Directed Research Report

CSE498R

Department of ECE

Detect Covid-19 Patients from X-Ray Images Using Deep Learning

Submitted By:

Name	ID
Rahat Islam Akash	1631239042
Shahin Arman Apu	1711280642
Khandakar Faizul Haque Onim	1631707042

Supervisor

Mr. Ahmed Fahmin (AFN1)

Professor,

ELECTRICAL AND COMPUTER ENGINEERING

NORTH SOUTH UNIVERSITY

Summer 2021

Agreement Form

We take great pleasure in submitting our senior design project report on "**Detect Covid-19 Patients from X-Ray Images Using Deep Learning**". This report is prepared as a requirement of the Directed Research CSE498R. This course requires a high level of self-directed learning. This learning requires students to read, conduct research, and complete written examinations, reports, projects, research papers, portfolios or similar assignments that are designed to measure competency in the stated objectives. We would like to request you to accept this report as a partial fulfillment of Bachelor of Science degree under Electrical and Computer Engineering Department of North South University.

Declared by:

…………………………………………
Name: Rahat Islam Akash
ID: 1631239042

…………………………………………
Name: Shahin Arman Apu
ID: 1711280642

…………………………………………
Name: Khandakar Faizul Haque Onim
ID: 1631707042

Approved by:

……………………

Supervisor
Mr. Ahmed Fahmin
Professor
Department of Electrical and Computer Engineering North South University, Dhaka, Bangladesh

....
Dr. Mohammad Rezaul Bari
Chair, Department of Electrical and Computer Engineering
North South University, Dhaka, Bangladesh

ACKNOWLEDGEMENT

First of all, we wish to express sincere gratitude to our supervisor, Mr. Ahmed Fahmin (AFN1), for his insightful comments, helpful information and practical advice in our research work. His immense knowledge, profound experience and professional expertise in senior design project has helped us to complete this research till this, successfully. Without his support and guidance, the work of this research would not have been possible.

We would like to express our gratitude towards our friends & family for their kind co-operation and encouragement which helped us in completion of this research work.

ABSTRACT

The number of COVID-19 Cases across the globe is still increasing at high-speed, every single day. Two of the most common symptoms of the novel coronavirus include pneumonia or lung infection. In the current times where technology is prevalent, usage of various techniques of artificial intelligence in the field of medical science enables us to get accurate solutions of problems in the field much quicker. The COVID-19 Virus can be detected through a special imaging technique called Computerized Technology (CT). CT Scans, X-Rays, Magnetic Resonance Imaging (MRI), PET/CT, and Lung Ultrasounds are commonly used to detect diseases using AI imaging techniques. In this research, we have demonstrated how to detect the COVID-19 Virus through X-Ray images by using multiple Convolutional Neural Network Models and Transfer Learning.

Index terms: Covid-19, Computer Tomography, Imaging, Artificial Intelligence

Table of Contents

Chapter 1

Introduction

1.1 Introduction

The very first detected case of the novel coronavirus can be traced back to December 2019 in the Wuhan Province in China. This was a new variant of a coronavirus that was occurring in humans for the very first time. The little virus soon transformed into an epidemic and before the world even realized, it had turned into a full-fledged pandemic. As of now, over 219 million cases of COVID-19 have been identified across the world, with over 4.55 million deaths from the identified cases.

The coronavirus is a large family of homologous viruses that can cause a wide range of serious illnesses – many of the epidemics and pandemics we have seen before fall under this category. This range of serious illnesses start from something as simple as the common cold, and can mutilate into graver diseases such as the Middle East Respiratory Syndrome (MARS) and the Severe Acute Respiratory Syndrome (SARS). However, only 7% of those identified are expected to fall seriously ill from the disease – they become prone to lung failure, septic shocks, paralysis, and in worst case scenarios, death [1]. 14% of those identified usually show the severe symptoms, the most prevalent of which are breathing problems. Most of those, around 80%, who are diagnosed with the virus show mild symptoms such as pneumonia symptoms along with fever and coughing. Older people and especially those with existing illnesses such as asthma, diabetes, heart diseases, high blood pressure, etc. are most likely to be most adversely affected by the virus.

Recent statistics show that the increasing number of cases each day could lead up to the world taking at least a few years to get rid of the virus and revive from it. To control the rapid spread of COVID-19 from one person to another, screenings of people who show symptoms to detect the virus is the most important step. Many countries, especially those with a very large population, are facing difficulties in testing people due to a limited number of testing kits. Even the most developed of nations such as USA, UK, and Italy have struggled to deal with the overwhelming number of cases – The increasing demand for Intensive Care Units for those showing the worst of symptoms such as pneumonia, extreme intense respiratory condition, and multi-organ failure have left them clutching at straws to save lives [2]. The most common technique to detect the COVID-19 Virus is the RT-PCR test, which is expensive and a very lengthy process. The main problem with this technique, keeping aside the cost factor, is how much time it takes to get back the results – If the patient does not maintain isolation and other necessary precautions, they are most likely to spread it to other people as well as have their lives in danger.

Airspace Opacities, Ground-Glass Opacity (GGO), and later consolidation are few of the primary signs in the lungs which act as the base of detecting COVID-19 primarily in the patients [3]. Due to the limited number of test kits, alternative solutions have been looked deeply into – Several studies have found that CT Scans of Chest X-Ray images can also be effective in early detection of COVID-19 in patients who show symptoms [4] [5]. X-Ray Machines are generally used to identify irritants in a body such as cracks, bone disengagements, lung contamination, pneumonia, and other tumours; CT Scans are modified and advanced versions of X-Ray Machines that magnifies into the more delicate structures of dynamic body parts, and gives us clearer pictures of intricate inward tissues and organs [5]. Given the advancement of CT Scans, X-Rays are comparatively the quicker, simpler, more cost-effective and less critical method [4]. As previously mentioned, pneumonia is one of the most prevalent signs of the presence of COVID-19 in a human body, and the disease opens up holes in a person's lungs much like the SARS disease, giving the lungs a "honeycomb-like" appearance. Chest X-Rays and CT Scans are both used to diagnose pneumonia, and so, they can be seen as the best method for early screening of COVID-19 [4]. Several studies so far have actually gone on to recommend using X-Rays over RT-PCR tests due to the limited availability of the latter in many countries.

As of now, Artificial Intelligence (AI) has become an imminent part of detecting and solving many medical-based issues. The emergence and advancement of Artificial Intelligence (AI) in the medical science field has enabled imaging-based analysis of many diseases, especially COVID-19. Further research in context of the COVID-19 situation has found that deep neural techniques are a better, more accurate and more sensitive method of detecting the virus as opposed to RT-PCR tests. In this study, we have applied multiple convolutional neural network models including VGG16, VGG19, MobileNetV2, Resnet50, Inception V3, and Xception to distinguish between COVID-19, Pneumonia, and normal patients. So, six different models were tested on the X-Ray image datasets, and then the accuracy and precision of the models were analyzed, too.

1.2 Background and Motivation

The novel coronavirus is an increasingly infectious diseases and has affected millions of people across the globe – The adversity and depth of the disaster caused by this virus has had the World

Health Organization (WHO) declaring it a pandemic. The best way to bring this pandemic under control is to detect those who are affected and put them in isolation until the virus leaves their system and can no longer be transferred onto someone else. To do this, those who show symptoms are needed to be brought under testing, but the RT-PCR test that has been designed to detect the virus is time-consuming, which leaves those showing symptoms at a higher chance of spreading the disease onto others. And RT-PCR tests are still quite expensive, meaning many people belonging to the lower-income sections might not be able to afford it in the first place. Deep Learning Techniques have been known to have contributed heavily to analyzing medical images, and show accurate and sensitive results. We will be implementing the deep neural network in two stages to clearly distinguish between the novel coronavirus and other bacterial viruses that lead to the same symptoms, including pneumonia. The two-stage approach has been designed accordingly to implement a computer-aided solution to detect COVID-19 cases in a faster, more systematic, and dependable manner. A study undertaken recently has found that around an impressive 98% accuracy can be achieved in detecting and classifying COVID-19 through this method. And so, we will be following these diverse methods to deliver more accurate and faster results to battle the COVID-19 pandemic more effectively.

