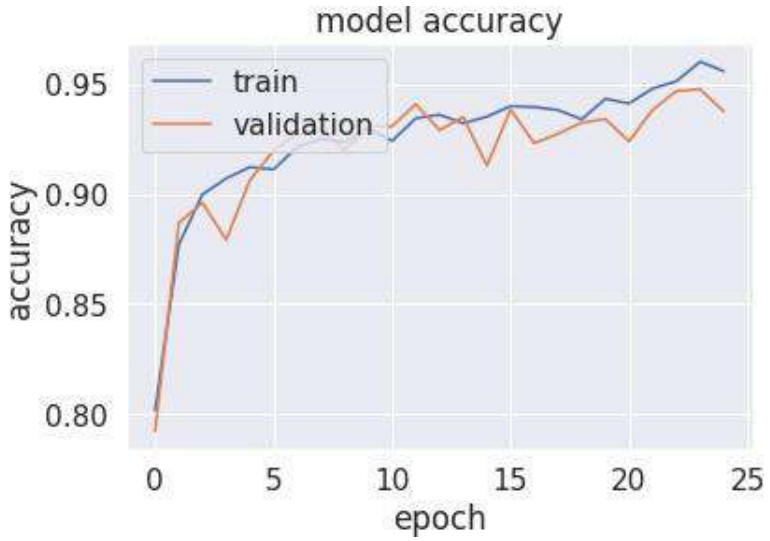


Figure 6.1- Model accuracy of densenet201

The above figure is experiment result of densenet201. In this figure we can see that in 25 epoch, the validation result accuracy is 93.08%.where f1 score was 95% at covid positive.which is nearly fullfil accurate.

6.3 Resnet50-

The table of the result is given below.

Epoch		precision	recall	f1-score	accuracy
25	COVID-19 Normal	0.96 0.75	0.79 0.96	0.87 0.84	87.58%

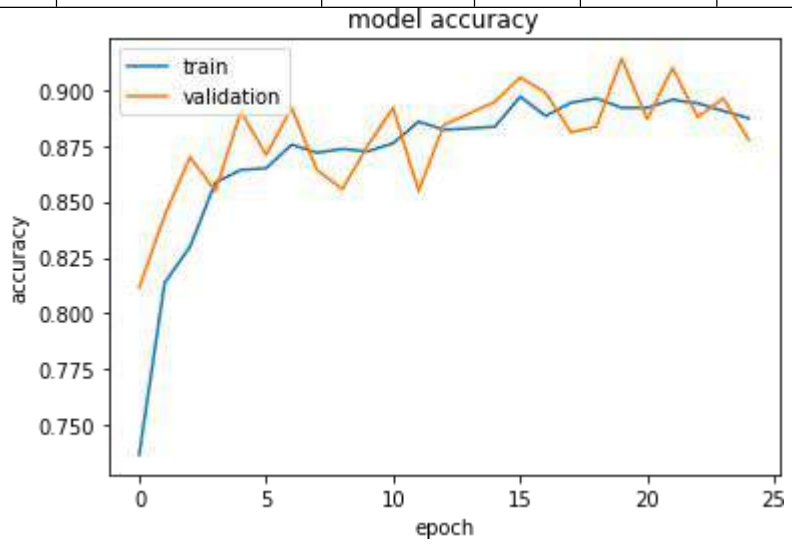


Figure 6.2- Model Accuracy of ResNet50

The experiment shown in the figure above was conducted by Resnet50. From looking at this picture, we can deduce that the validation result accuracy after 25 epochs is 87.58%. in this case the f1 score was 87% at the covid positive. This is not nearly as precise when it comes to detecting.

6.4 NasnetLarge-

The table of the result is given below.

