**COVID-19 RECOGNITION FROM CHEST X-RAY IMAGES USING DEEP LEARNING**

##### A PROJECT REPORT

###### ***Submitted by***

##### KEERTHANA V [211418104119]

##### LAKSHMIPRIYA B [211418104137]

**MADHUMITHA CT** [**211418104142**]

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

**in**

**COMPUTER SCIENCE AND ENGINEERING**

## PANIMALAR ENGINEERING COLLEGE

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

**MAY 2022**

# PANIMALAR ENGINEERING COLLEGE

###### **(An Autonomous Institution, Affiliated to Anna University, Chennai)**

**BONAFIDE CERTIFICATE**

Certified that this project report **“COVID-19 RECOGNITION FROM CHEST XRAY IMAGES USING DEEP LEARNING”** is the bonafide work of “**KEERTHANA V (211418104119), LAKSHMIPRIYA B(211418104137), MADHUMITHA CT (211418104142)** “who carried out the project work under my supervision.

**SIGNATURE SIGNATURE**

**Dr.S.MURUGAVALLI,M.E.,Ph.D., Dr.L.JABA SHEELA, M.E., Ph.D.,**

**HEAD OF THE DEPARTMENT SUPERVISOR**

**PROFESSOR**

DEPARTMENT OF CSE, DEPARTMENT OF CSE,

PANIMALAR ENGINEERING COLLEGE, PANIMALAR ENGINEERING COLLEGE,

NASARATHPETTAI, NASARATHPETTAI,

POONAMALLEE, POONAMALLEE,

CHENNAI-600 123. CHENNAI-600 123.

Certified that the above mentioned students were examined in Anna University project viva-voice held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**DECLARATION BY THE STUDENT**

We Keerthana V (211418104119), Lakshmipriya B (211418104137), Madhumitha CT (211418104142) hereby declare that this project report titled “Covid-19 Recognition From Chest X-Ray Images Using Deep Learning”, under the guidance of Dr.L.Jaba Sheela, M.CA., M.E., Ph.D., is the original work done by us and we have not plagiarized or submitted to any other degree in any other university by us.

**1.KEERTHANA V**

**2.LAKSHMIPRIYA B**

**3.MADHUMITHA CT**

**ACKNOWLEDGEMENT**

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P. CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our Directors **Tmt.C.VIJAYARAJESWARI**, **Dr.C.SAKTHI KUMAR,M.E.,Ph.D** and **Dr.SARANYASREE SAKTHI KUMAR B.E.,M.B.A.,Ph.D.,** for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr. S.MURUGAVALLI , M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank my **Project Guide** **Dr.L.JABA SHEELA, M.CA., M.E., Ph.D.,** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

**KEERTHANA V**

**LAKSHMIPRIYA B**

**MADHUMITHA CT**

**ABSTRACT**

Coronavirus disease (COVID-19) is a pandemic virus that has caused thousands of deaths and infected millions of people worldwide. Effective screening of infected patients is a key step in the fight against COVID-19 . Due to the increase in cases of COVID-19, test kits available in the hospital is minimal. In order to prevent the spread of the disease, an automated detection system is needed . To prevent the further spread of this disease, it is necessary to detect positive cases as early as possible. The use of convolutional neural networks (CNN) in conjunction with medical imaging can be helpful in accurate detecting this disease. Current methods of diagnosis for COVID-19 use RT-PCR, which is a less-sensitive and expensive diagnostic test that requires highly trained medical personnel. The use of X-rays in COVID-19 diagnostics is a quick, accessible, and reliable alternative. Using pre-trained deep-learning algorithms and maximizing the detection accuracy, we propose a robust approach for the automatic detection of COVID-19 from digital chest X-ray images. Networks were trained to classify two different schemes: covid and non-covid. The proposed model has achieved an accuracy level of 97.19%, 95.42%, 96.23% and 93.87% based on Xception, VGG16, Resnet and Inception. COVID-19 diagnosis can be significantly accelerated and more accurate with this computer-aided diagnostic tool due to the high accuracy of this tool.