1.3 Literature Review

This chapter is a review of related literature focusing on detecting COVID-19 automatically from X-Ray images, using CNN and transfer learning. Mondhur et al. conducted a study which aims to examine the effectiveness of Transfer learning to detect COVID-19 automatically from Xray images, using convolutional neural networks. They used pre-trained models with some modification in the top of these models. Their dataset used in training is taken from multiple sources of X-rays and Kaggle. The number of images in this dataset involves 274 COVID-19 cases, 380 viral pneumonia cases, and 380 healthy. Before being passed to pre-trained model the images resized to the size of $224 \times 224 \times 3$. After that they divided the dataset into 70% for the training set and 30% for the validation set. Transfer learning technique was applied to resolve inadequate data and the time for preparation. The proposed deep transfer learning models were trained separately using the Python Programming Language. All experiments were conducted with Tesla K80 GPU graphics card on Google Collaboratory with Windows 10 operating system. We used tenfold-cross-validation approach to assess the performance of our models. A confusion matrix was also introduced to analyze whether the prediction is consistent with the actual results. They used some different types of models. When they used ResNet50 CNN model along with SVM got 95.38% of accuracy on the 2-class problem. They used COVIDX-Net and got 90% accuracy. For DarkCOVIDNet, they got 98.08% of accuracy on the 2-class and 87.02% of accuracy on the 3-class. For VGG16 got 99.78% of accuracy on the 2-class and 99.57% of accuracy on the 3-class. They mentioned that available information at this stage is inadequate as it was mainly based on results obtained from selected regions. Therefore, more detailed and extensive work on more dataset is required to detect COVID-19 automatically from Xray images, using convolutional neural networks.

Ankita et al. conducted another study that aims to examine the effectiveness of Transfer learning to detect COVID-19 automatically from Xray images, using CNN models. They used pre-trained models with A total of 2271 chest X-ray images (895x1024x3 pixels) were obtained from Clinic Diagnostic Lab, Mumbai India. The image was converted from RGB to gray. The resolution of images was reduced to (64x64x1). A total of 1071 images of all three types (i.e., Normal, TB, Pneumonia), split into 70 % training images and the remaining 30 % test images. The training dataset consisted of 388 normal images, 500 pneumonia images, and 303 TB images. They were doing a 3-way classification. They have used various pre-trained network to get their expected accuracy. At the time of using VGG16 they flatten the output of the Max pooling layer and then have a dense layer with 3 neurons. The learning rate was kept at 0.001 with Adam as the optimizer. They trained the model for 12 epochs and got accuracy of 95.9%. Then they used Deep CNN and got accuracy of 98%. They got 98.9% of accuracy by using DenseNet-161 model. They mentioned that their proposed model will help yield faster and accurate results and will be cost-effective as compared to the conventional RT-PCR method.

Sohaib et al. conducted a research to investigate the efficacy of Transfer learning to detect COVID-19 automatically from Xray images using coevolutionary neural networks. They used pre-trained models with some modification at the top of these models. For the training and evaluation of the COV-19Net they have collected chest radiographic images from 3 open-source databases: 1) GitHub open-source dataset by Chowdhury et al. 2) Dataset from GitHub by Cohen et al. 3) Positive radiographic images (CXR and CT) were carefully chosen from Italian Society of Medical and Interventional Radiology (SIRM) COVID-19 DATABASE. The network was equipped using their dataset by random weight initialization. The inputs were 64x64x3 in format. Adam optimization is being used and the hyper-parameters used were a constant learning rate of 0.001, a batch size of 32, steps per epoch 100 and a number of epochs of 20. Further we implemented data augmentation due to shortage of resources to train our model and the types of augmentation used were: rescaling, rotating, zooming, vertical flip, horizontal flip, intensity shift and shear intensity shift (shear angle in counter-clockwise direction). The proposed VOC-19Net model was applied via the Keras platform using the TensorFlow backend. Because they got accuracy of 99.45% using this model.

Kiran et al. proposed deep augmented model to solve the problem. Their dataset was of 2 types, X-Ray dataset and CT Scan dataset. X-Ray dataset that contains 536 images of COVID-19 and 536 images non- COVID- 19 including augmented images, the CT Scan dataset contains 2760 images of COVID-19 and 2760 images of non-COVID-19 individuals including augmented images. The ratio of train and test samples was 70:30. At first, they accept the colored input images from the data set. Convert the image into grayscale. Then they down sample the images to 50×50 dimension from their original size. In CNN based deep learning model they choose *layer sizes* = [32*;* 64*;* 128], *dense layers* = [0*;* 1*;* 2], and *conv_layers* = [1*;* 2*;* 3]. Convolution with a 3×3 filter size is applied. Activation function RELU is used after convolutional layer. Then 2×2 filter size is applied for Max Pooling. In Step 5 if *conv_layers*− 1>0 Flatten the matrix. Activation function Sigmoid is used for classification of test image into COVID and non- COVID classes. In training stage, the standard first-order stochastic gradient descent optimizer with a batch size of 32, maximum epochs 30 and binary cross entropy-based loss function are used. The random subsampling method is adopted. Whole dataset is divided into a number of ratios like 90:10, 80:20, 70:30 and 60:40 as training and testing samples. Various Evaluation Metrics are calculated such as classification accuracy, loss, area under ROC curve (AUC), precision, recall (Sensitivity), Specificity and F1 score. With this approach, the proposed model exhibits higher classification accuracy around 95.38% and 98.97% for CT scan and X-Ray images respectively. CT scan images with multi-image augmentation achieves sensitivity of 94.78% and specificity of 95.98%, whereas X-Ray images with multi-image augmentation achieves sensitivity of 99.07% and specificity of 98.88%. Due to a smaller number of X-Ray or CT Scan images that are used for training the deep learning models, often exhibit the classification accuracies not up to the mark.

1.4 Proposed solution

The primary purpose of our research is to develop a Deep Learning Technique for the detection of the novel coronavirus which is appropriate for people of all ages and demographics. Our proposed solution is to build a Convolutional Neural Network to detect the existence of the coronavirus in a human body from X-ray Images with maximum accuracy. We will train on different models of Convolutional Neural Network such as VGG16, VGG19, ImageNet, DenseNet, ResNet, etc. on an X-Ray dataset. Then, each model on the dataset will be trained separately to discern the accuracy that each of them bring. After analyzing the accuracy of all six models, we can conclude which model will give us the best results. Our research can help countries where testing kits are limited in numbers and costly. This might even end up motivating people to get tested if they show symptoms as opposed to the RT-PCR tests that are lengthy and expensive, especially for people in countries such as Bangladesh.

CHAPTER 2
Methodology

2.1 Introduction

In this section we are going discuss about methodology of our research. There are also details of some existing platforms, algorithms and techniques related to our research. Then the details of our proposed system are mentioned. There is also a workflow of our research method.

2.2 Required Knowledge and Skills

a. Basics of Machine Learning Algorithms
b. Deep Learning and Imaging techniques
c. Data Cleaning Techniques
d. Technical Terms of Data Science
e. Programming Language (Python)
f. Use of Scientific Computing Platforms (Anaconda, Jupyter Notebook)

2.3 Deep Learning Techniques

A. Transfer Learning

Transfer learning is a technique to use a pre-trained model as a starting point or basis for another model. This is a common and famous machine learning technique which is widely used in deep learning. In deep learning we use a lot of hidden layers with hundreds or thousands of nodes in it. It is too much time consuming and somewhat impossible to train a too much large dataset on this kind of model. We will need a computer or machine with high requirements. Again, to get a better result in deep learning it is needed to input the dataset for number of times for the model to learn. That's why transfer learning technique was introduced. In this technique, a model is trained on a dataset of millions of data and the model is saved with the learned weights. Then we build a model and use that pre-trained model as starting point of the new model. After that, the dataset is trained on the new modified model and this time the model gives better result as the model has already learned on a big dataset.

B. Image Augmentation

In order to perform well, deep networks require comprehensive training data. In order to create a powerful image classifier with very little data preparation, an increase in image is normally important to enhance deep networking efficiency. Image enlargement produces training images artificially by various rendering processes or variations such as spontaneous rotation, moves, shear and rotating, etc.

2.4 Algorithms

A. VGG16: VGG16 is a convolution neural net (CNN) architecture which was used to win ILSVR (ImageNet) competition in 2014. It is considered to be one of the excellent vision model architecture till date. Most unique thing about VGG16 is that instead of having a large number of hyper-parameter they focused on having convolution layers of 3x3 filter with a stride 1 and always used same padding and maxpool layer of 2x2 filter of stride 2. It follows this arrangement of convolution and max pool layers consistently throughout the whole architecture. In the end it has 2 FC (fully connected layers) followed by a SoftMax for output. The 16 in VGG16 refers to it has 16 layers that have weights. This network is a pretty large network and it has about 138 million (approx.) parameters **[6]**.