Epoch		precision	recall	f1-score	accuracy
	COVID-19	0.94	0.87	0.90	
25	Normal	0.82	0.94	0.87	91.58%

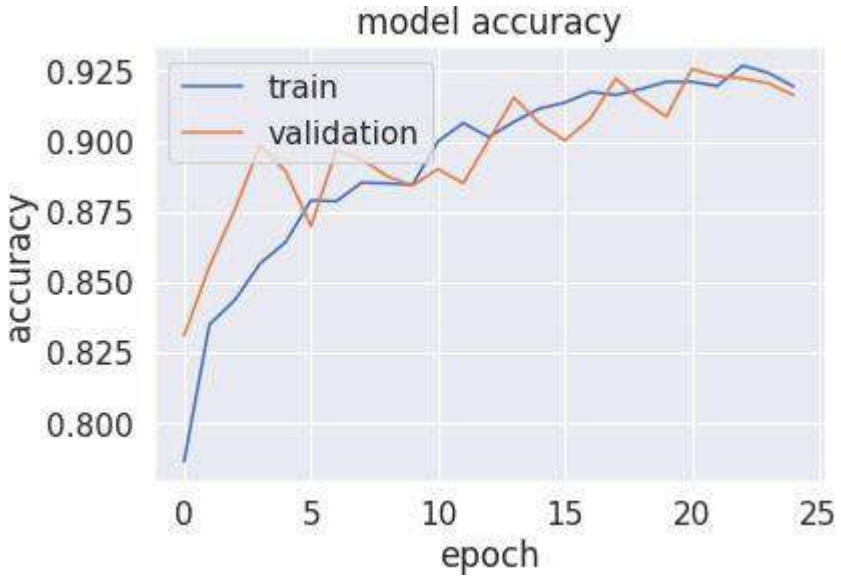


Figure 6.3-Model Accuracy of NasnetLarge

NasNetlarge performed the experiment shown in the preceding graphic. Visual inspection reveals a validation result accuracy of 91.58% after 25 iterations. In this instance, the f1 score at the covid positive was 90%. When compared to other methods of detection, this one is far less reliable.

6.5 Xecption

Epoch		precision	recall	f1-score	accuracy
25	COVID-19 Normal	0.92 0.84	0.90 0.90	0.91 0.87	90.33%

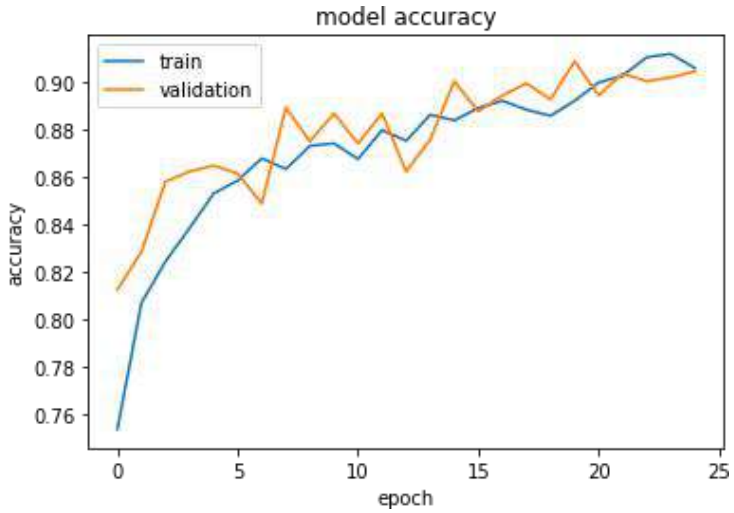


Figure 6.4-Model Accuracy of xecption

Xecption conducted the study shown in the reference image. After 25 epochs, the validation results seem to be accurate 90.33% of the time. The f1 score at the covid positive in this case was 91%. This technique of detection is rather unreliable when compared to others.

6.6 VGG19-

Epoch		precision	recall	f1-score	accuracy
25	COVID-19 Normal	0.90 0.86	0.92 0.92	0.91 0.89	89.83%