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **Table No** | **Table description** | **Page No** |
| 5.1 | Training model of covid-19 recognition using chest x-ray | 24 |
|  |  |  |
| 7.1 | Precision, Recall and F1-score of Xception | 61 |
|  |  |  |

 7.2 Precision, Recall and

F1-score of ResNet 50 61

7.3 Precision, Recall and

F1-score of Inception 61

7.4 Precision, Recall and

F1-score of VGG 16 61

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **FIG NO.** | **FIGURE DESCRIPTION** | **PAGE NO.** |
| 4.1 | Er diagram | 14 |
| 4.2 | Data flow diagram | 15 |
| 4.7 | Use case diagram | 17 |
| 4.8 | Sequence diagram | 18 |
| 4.9 | Collaboration diagram | 19 |
| 4.10 | Activity diagram | 20 |
| 5.1 | Architecture diagram | 21 |
| 5.2 | Vgg 16 model | 27 |
| 5.3 | Xception model | 28 |
| 8.1 | Inception training model | 62 |
| 8.2 | Xception training model | 63 |
| 8.3 | Resnet training model | 63 |
| 8.4 | Vgg training model | 63 |
| A.1 | Home page | 65 |
| A.2 | Upload image | 65 |
| A.3 | View uploaded image | 66 |
| A.4 | Result page | 66 |
| A.5 | Received report via email | 67 |
|  |  |  |

**LIST OF SYMBOLS**

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | Actor |  | It aggregates several classes into a single classes. |
| 2. | Relation (extends) | extends | Extends relationship is used when one use case is similar to another use case but does a bit more. |
| 3. | Communication |  | Communication between various use cases. |
| 4. | State | State | State of the process. |
| 5. | Initial State |  | Initial state of the object |
| 6. | Final state |  | Final state of the object |
| 7. | Use case | Uses case | Interact ion between the system and external environment. |

|  |  |  |  |
| --- | --- | --- | --- |
| 8. | Decision box |  | Represents decision making process from a constraint |

|  |  |  |  |
| --- | --- | --- | --- |
| 9. | Message | Message | Represents the message exchanged. |
| 10. | Data Process/State |  | A circle in DFD represents a state or process which has been triggered due to  some event or action. |
| 11. | External entity |  | Represents external entities such as keyboard, sensors, etc. |

**ABBREVIATIONS**

|  |  |
| --- | --- |
| **CNN** | Convolutional Neural Network |
| **RT-PCR** | Reverse Transcription polymerase Chain Reaction |
| **VGG 16** | Visual Geometry Group |
| **CXR** | Chest X-Ray |
| **PPE** | Personal Protective Equipment |
| **RESNET** | Residual Neural Network |
| **DHE** | Dynamic Histogram Equalization |
| **CT** | Computed Tomography |
| **ML** | Machine Learning |
| **DL** | Deep Learning |

**TABLE OF CONTENTS**

| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
| --- | --- | --- |
|  | **ABSTRACT** | v |
|  | **LIST OF TABLES** | vi |
|  | **LIST OF FIGURES** | vii |
|  | **LIST OF SYMBOLS, ABBREVIATIONS** | viii |
| **1.** | **INTRODUCTION** |  |
|  | 1.1 Overview | 1 |
|  | 1.2 Problem Definition | 2 |
| **2.** | **LITERATURE SURVEY** | 3 |
| **3.** | **SYSTEM ANALYSIS** |  |
|  | 2.1 Existing System | 8 |
|  | 2.2 Proposed system | 8 |
|  | 2.3 Feasibility Study | 9 |
|  | 2.4 Hardware Environment | 11 |
|  | 2.5 Software Environment | 12 |
| **4.** | **SYSTEM DESIGN** |  |
|  | 4.1. ER diagram | 13 |
|  | 4.2 Data Flow Diagram | 15 |
|  | 4.3 UML Diagrams | 17 |
| **5.** | **SYSTEM ARCHITECTURE** |  |
|  | 5.1 System overview | 21 |
|  | 5.2 Module design specifications | 22 |
|  | 5.2 Algorithms | 25 |
| **6.** | **SYSTEM IMPLEMENTATION** |  |
|  | 6.1 Coding | 29 |
| **7.** | **PERFORMANCE ANALYSIS** |  |
|  | 7.1 Performance analysis | 60 |
|  | 7.2 Results & Discussions | 62 |
| **8.** | **CONCLUSION** |  |
|  | 8.1 Conclusion and Future Enhancements | 64 |
|  | **APPENDICES** |  |
|  | A.1 Sample Screens | 65 |
|  | **REFERENCES** |  |
|  |  |  |