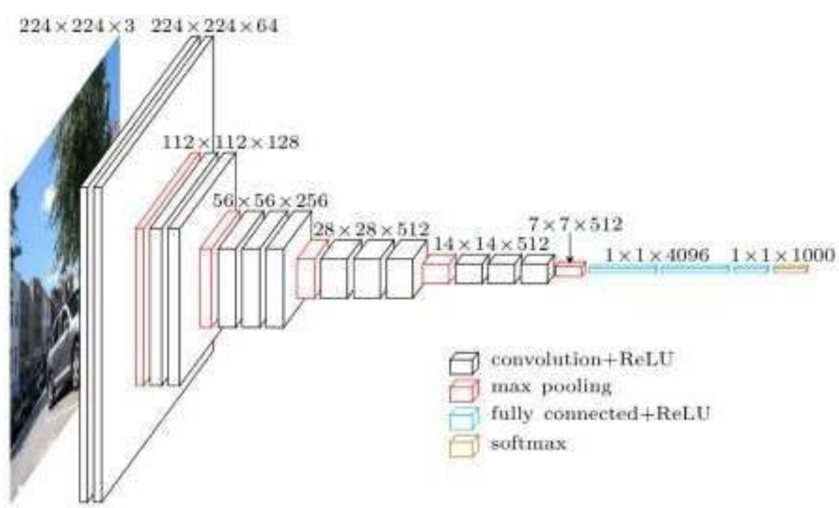


Figure: Architecture of VGG16.

B. VGG19: VGG-19 is a convolutional neural network that is 19 layers deep. You can load a pre- trained version of the network trained on more than a million images from the ImageNet database. The pre-trained network can classify images into 1000 object categories, such as keyboard, mouse, pencil, and many animals. As a result, the network has learned rich feature representations for a wide range of images. The network has an image input size of 224-by-224 **[7]**.

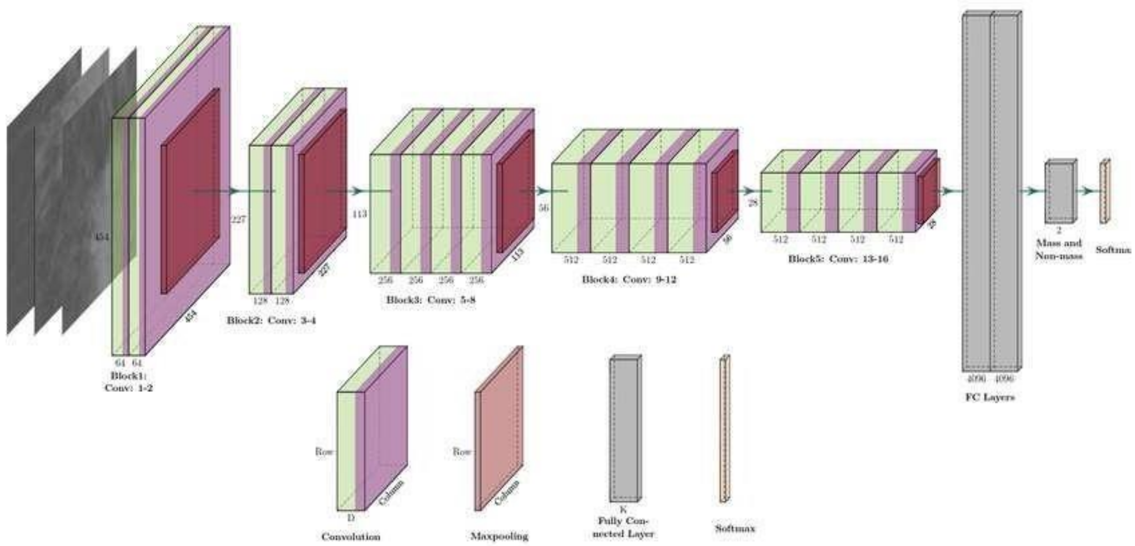


Figure: Architecture of VGG19

C. Xception: Xception is a convolutional neural network that is 71 layers deep. You can load a pre-trained version of the network trained on more than a million images from the ImageNet database. The pre-trained network can classify images into 1000 object categories, such as keyboard, mouse, pencil, and many animals. As a result, the network has learned rich feature representations for a wide range of images. The network has an image input size of 299-by-299 **[8]**.

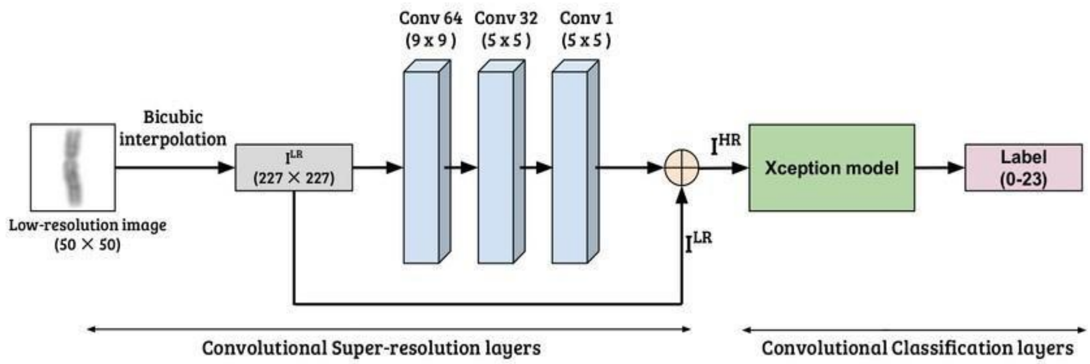


Figure: Architecture of Xception

D. ResNet50: ResNet, short for Residual Networks is a classic neural network used as a backbone for many computer vision tasks. This model was the winner of ImageNet challenge in 2015. The fundamental breakthrough with ResNet was it allowed us to train extremely deep neural networks with 150+layers successfully. Prior to ResNet training very deep neural networks was difficult due to the problem of vanishing gradients **[9].**

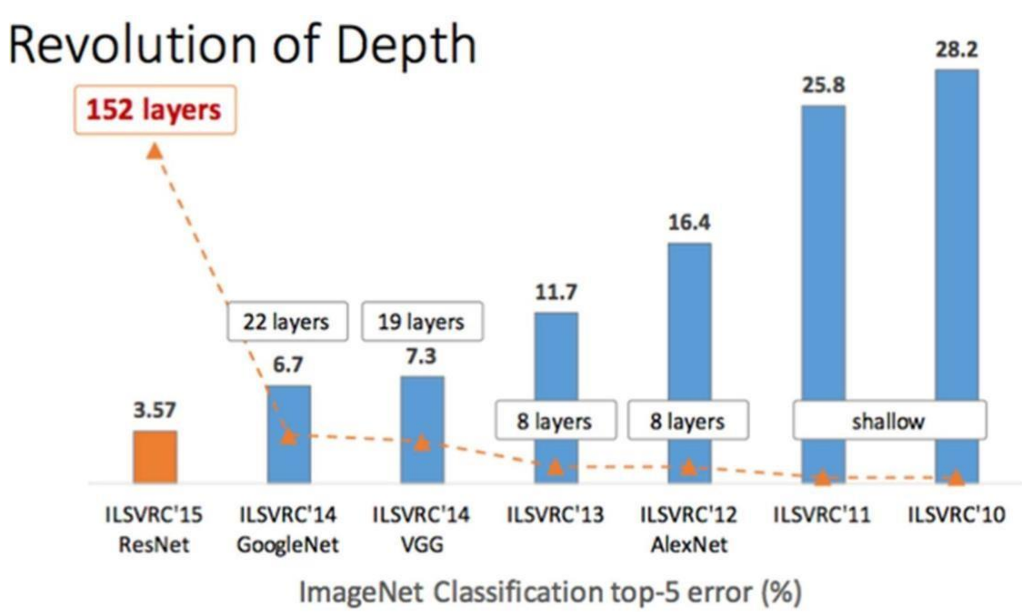


Figure: Architecture of Resnet50

E. MobileNetV2: MobileNetV2, by Google, is briefly reviewed. In the previous version MobileNetV1, Depth wise Separable Convolution is introduced which dramatically reduce the complexity cost and model size of the network, which is suitable to Mobile devices, or any devices with low computational power. In MobileNetV2, a better module is introduced with inverted residual structure. Non-linearities in narrow layers are removed this time. With MobileNetV2 as backbone for feature extraction, state-of-the-art performances are also achieved for object detection and semantic segmentation [10].