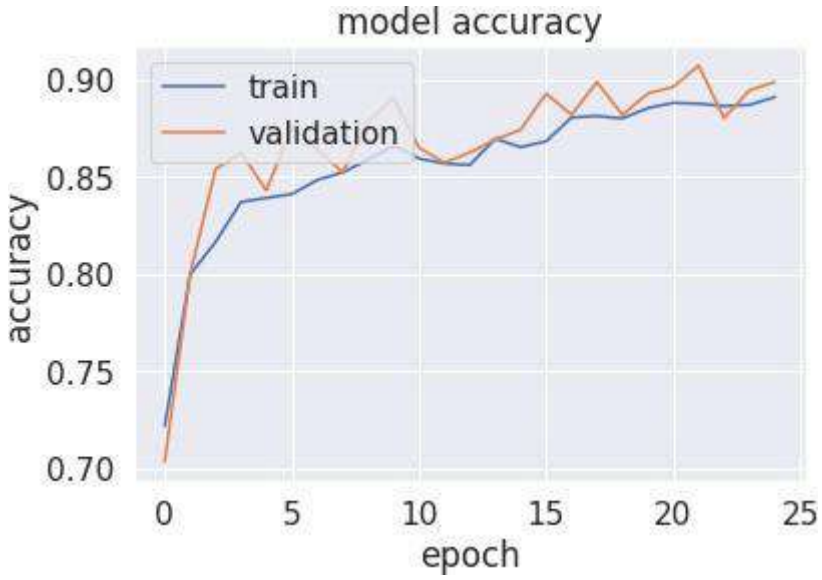


Figure 6.5- Model Accuracy of VGG19

This investigation shown in the picture reference was carried out by VGG19. The validation findings seem to be correct 89.83% of the time after 25 iterations. The f1 score was 91% at the covid positive. The reliability of this detection method is low compared to others.

6.7 InceptionResnetV2

Epoch		precision	recall	f1-score	accuracy
	COVID-19	1.00	0.95	0.97	
25	Normal	0.95	1.00	0.98	97.5%

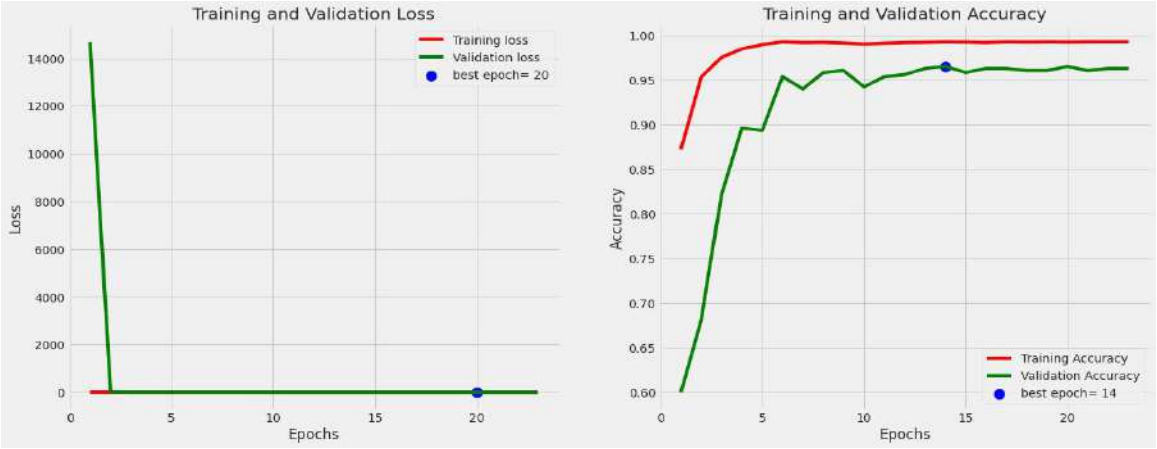


Figure 6.6- Loss and accuracy of our model

The experiment displayed in the graphic above was done using Inception Resnet V2. This illustration shows that after 23 iterations, the validation result accuracy is 97.5 %, or approximately 98 %. In this instance, thCOVIDid positive had the highesF1f1 score of all of the aforementioned models at 99 percent. Kares's callback at an earlier time was a huge assistance. and we had almost no losses, which is promising for the quality of our detection. Both algorithms performed well when trained on the most data possible throughout the training phase. In addition, the CNN algorithm employed maximum epochs to get the best possible accuracy rating from the training data.

6.8 Summary

According to the results of this research, a TL-based system may accurately classify patients into binary and multi-class categories depending on whether or not they have symptoms connected to the chest. Clearly, the suggested model is superior than the alternatives. Numerous results of the analysis indicate that the proposed model surpasses competitor methods.