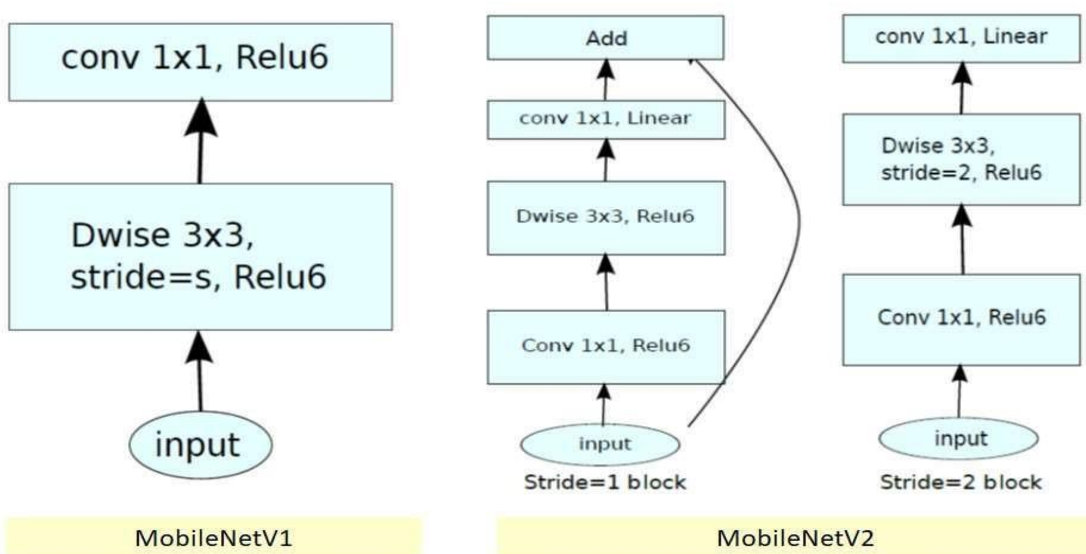


Figure: Architecture of MobileNet

F. Inception V3: In the inception architecture, computational efficiency and fewer parameters are realized. With fewer parameters, 42-layer deep learning network, with similar complexity as VGGNet, can be achieved. With 42 layers, lower error rate is obtained and make it become the 1st Runner Up for image classification in ILSVRC (ImageNet Large Scale Visual Recognition Competition) 2015 **[11]**.

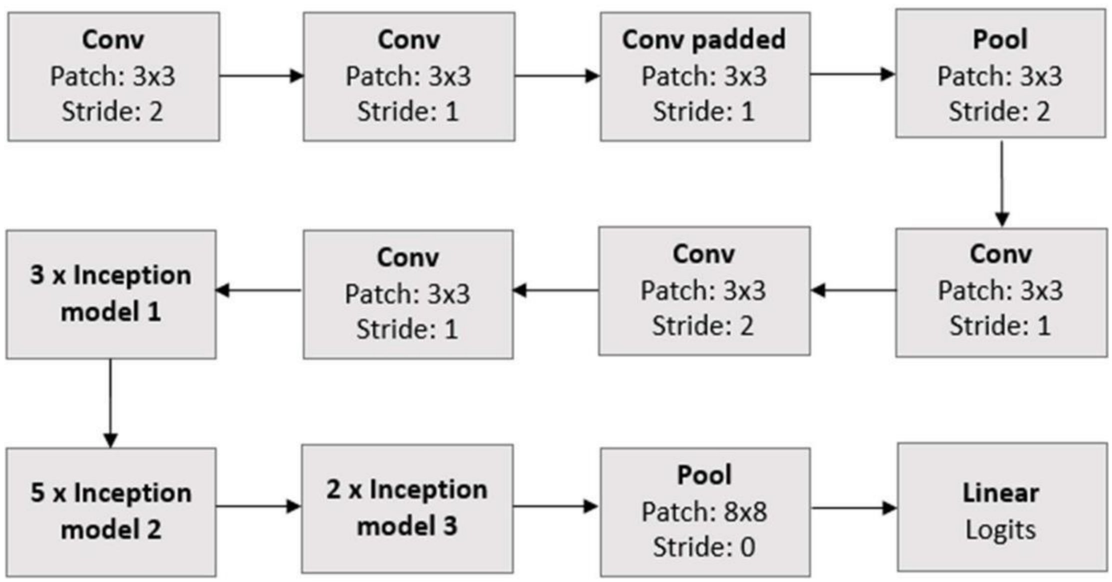


Figure: Architecture of Inception V3

2.5Tools

- **Python:** Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together **[12]**.

- **TensorFlow:** TensorFlow Federated (TFF) is an open-source framework for machine learning and other computations on decentralized data. TFF has been developed to facilitate open research and experimentation with Federated Learning (FL), an approach to machine learning where a shared global model is trained across many participating clients that keep their training data locally. For example, FL has been used to train prediction models for mobile keyboards without uploading sensitive typing data to servers **[13]**

- **GitHub**: GitHub is where over 56 million developers shape the future of software, together. Contribute to the open-source community, manage your Git repositories.

2.6 Software and Hardware Requirements

A. Software:

i) Anaconda

ii) Jupyter Notebook

iii) Nvidia CUDA ToolKit

iv) Microsoft Visual Studio

iv) Browser i.e., Mozilla Firefox, Google Chrome

B. Hardware:

i) NVIDIA GTX-830M or higher

ii) 8 GB RAM

iii) Solid State Drive (SSD)

2.7 Research Workflow

We followed a basic research workflow model while conducting the research. The stages are described below -

A. Define Research Problem

The initial stage of research is to define the research problem. This stage is related to finding the problem that we want to solve and explain various aspects of the problem. It is important to know the problem in detail to solve the problem in efficient way. We researched about the problem and found it as an important problem to solve in this pandemic period.

B. Review Previous Findings

In this stage, we need to find the previous finding of the problem and see what others have done so far. This is also an important step as our solution might have already been proposed or tested. We need to know others finding so that we do not repeat the same thing which is time consuming. Moreover, we can learn from others findings and tweak their solutions to get the better solution. In this stage, we can read papers, journals, online articles to review the previous findings. In a research, published research papers and journals are more acceptable to review. We collected almost 50 research papers from various resources and analyzed those to find the already available solutions of our problem domain.

C. Review Concepts and Theories

After reviewing previous findings, we need to review the new concepts and theories that others have used. This stage is helpful to understand others findings in details. Moreover, we cannot tweak any solution without knowing about it in details. In our research, we had to review some deep learning concepts like transfer learning, image classification etc.

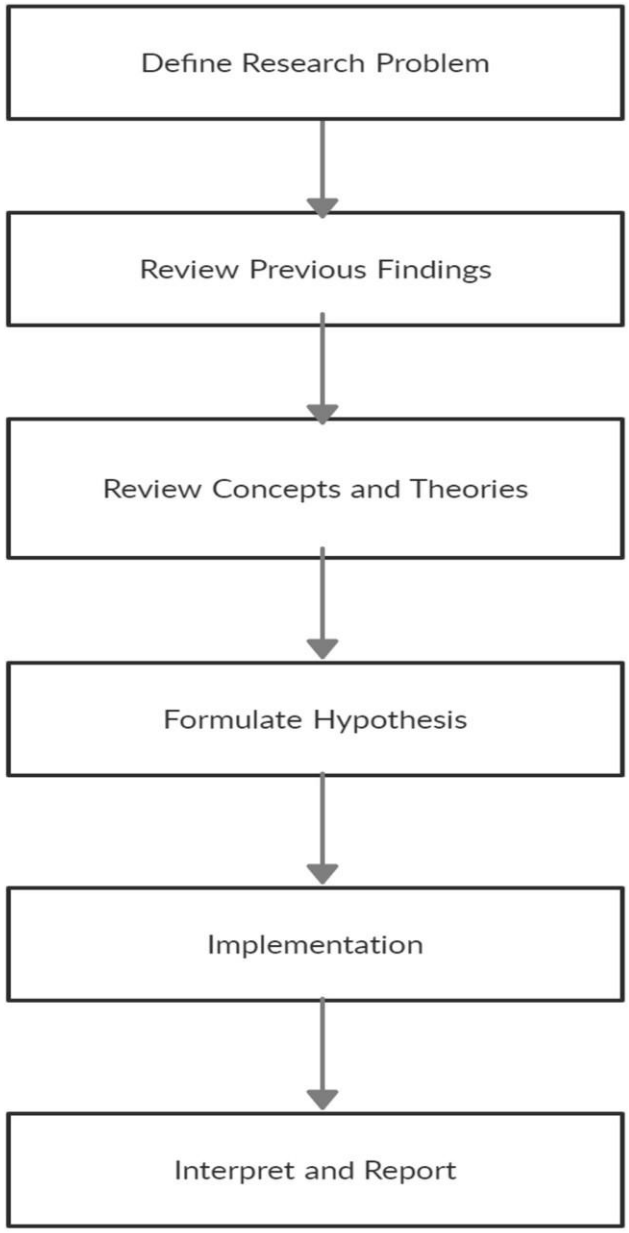


Figure: Research Workflow

D. Formulate Hypothesis

The hypothesis is a guess or proposed explication made as a starting point for further research based on insufficient proof. We need to propose that there is a solution to the research before starting the research and conduct research on it to proof it.

E. Implementation

Implementation is not a single stage. It actually consists of different stages. As we are trying to conduct a research using deep learning approach our stages will be different from other research areas. These steps will be further discussed later in this section.

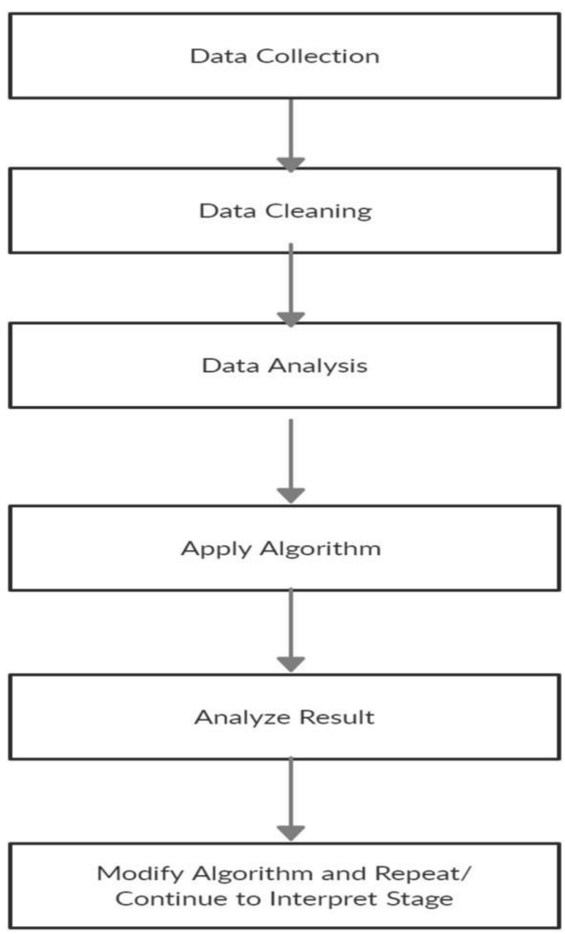


Figure: Implementation Stages

F. Interpret and Report

In this stage we need to interpret the result and make a detailed report about it with the comparisons with existing solutions.

2.8 Research Implementation Workflow

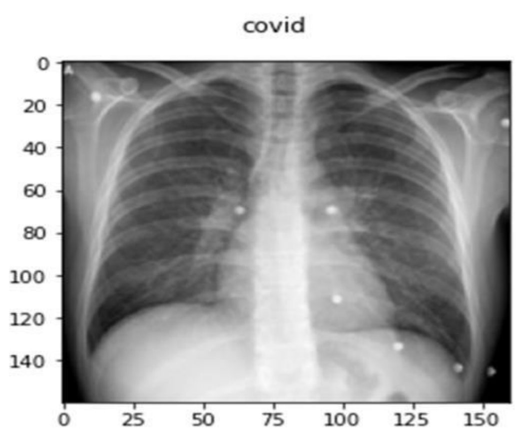
Research Implementation stage can be divided into some more stages. As we are using deep learning to solve the problem, we need to follow common deep learning implementation stages. The stages are Data Collection, Data Cleaning, Data Analysis, Apply Algorithm, Analyze Result and Modify Algorithm or continue to interpret stage. These stages are further described below:

A. Data Collection

In this stage, we need to find a proper dataset to conduct our research. The dataset needs to be from a valid source and be acceptable by researchers. Also, there should be enough data in the dataset to be acceptable. There are a lot of dataset available on the internet that are only for testing purpose. We need to avoid those for research purpose and only select the dataset with enough data. We selected a dataset that is a latest dataset with the most number of data available on this research area. We collected our dataset from- Mendeley Data - Curated Dataset for COVID-19 Posterior- Anterior Chest Radiography Images (X-Rays). This is a combined curated dataset of COVID-19 Chest X-ray images obtained by collating 15 publicly available datasets as listed under the references section. The present dataset contains 1281 COVID-19 X-Rays, 3270 Normal X-Rays, 1656 viral-pneumonia X-Rays, and 3001 bacterial-pneumonia X-Rays. We only used COVID-19 X- Rays, Normal X-Rays and Viral-Pneumonia X-Rays for our research purpose.

B. Data Cleaning

The dataset we collect might have raw data or might have invalid or poor data. That's why we need to apply data cleaning techniques before applying algorithm on it. Data cleaning techniques includes removing incomplete data, filling up missing data etc. As we are dealing with image data, we need to clean the images such that unclear images are removed or brightened in such a way that they become clear. We can also use different image cleaning techniques available as python functions and test which technique work better. In our research, we just resized our images to a same size as this is a crucial stage for deep learning techniques. Moreover, our dataset was already preprocessed by the owner of the dataset. According to the source of our dataset, Images with Noise, Pixelated, Compressed, Medical Implants, washed out image, Side View, CT (sliced) image, Aspect Ratio distortion / Cropped / Zoomed, Rotated Images, Images with annotations were detected and removed from the dataset.



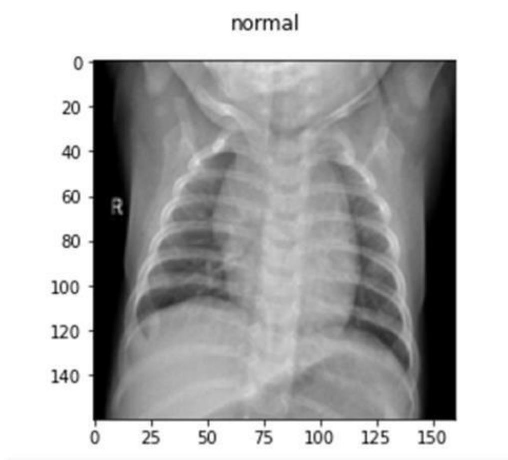


Figure: Visualization of Data

C. Data Analysis

In this stage, we need to visualize our data and see if the data are okay for the next stage. As we dealing with image data, we visualized X-ray images of different types to see if the images were clearly visible and good enough to apply deep learning algorithms. In our research, we used Image Augmentation technique to create a modified version of tensor images from our dataset. We also split our dataset into training set and validation set.

```
target_size = (224,224)

train_datagen = ImageDataGenerator(rescale=1/255,validation_split=0.2)
train_data_gen = train_datagen.flow_from_directory(train_path,
                                                   target_size=target_size,
                                                   batch_size=32,
                                                   class_mode='categorical')
```

Figure: Image Augmentation

Then we visualized the images to see if they are perfectly stored as tensor images.

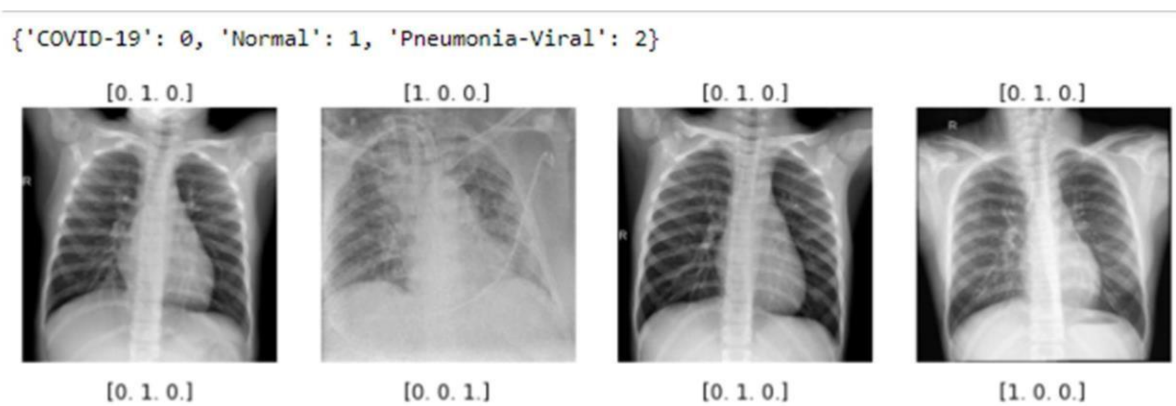


Figure: Analysis of dataset after data cleaning

D. Apply Algorithm

In this stage, we apply our deep learning algorithm on our dataset to classify the images according to the labels we've given below. We can use different algorithms or some techniques like transfer learning in this stage. In our research we used the both techniques.