Chapter 7
Conclusion and Future work

7.1 Conclusion

Two years have passed since the first COVID-19 epidemic, yet a rapid and precise diagnosis is still required. This study implemented and evaluated four pretrained models, DenseNet201, VGG19, ResNet-50, Xception, and NasnetLarge architecture models, for COVID-19 disease detection from chestX-ray images, which are regarded as an inexpensive, rapid, and readily available test that may be used for COVID-19 diagnosis. We used our suggested technique to a large benchmark dataset compiled from publicly accessible chest X-ray image sources. Our method explored the importance of picture preprocessing and enhancement approaches, such as smoothing, denoising, and contrast equalization, for improving model performance, especially with this complicated dataset assembled from many sources. This might be beneficial for other researchers who desire to use this dataset for their own studies. Our findings indicated the efficacy of transfer learning-based approaches in tackling such difficulties with satisfactory outcomes. In this work, we suggested a CNN-based deep learning model that was trained using a huge dataset of 30,000 chest X-ray images. The suggested model was able to reach 97.5% accuracy with an F1-score of 97.0%, therefore it can successfully classify normal and COVID-19-infected samples. Compared to prior research, it seems that our suggested model gets superior results, particularly when the quantity of the dataset is considered.The detection of COVID-19 using radio images and deep learning techniques seems to be a very promising method because of the actual issues encountered by the vaccines regarding the non-acceptation of vaccination by people and regarding the newly appearing COVID-19 variants threatening the efficiency of vaccines. In the near future, these approaches might also be used to identify other chest-related diseases, such as pneumonia, TB, etc. The robustness and precision of the model might be improved by using more varied data sets.

7.2 Future work

The Internet of Things (IoT) has made recent strides that have made it possible to offer better healthcare support services. Future COVID-19 epidemiological surveillance can be managed by a cloud-based wireless system, as depicted in Figure. In a few remote facilities, X-rays of the patient's chest can be taken.

Most remote hospitals have X-ray facilities accessible, and the process is quick, affordable, and less intrusive. The institute's ethical committee should give its permission prior to data collection, and imaging data should only be obtained with patient's written agreement. A secure cloud-based server is used to store the acquired data, which includes a unique identifying number for each patient.The X-ray pictures are then evaluated using a cloud-based technology, and the results are delivered to the physicians.

Following a detailed study of the imaging, the doctor gives the patient appropriate recommendations, as well as prescriptions and treatment directions. As a result, even in remote places, medical practitioners and patients can communicate remotely for further treatment.