At first step, we created a basic CNN network model to see how our dataset is working. Our basic CNN architecture looks like this -

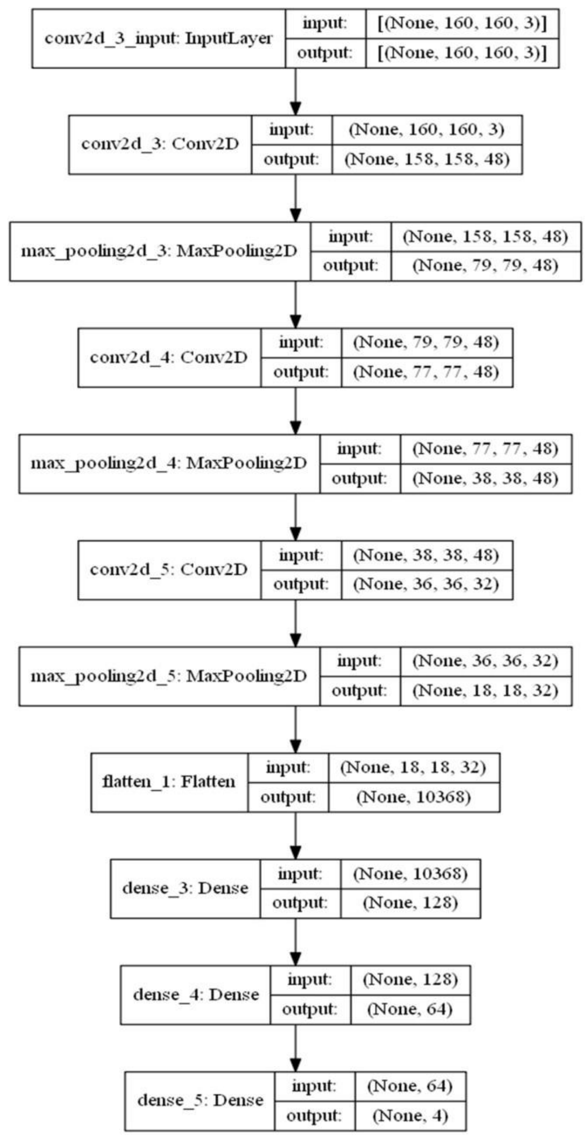


Figure: Basic CNN Architecture

Then we used the pre-trained models on our dataset. In CNN, we used all the 4 classes. In this stage, we used all the classes too.

Our first transfer learning model was VGG16. We resized the images by 150*150*3. We added a convolution base layer after VGG model. We also added a Flatten layer, a Dense layer with relu activation function and another dense layer with softmax activation function. Our optimizer was adam optimizer and we run it for 25 epochs.

Then we modified our VGG16 model for 3 classes and changed our batch size to 16 and number of epochs to 20. Our code for VGG16 model was like this-

```
from keras import models
from keras import layers
from keras.applications import VGG16
from keras import optimizers
from keras.layers.core import Flatten, Dense, Dropout, Lambda
conv_base = VGG16(weights='imagenet',include_top=False,input_shape=(224, 224, 3))
conv_base.trainable = False
model = models.Sequential()
model.add(conv_base)
model.add(layers.Flatten())
model.add(layers.Dense(256, activation='relu'))
model.add(layers.Dense(3, activation='softmax'))
model.compile(loss='categorical_crossentropy',
optimizer=optimizers.Adam(lr=INIT_LR),
metrics=['acc'])
```

Figure: VGG-16 Model

After that we applied VGG19 model. Our input image size was 224*224*3. We added a flatten layer, a dropout layer and a dense layer after the pre-trained model. The code looks like this -

```
vggModel = VGG19(weights="imagenet", include_top=False,
    input_tensor=Input(shape=(224, 224, 3)))

outputs = vggModel.output
outputs = Flatten(name="flatten")(outputs)
outputs = Dropout(0.5)(outputs)
outputs = Dense(3, activation="softmax")(outputs)

model = Model(inputs=vggModel.input, outputs=outputs)

for layer in vggModel.layers:
    layer.trainable = False

model.compile(
        loss='categorical_crossentropy',
        optimizer='adam',
        metrics=['accuracy']
)
```

Figure: VGG-19 Model

Then we used Inception V3 model. Again we added a flatten layer, a dropout layer, a dense layer and an activation function layer to the pre-trained model.

Our Inception V3 model summary is shown below-

Model: "sequential"

Layer (type)	Output Shape	Param #
inception_v3 (Functional)	(None, 5, 5, 2048)	21802784
flatten (Flatten)	(None, 51200)	0
dropout (Dropout)	(None, 51200)	0
dense (Dense)	(None, 3)	153603
activation_94 (Activation)	(None, 3)	0

Total params: 21,956,387
Trainable params: 153,603
Non-trainable params: 21,802,784

Figure: Inception V3 Model Summary

After that we used Resnet50 model. Our model summary is shown below-

Model: "sequential_3"

Layer (type)	Output Shape	Param #
resnet50 (Functional)	(None, 7, 7, 2048)	23587712
flatten_3 (Flatten)	(None, 100352)	0
dropout_3 (Dropout)	(None, 100352)	0
dense_6 (Dense)	(None, 3)	301059
activation_3 (Activation)	(None, 3)	0

Total params: 23,888,771
Trainable params: 301,059
Non-trainable params: 23,587,712

Figure: Resnet50 Model Summary

Then Xception model. The model summary is shown below-

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
xception (Functional)        (None, 8, 8, 2048)        20861480
_________________________________________________________________
flatten (Flatten)            (None, 131072)            0
_________________________________________________________________
dense (Dense)                (None, 512)               67109376
_________________________________________________________________
activation (Activation)      (None, 512)               0
_________________________________________________________________
dropout (Dropout)            (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 3)                 1539
_________________________________________________________________
activation_1 (Activation)    (None, 3)                 0
=================================================================
Total params: 87,972,395
Trainable params: 67,110,915
Non-trainable params: 20,861,480
_________________________________________________________________
```

Figure: Xception Model Summary

And finally, MobileNet V2 Model. The model summary is given below-

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
mobilenet_1.00_224 (Function (None, 7, 7, 1024)        3228864
_________________________________________________________________
flatten (Flatten)            (None, 50176)             0
_________________________________________________________________
dropout (Dropout)            (None, 50176)             0
_________________________________________________________________
dense (Dense)                (None, 3)                 150531
_________________________________________________________________
activation (Activation)      (None, 3)                 0
=================================================================
Total params: 3,379,395
Trainable params: 150,531
Non-trainable params: 3,228,864
```

Figure: MobileNet Model Summary

E. Analyze Result

In this step, we analyze the result and see if our algorithm model is working fine on the dataset. We used different result analysis techniques like accuracy, precision etc. to analyze our result. Our analysis of the result will be discussed in Results and Discussion section.

F. Modify or Continue to Next Stage

We have to modify our algorithm or code as there are thousands of parameters in deep learning algorithms. Also changing the data processing techniques can also change the result of the solution. In our research we had to modify our model several times to find a satisfactory result.

CHAPTER 3
RESEARCH IMPACT

3.1 Environmental Impacts of the Research

The Advancement of Artificial Intelligence have also had a positive impact on our environment. Through this research work we have been able to detect Covid-19 with CNN model which will have a good impact on the environment. The problem with testing with RT-PCR is that it requires a lot more material to test a patient. In particular, if the cotton that is used as a sample is thrown into the environment, there is a possibility that the environment will be polluted and as a result, other people may also be infected with corona. Since we will detect Covid-19 through X-ray imaging with CNN model, there is no possibility of environmental pollution. The X-Ray images can also be used further for detecting others disease such as pneumonia and lung infection. As a result, patient does not have to repeat the same test again which reduces usage of material and that will be beneficial for the environment.

3.2 Economic Impacts of the Research

Our proposed model can create a significant economic impact on a developing country like Bangladesh. RT-PCR machines are not set up in all parts of the country. Purchasing RT-PCR machine and collect samples requires a lot of money. Since our proposed model can successfully detect a coronavirus with hundred percent accuracy and X-ray machines are widely available in all parts of the country, no new set up is required and this will save a lot of money.

3.3 Social Impacts of the Research

The current RT-PCR method of detecting coronavirus is extremely expensive and time consuming. Also the process of collecting samples is very painful. As a result, many people may be afraid to do their testing through this process. But in order to prevent the corona virus, all the members of the society must be tested. Our proposed model can play a significant role in solving this problem. Since it is very simple process and less expensive so it will be very useful for people of all classes in the society. They will be more interested in testing through this process rather than RT-PCR test.

CHAPTER 4
Research Summary

4.1 Results and Discussion

In this part, we’ll discuss the results of the CNN and transfer learning techniques on our dataset. We’ll go through the algorithms one by one and then compare it at the last.

Basic CNN Model:

After running 20 epochs we got training accuracy of 82.55% and validation accuracy of 84.09%. Our validation loss decreased in each stage. At first epoch, it was 0.6193. After 20 epochs, it was reduced to 0.4092 which is not the best, but a good improvement. We came to the conclusion from this training that our dataset was working and we can do major improvements on it using transfer learning models.

```
Epoch 18/20
1074/1074 [==============================] - 704s 655ms/step - loss: 0.4015 - accuracy: 0.8288 - val_loss: 0.3996 - val_accurac
y: 0.8366
Epoch 19/20
1074/1074 [==============================] - 824s 767ms/step - loss: 0.3947 - accuracy: 0.8302 - val_loss: 0.4086 - val_accurac
y: 0.8460
Epoch 20/20
1074/1074 [==============================] - 849s 791ms/step - loss: 0.4016 - accuracy: 0.8255 - val_loss: 0.4092 - val_accurac
y: 0.8409
```

Figure: CNN Model Accuracy and Loss

VGG16 (With 4 Classes):

After running 25 epochs we got training accuracy of 88.07 % and validation accuracy of 91.98%. It was not that much improvement of accuracy from basic CNN model. But there was significant improvement in validation loss. Before it was 0.4092. But in this model, it reduced to 0.2196 which was surely a major improvement. At this time, we researched about the problem a bit and understood that using viral pneumonia and bacteria pneumonia as different classes might be the problem. To make it clear, we modified our model for 3 classes and trained it again.

```
Epoch 18/25
912/912 [==============================] - 363s 398ms/step - loss: 0.3130 - acc: 0.8702 - val_loss: 0.2371 - val_acc: 0.9132
Epoch 19/25
912/912 [==============================] - 355s 389ms/step - loss: 0.3053 - acc: 0.8729 - val_loss: 0.2525 - val_acc: 0.9110
Epoch 20/25
912/912 [==============================] - 414s 454ms/step - loss: 0.3113 - acc: 0.8704 - val_loss: 0.2430 - val_acc: 0.9132
Epoch 21/25
912/912 [==============================] - 525s 575ms/step - loss: 0.2967 - acc: 0.8777 - val_loss: 0.2419 - val_acc: 0.9088
Epoch 22/25
912/912 [==============================] - 367s 403ms/step - loss: 0.2917 - acc: 0.8814 - val_loss: 0.2750 - val_acc: 0.8814
Epoch 23/25
912/912 [==============================] - 363s 398ms/step - loss: 0.2902 - acc: 0.8776 - val_loss: 0.2331 - val_acc: 0.9165
Epoch 24/25
912/912 [==============================] - 432s 474ms/step - loss: 0.2945 - acc: 0.8757 - val_loss: 0.2787 - val_acc: 0.8825
Epoch 25/25
912/912 [==============================] - 418s 459ms/step - loss: 0.2792 - acc: 0.8807 - val_loss: 0.2196 - val_acc: 0.9198
```

Figure: VGG16(4 Classes) Model Accuracy and Loss

VGG16 (With 3 Classes)

Using VGG16 for 3 classes we noticed a major improvement on our result. We got training accuracy of 97.71% and validation accuracy of 99.07%. We got validation loss of 0.0313. Surely, this gave us a far better result and our assumption was correct.

```
Epoch 15/20
236/236 [==============================] - 181s 764ms/step - loss: 0.0741 - acc: 0.9715 - val_loss: 0.0847 - val_acc: 0.9707
Epoch 16/20
236/236 [==============================] - 182s 770ms/step - loss: 0.0868 - acc: 0.9695 - val_loss: 0.0713 - val_acc: 0.9827
Epoch 17/20
236/236 [==============================] - 181s 767ms/step - loss: 0.0835 - acc: 0.9674 - val_loss: 0.0250 - val_acc: 0.9934
Epoch 18/20
236/236 [==============================] - 181s 767ms/step - loss: 0.0611 - acc: 0.9775 - val_loss: 0.0432 - val_acc: 0.9827
Epoch 19/20
236/236 [==============================] - 182s 771ms/step - loss: 0.0673 - acc: 0.9722 - val_loss: 0.0299 - val_acc: 0.9867
Epoch 20/20
236/236 [==============================] - 182s 768ms/step - loss: 0.0668 - acc: 0.9771 - val_loss: 0.0313 - val_acc: 0.9907
```

Figure: VGG16 Model Accuracy and Loss

VGG19

Using VGG19, the performance did not improve, rather it declined. We got training accuracy of 93.91% and a validation accuracy of 97.87%. Validation loss also increased from 0.0313 to 0.0987.

```
Epoch 17/20
236/236 [==============================] - 177s 752ms/step - loss: 0.3058 - accuracy: 0.9273 - val_loss: 0.0803 - val_accuracy:
0.9801
Epoch 18/20
236/236 [==============================] - 174s 735ms/step - loss: 0.2476 - accuracy: 0.9408 - val_loss: 0.0556 - val_accuracy:
0.9854
Epoch 19/20
236/236 [==============================] - 180s 761ms/step - loss: 0.2185 - accuracy: 0.9518 - val_loss: 0.0861 - val_accuracy:
0.9761
Epoch 20/20
236/236 [==============================] - 179s 759ms/step - loss: 0.2636 - accuracy: 0.9391 - val_loss: 0.0987 - val_accuracy:
0.9787
```

Figure: VGG19 Model Accuracy and Loss

Inception V3

In this case, we got training accuracy of 98.56% and validation accuracy of 95.27%. Validation loss was 0.6370. The changes of accuracy and loss are shown in the graphs.

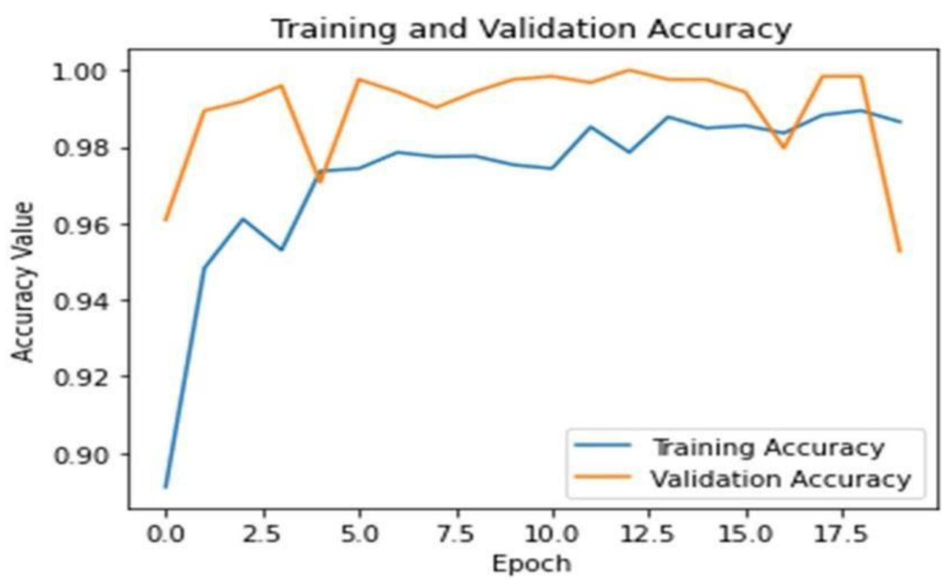


Figure: Inception V3 Training and Validation Accuracy Graph

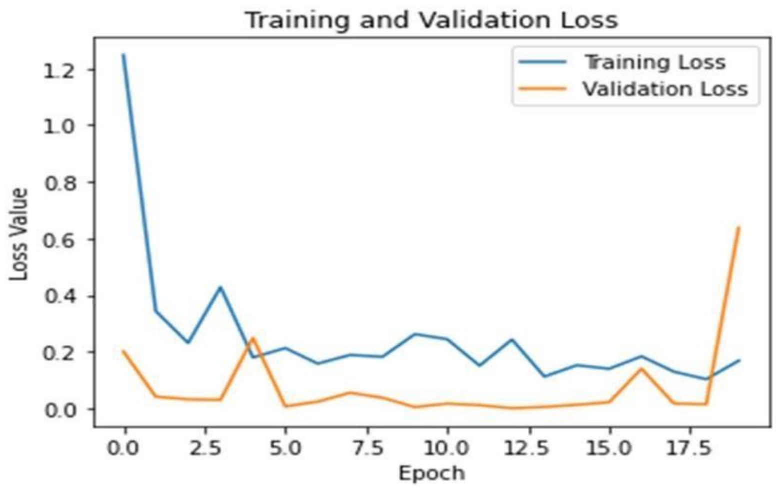


Figure: Inception V3 Training and Validation Loss Graph

Confusion Matrix:

PREDICTED → / ACTUAL ↓	COVID-19	Normal	Viral Pneumonia
COVID-19	199	8	44
Normal	0	643	6
Viral Pneumonia	0	0	326

Table: Inception V3 Confusion Matrix

From the table, it is clear that a total of 58 images were misclassified and 44 images of viral pneumonia are being classified as Covid-19.