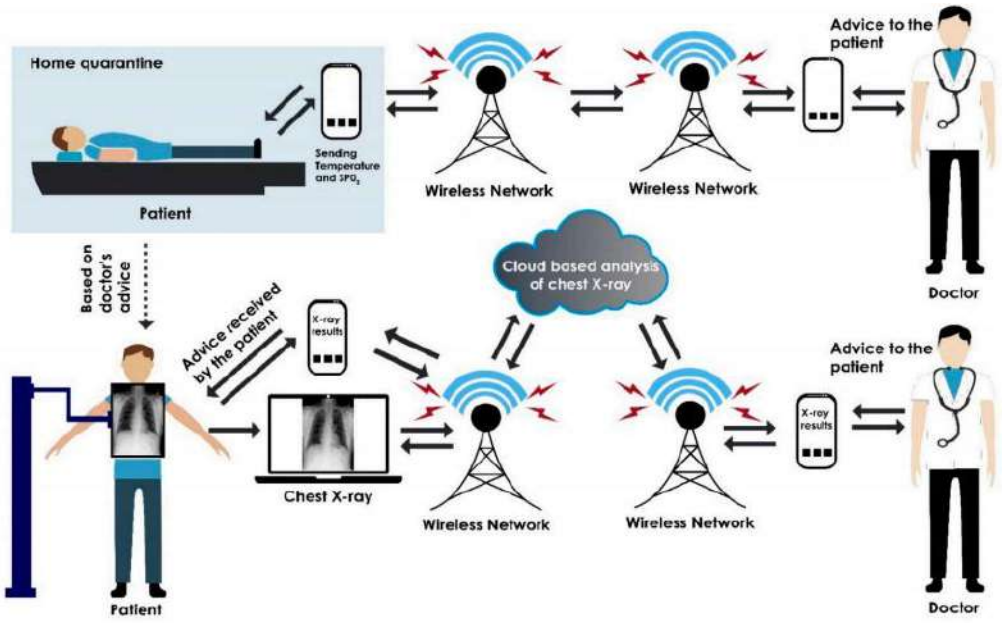


Figure 7.1 - Future concept of our work

Bibliography

1. World Health Organization. (n.d.). Coronavirus disease (covid-19). World Health Organization. Retrieved August 2022, from https://www.who.int/emergencies/diseases/novel-coronavirus-2019?adgroupsurvey=%7Badgroupsurvey%7D&gclid=Cj0KCQjwqc6aBhC4ARIsAN06NmPHVlvyh8KTZPlDnu1W7F4fTLSws7um1ljnHQAyTalkIzda3jEaosaAnOZEALw_wcB

2. World Health Organization. (n.d.). Coronavirus disease (covid-19). World Health Organization.Retrieved August 2022, from https://www.who.int/emergencies/diseases/novel-coronavirus-2019?adgroupsurvey=%7Badgroupsurvey%7D&gclid=Cj0KCQjwqc6aBhC4ARIsAN06NmPHVlvyh8KTZPlDnu1W7F4fTLSws7um1ljnHQAyTalkIzda3jEaosaAnOZEALw_wcB

3. Hu, B., Guo, H., Zhou, P., & Shi, Z.-L. (2020). Characteristics of SARS-COV-2 and COVID-19. Nature Reviews Microbiology, 19(3), 141–154. https://doi.org/10.1038/s41579-020-00459-7

4. Xiao Y, Becerik-Gerber B, Lucas G, Roll SC. Impacts of Working From Home During COVID-19 Pandemic on Physical and Mental Well-Being of Office Workstation Users. J Occup Environ Med. 2021 Mar 1;63(3):181-190. doi: 10.1097/JOM.0000000000002097. PMID: 33234875; PMCID: PMC7934324.

5. Li Z, Chen Q, Feng L, Rodewald L, Xia Y, Yu H, Zhang R, An Z, Yin W, Chen W, Qin Y, Peng Z, Zhang T, Ni D, Cui J, Wang Q, Yang X, Zhang M, Ren X, Wu D, Sun X, Li Y, Zhou L, Qi X, Song T, Gao GF, Feng Z; China CDC COVID-19 Emergency Response Strategy Team. Active case finding with case management: the key to tackling the COVID-19 pandemic. Lancet. 2020 Jul 4;396(10243):63-70. doi: 10.1016/S0140-6736(20)31278-2. Epub 2020 Jun 4. PMID: 32505220; PMCID: PMC7272157.

6. Younes N, Al-Sadeq DW, Al-Jighefee H, Younes S, Al-Jamal O, Daas HI, Yassine HM, Nasrallah GK. Challenges in Laboratory Diagnosis of the Novel Coronavirus SARS-CoV-2. Viruses. 2020 May 26;12(6):582. doi: 10.3390/v12060582. PMID: 32466458; PMCID: PMC7354519.

7. IAEA. (2020, March 27). How is the COVID-19 virus detected using real time RT-PCR? IAEA. Retrieved 2022, from https://www.iaea.org/newscenter/news/how-is-the-covid-19-virus-detected-using-real-time-rt-pcr

8. Banoula, M. (2022, October 11). What is Perceptron: A beginners guide for Perceptron [updated]. Simplilearn.com. Retrieved August 2022, from https://www.simplilearn.com/tutorials/deep-learning tutorial/perceptron#:~:text=Perceptron%20was%20introduced%20by%20Frank,set%20one%20at%20a%20time.
9. Bhardwaj, A. (2020, October 12). What is a Perceptron? – Basics of Neural Networks. Medium. Retrieved August 2022, from https://towardsdatascience.com/what-is-a-perceptron-basics-of-neural-networks-c4cfea20c590
10. Bhadra, R. (2019, September 5). What is a neural network? Medium. Retrieved August 2022, from https://towardsdatascience.com/what-is-a-neural-network-a02b3c2fe3fa
11. Sharma, S. (2021, July 4). Activation functions in neural networks. Medium. Retrieved August 2022, from https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6
12. Géron, A. (n.d.). Neural networks and deep learning. O'Reilly Online Learning. Retrieved August 2022, from https://www.oreilly.com/library/view/neural-networks-and/9781492037354/ch01.html
13. Yamashita, R., Nishio, M., Do, R.K.G. et al. Convolutional neural networks: an overview and application in radiology. Insights Imaging 9, 611–629 (2018). https://doi.org/10.1007/s13244-018-0639-9
14. Team, G. L. (2022, July 19). AlexNet: The first CNN to win Image net. Great Learning Blog: Free Resources what Matters to shape your Career! Retrieved May 2022, from https://www.mygreatlearning.com/blog/alexnet-the-first-cnn-to-win-image-net/

15. Ahn, Jungmo & Park, Jaeyeon & Park, Donghwan & Paek, Jeongyeup & Ko, Jeonggil. (2018). Convolutional neural network-based classification system design with compressed wireless sensor network images. PLOS ONE. 13. e0196251. 10.1371/journal.pone.0196251.
16. Alake, R. (2021, November 3). Deep learning: Googlenet explaine. Medium. Retrieved August 2022, from https://towardsdatascience.com/deep-learning-googlenet-explained-de8861c82765
17. Li, Baoqi & He, Yuyao. (2018). An Improved ResNet Based on the Adjustable Shortcut Connections. IEEE Access. PP. 1-1. 10.1109/ACCESS.2018.2814605.
18. Singh, R. (2022, October 4). Recent advances in Modern Computer Vision. Medium. Retrieved July 2022, from https://towardsdatascience.com/recent-advances-in-modern-computer-vision-56801edab980
19. Elhamraoui, Z. (2020, May 16). INCEPTIONRESNETV2 simple introduction. Medium. Retrieved June 2022, from https://medium.com/@zahraelhamraoui1997/inceptionresnetv2-simple-introduction-9a2000edcdb6
20. G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 4700–4708.
21. Tsang, S.-H. (2021, July 27). Review: Nasnet-neural architecture search network (image classification). Medium. Retrieved July 2022, from https://sh-tsang.medium.com/review-nasnet-neural-architecture-search-network-image-classification-23139ea0425d
22. Architecture of the xception deep CNN model - researchgate. (n.d.). Retrieved July 2022, from https://www.researchgate.net/figure/Architecture-of-the-Xception-deep-CNN-model_fig2_351371226
23. Hanafi, H., Pranolo, A. and Mao, Y., 2021. CAE-COVIDX: automatic covid-19 disease detection based on x-ray images using enhanced deep convolutional and autoencoder. International Journal of Advances in Intelligent Informatics, 7(1), p.49.
24. Wu, T., Tang, C., Xu, M., Hong, N. and Lei, Z., 2021. ULNet for the detection of coronavirus (COVID-19) from chest X-ray images. Computers in Biology and Medicine, 137, p.104834.