Classification Report also indicates the same.

	precision	recall	f1-score	support
COVID-19	1.00	0.79	0.88	251
Normal	0.99	0.99	0.99	649
Pneumonia-Viral	0.87	1.00	0.93	326
accuracy			0.95	1226
macro avg	0.95	0.93	0.93	1226
weighted avg	0.96	0.95	0.95	1226

Figure: Inception V3 Classification Report

Xception

In this model we got training accuracy of 97.30% and validation accuracy of 99.51%. The graphs of accuracy and validation are shown below-

Figure: Xception Training and Validation Accuracy Graph

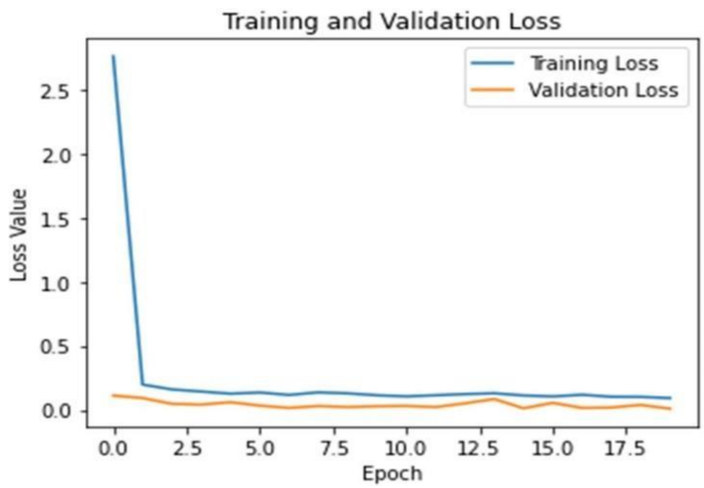


Figure: Xception Training and Validation Loss Graph

Confusion Matrix:

PREDICTED → / ACTUAL ↓	COVID-19	Normal	Viral Pneumonia
COVID-19	250	1	0
Normal	0	649	0
Viral Pneumonia	1	4	326

Table: Xception Confusion Matrix

From the confusion matrix, we can see that the model misclassified only 6 images and 1 Covid-19 patient was classified as normal and 1 Viral Pneumonia patient was classified as Covid-19 patient.

Classification Report:

	precision	recall	f1-score	support
COVID-19	1.00	1.00	1.00	251
Normal	0.99	1.00	1.00	649
Pneumonia-Viral	1.00	0.98	0.99	326
accuracy			1.00	1226
macro avg	1.00	0.99	0.99	1226
weighted avg	1.00	1.00	1.00	1226

Figure: Xception Classification Report

Resnet50:

In this model, we got training accuracy of 87.71% and validation accuracy of 95.41%.

The accuracy and loss graphs are shown below-

Figure: Resnet50 Training and Validation Accuracy Graph

Figure: Resnet50 Training and Validation Loss Graph

From the graph, we can see that the loss and accuracy was not stable throughout the training.

Confusion Matrix:

PREDICTED → / ACTUAL ↓	COVID-19	Norma l	Viral Pneumonia
COVID-19	239	4	8
Normal	1	644	4
Viral Pneumonia	18	16	292

Table: Resnet50 Confusion Matrix

From the confusion matrix, it is clear that the model performed poorly as it classified 4 Covid-19 patients as normal which is a major risk of the solution.

Classification Report:

	precision	recall	f1-score	support
COVID-19	0.93	0.95	0.94	251
Normal	0.97	0.99	0.98	649
Pneumonia-Viral	0.96	0.90	0.93	326
accuracy			0.96	1226
macro avg	0.95	0.95	0.95	1226
weighted avg	0.96	0.96	0.96	1226

Figure: Resnet50 Classification Report

MobileNet V2

In this model, we got training accuracy of 99.74% and validation accuracy of 100%.

The validation loss was close to zero. These are shown in the graphs are like this-

Figure: MobileNet Training and Validation Accuracy Graph

Figure: MobileNet Training and Validation Loss Graph

Confusion Matrix:

PREDICTED → / ACTUAL ↓	COVID-19	Norma l	Viral Pneumonia
COVID-19	251	0	0
Normal	0	649	0
Viral Pneumonia	0	0	326

Table: MobileNet Confusion Matrix

From the confusion matrix, we can see that it classified all the classes correctly which was our research target.

Classification Report:

	precision	recall	f1-score	support
COVID-19	1.00	1.00	1.00	251
Normal	1.00	1.00	1.00	649
Pneumonia-Viral	1.00	1.00	1.00	326
accuracy			1.00	1226
macro avg	1.00	1.00	1.00	1226
weighted avg	1.00	1.00	1.00	1226

Figure: MobileNet Classification Report

4.2 Results Comparison

In our research we noticed better performance from some algorithms that was expected. But some algorithm gave poor result which worked a little bit better for others. We can understand how our selected algorithms worked, if we look at the accuracy and loss comparison table of our algorithms:

Algorithm	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss
VGG16	0.9771	0.9907	0.0668	0.0313
VGG19	0.9391	0.9787	0.2636	0.0987
Inception V3	0.9856	0.9527	0.1601	0.6370
Xception	**0.9730**	**0.9951**	**0.0787**	**0.0098**
Resnet50	0.8771	0.9584	0.5243	0.1342
MobileNet	**0.9974**	**1.0000**	**0.0263**	**0.0001**

Table: Accuracy and Loss Comparison Table

From the table, we can see that MobileNet performed best to classify and detect the images. The second-best model was Xception. The worst model was ResNet50 as it had highest training loss and it could not classify the images accurately.

We can understand our result more accurately if we compare using precision by classes.

Algorithm	COVID-19	Normal	Viral Pneumonia
VGG16	1.00	0.99	0.97
VGG19	0.98	0.97	0.97
Inception V3	1.00	0.99	0.87
Xception	1.00	0.99	1.00
Resnet50	0.93	0.97	0.96
MobileNet	**1.00**	**1.00**	**1.00**

Table: Precision Comparison Table by Class

Here we can see that most of the models could differentiate and detect Covid-19 images. But some could not differentiate well among Covid-19 and Viral-Pneumonia images.

4.3 Conclusion

As of now, intelligent medical imaging continues to play an imminent role in detecting coronavirus infections across the world, leading to preventing the spread of the virus. In our proposed model, we have elaborately discussed the details of the different methods of deep learning technology, and how each of the models we evaluated worked. We applied those models on the X-Ray dataset, and achieved a wide range of different results for each of the models. After further analysis and comparison of the results we found, we have concluded the best model in terms of COVID-19 detection. Between the models VGG16, VGG19, Xception, MobileNet V2, ResNet50, and Inception, MobileNetV2 has given us the best accuracy (99.74). So it is safe to conclude that a combination of Artificial Intelligence usage and Imaging Techniques can be efficient and effective for overcoming many medical limitations and hence, solving them. In the future, we would like to make the dataset even wider and larger, as a larger set of data will give us more accurate validation when studying the combination of Artificial Intelligence and Imaging Techniques. Furthermore, there are also opportunities to conduct more detailed and in-depth researches on how our model would work on the different strains and variants of the novel coronavirus.

APPENDIX

REFERENCES

[1] E. Mahase, "Coronavirus: covid-19 has killed more people than sars and mers combined, despite lower case fatality rate,," 2020.

[2] E. H. M. Z. P. P. P. S. a. A. G. R. Janice Hopkins Tanne, "Covid-19: how doctors and healthcare systems are tackling coronavirus worldwide," *bmj,* 2020.

[3] "Chronic airspace disease: Review of the causes and key computed tomography findings," *World journal of radiology*, no. 10.4329/wjr.v12.i4.29, 2020.

[4] C. K. Z. P. Ali Narin, "Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks.," no. arXiv:2003.10849, 2020.

[5] Envision Radiology, "Envision Radiology," 2017. [Online]. Available: https://www.envrad.com/difference- between-x-ray-ct-scan-and-mri/.

[6] R. Thakur, "Towards data science," 6 Aug 2019. [Online]. Available: https://towardsdatascience.com/step- by-step-vgg16-implementation-in-keras-for-beginners-a833c686ae6c.

[7] MathWorks, "MathWorks," 2020. [Online]. Available: https://www.mathworks.com/help/deeplearning/ref/vgg19.html.

[8] MatWorks, "MatWorks," 2020. [Online]. Available: https://www.mathworks.com/help/deeplearning/ref/xception.html.

[9] P. Dwivedi, "towards data science," 4 January 2019. [Online]. Available: https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras-446d7ff84d33.

[10] S.-H. Tsang, "towards data science," 19 May 2019. [Online]. Available: https://towardsdatascience.com/review-mobilenetv2-light-weight-model-image-classification- 8febb490e61c.

[11] S.-H. Tsang, "Medium," 10 September 2018. [Online]. Available: https://sh-tsang.medium.com/review- inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c.

[12] Python, "Python," [Online]. Available: https://www.python.org/.

[13] TensorFlow, "TensorFlow," [Online]. Available: https://www.tensorflow.org/federated.