25. Jia, G., Lam, H. and Xu, Y., 2021. Classification of COVID-19 chest X-Ray and CT images using a type of dynamic CNN modification method. Computers in Biology and Medicine, 134, p.104425.
26. Khasawneh, N., Fraiwan, M., Fraiwan, L., Khassawneh, B. and Ibnian, A., 2021. Detection of COVID-19 from Chest X-ray Images Using Deep Convolutional Neural Networks. Sensors, 21(17), p.5940.
27. Reshi, A., Rustam, F., Mehmood, A., Alhossan, A., Alrabiah, Z., Ahmad, A., Alsuwailem, H. and Choi, G., 2021. An Efficient CNN Model for COVID-19 Disease Detection Based on X-Ray Image Classification. Complexity, 2021, pp.1-12.
28. Kundu, R., Singh, P., Mirjalili, S. and Sarkar, R., 2021. COVID-19 detection from lung CT-Scans using a fuzzy integral-based CNN ensemble. Computers in Biology and Medicine, 138, p.104895.
29. Sun, J., Li, X., Tang, C., Wang, S. and Zhang, Y., 2021. MFBCNNC: Momentum factor biogeography convolutional neural network for COVID-19 detection via chest X-ray images. Knowledge-Based Systems, 232, p.107494.
30. Chandra, T., Verma, K., Singh, B., Jain, D. and Netam, S., 2021. Coronavirus disease (COVID-19) detection in Chest X-Ray images using majority voting based classifier ensemble. Expert Systems with Applications, 165, p.113909.
31. Hussain, E., Hasan, M., Rahman, M., Lee, I., Tamanna, T. and Parvez, M., 2021. CoroDet: A deep learning based classification for COVID-19 detection using chest X-ray images. Chaos, Solitons & Fractals, 142, p.110495.
32. Pandit, M., Banday, S., Naaz, R. and Chishti, M., 2021. Automatic detection of COVID-19 from chest radiographs using deep learning. Radiography, 27(2), pp.483-489.
33. Marques, G., Agarwal, D. and de la Torre Díez, I., 2020. Automated medical diagnosis of COVID-19 through EfficientNet convolutional neural network. Applied Soft Computing, 96, p.106691.
34. Turkoglu, M., 2021. COVID-19 Detection System Using Chest CT Images and Multiple Kernels-Extreme Learning Machine Based on Deep Neural Network. IRBM, 42(4), pp.207-214.
35. Iraji, M., Feizi-Derakhshi, M. and Tanha, J., 2021. COVID-19 Detection Using Deep Convolutional Neural Networks and Binary Differential Algorithm-Based Feature Selection from X-Ray Images. Complexity, 2021, pp.1-10.

36. Ter-Sarkisov, A., 2022. Lightweight Model for the Prediction of COVID-19 Through the Detection and Segmentation of Lesions in Chest CT Scans.
37. Jain, R., Gupta, M., Taneja, S. and Hemanth, D., 2020. Deep learning based detection and analysis of COVID-19 on chest X-ray images. Applied Intelligence, 51(3), pp.1690-1700.
38. Jain, G., Mittal, D., Thakur, D. and Mittal, M., 2020. A deep learning approach to detect Covid-19 coronavirus with X-Ray images. Biocybernetics and Biomedical Engineering, 40(4), pp.1391-1405.
39. Narin, A., Kaya, C. and Pamuk, Z., 2021. Automatic detection of coronavirus disease (COVID-19) using X-ray images and deep convolutional neural networks. Pattern Analysis and Applications, 24(3), pp.1207-1220.
40. Hassantabar, S., Ahmadi, M. and Sharifi, A., 2020. Diagnosis and detection of infected tissue of COVID-19 patients based on lung x-ray image using convolutional neural network approaches. Chaos, Solitons & Fractals, 140, p.110170.
41. Mahmud, T., Rahman, M. and Fattah, S., 2020. CovXNet: A multi-dilation convolutional neural network for automatic COVID-19 and other pneumonia detection from chest X-ray images with transferable multi-receptive feature optimization. Computers in Biology and Medicine, 122, p.103869.
42. Ieeexplore.ieee.org. 2022. CovFrameNet: An Enhanced Deep Learning Framework for COVID-19 Detection. [online] Available at: <https://ieeexplore.ieee.org/document/9440437>
43. Hemdan, E.E.-D.; Shouman, M.A.; Karar, M.E. Covidx-net: A framework of deep learning classifiers to diagnose COVID-19 inX-ray images. arXiv 2020, arXiv:2003.11055. Sarki R, Ahmed K, Wang H, Zhang Y, Wang K (2022) Automated detection of COVID-19 through convolutional neural network using chest x-ray images.
44. Rajaraman, S., Siegelman, J., Alderson, P. O., Folio, L. S., Folio, L. R., & Antani, S. K. (2020). Iteratively Pruned Deep Learning Ensembles for COVID-19 Detection in Chest X-Rays. IEEE Access, 8, 115041–115050. https://doi.org/10.1109/access.2020.3003810
45. Karar, M. E., Hemdan, E. E. D., & Shouman, M. A. (2020). Cascaded deep learning classifiers for computer-aided diagnosis of COVID-19 and pneumonia diseases in X-ray scans. Complex & Intelligent Systems, 7(1), 235–247. https://doi.org/10.1007/s40747-020-00199-4.