**CHAPTER 1**

**INTRODUCTION**

* 1. **OVERVIEW**

A reverse transcription-polymerase chain reaction (RT-PCR) is the standard method for detecting the presence of SARS-CoV-2 in respiratory secretions or nasopharynx samples. Although RT-PCR is said to be highly specific, its sensitivity can range from 60 to 70%. Therefore, false negatives are a serious concern especially in the beginning of the disease process. Diagnostic imaging tests have been critical in detecting and treating these patients and have been used to diagnose, determine the severity, guide treatment, and determine whether the treatments are working. According to most scientific and radiological associations, imaging tests should not be used for screening. Convolutional Neural Networks (CNNs) can be very useful for image classification on large datasets. COVID-19 classification and detection have been extensively explored using CNN , reaching a far better level of accuracy than previous methods. The CNN report shows that healthcare professionals can pinpoint patients at higher risk of disease development with remarkable accuracy. It can be used for binary classification to multi-class classification **.** CNNs have already demonstrated promising results in detecting intricate structures in high-dimensional datasets using multi-layer function representations. Because of its usefulness, availability, and low cost, chest X-ray is generally the first-line imaging test in patients with suspected or confirmed COVID-19, though it is less sensitive than computed tomography (CT).

**1.2 PROBLEM DEFINITION**

RT-PCR is the current screening method for COVID-19. COVID-19 is diagnosed by many practitioners/doctors using this method. There is a problem with this method as it is very time-consuming, and it takes days to weeks to obtain the results. Clinics and hospitals with fewer resources will find it very difficult to use this method. CXR methods are much cheaper and easier to obtain than PCR techniques. They are also readily available compared with PCR techniques. As part of this project, a new model has been developed for the detection of COVID-19 in raw chest x-ray images. X-ray images are recommended for diagnosing COVID-19 since they can be obtained quickly and at a low cost at nearby hospitals or clinics. COVID-19 was detected using chest X-ray images using a diagnosis model based on VGG16. Based on augmented data, the model has been able to achieve an F-measure of 96%, enabling it to detect COVID-19 rapidly and reliably. It is noticeable that coastal areas have a higher prevalence of the disease than other non-coastal areas. This means coastal cities need extra attention and care.

**CHAPTER 2**

**LITERATURE SURVEY**

**1.Wassim Zouch, Dhouha Sagga, Amira Echtioui, Rafik Khemakhem, Mohamed Ghorbel, Chokri Mhiri & Ahmed Ben Hamida, "Detection of COVID-19 from CT and Chest X-ray Images Using Deep Learning Models",Springer,2022. [25]**

In this paper they have used two databases one is for CT images and the other one is for x-ray images. These images are taken from the open-source git hub repository shared by Dr.Jkooy. These images are then pre-processed to a size of 224\*224 pixels. The resized images are then trained by using the Deep learning models. The images are trained for about 50 epochs using DL models. In order to identify chest X-ray and CT images as normal or affected by COVID-19 they have proposed two DL models: ResNet50 and VGG19.The results show that for VGG19, the training accuracy rate reached 99.35 and 88.87% with a loss of the training reduced to 0.1, for the CT and chest X-ray images respectively. While for the ResNet50 models, the training precision attained 96.77 and 76.32% with a training loss varying between 0.1 and 0.2, for CT and chest X-ray images respectively.

**2.S.V.Kogilavani , J. Prabhu, R. Sandhiya, M. Sandeep Kumar, UmaShankar Subramaniam, Alagar Karthick ,M. Muhibbullah and Sharmila Banu Sheik Imam,"COVID-19 Detection Based on Lung Ct Scan Using Deep Learning Techniques",Hindawi, 2022.[19]**

CNN is a deep learning method that takes an input image and assigns weight to different objects in the image, allowing it to distinguish between them. To classify the data, deep learning architectures such as VGG16, DenseNet, MobileNet, Xception, Efficient Net, and NASNet are used. Each model has been trained over 50 epochs. The input is CT scan images from a chest tomography. To filter the size of all input samples, the image filtering preprocessing technique is used. There are several methods for assessing the performance of a model. The measures used to estimate chest CT scan images are accuracy, precision, recall, and F-score.

**3.Amir Rehman, Muhammad Azhar Iqbal, Huanlai Xing and Irfan Ahmed, ” COVID-19 Detection Empowered with Machine Learning and Deep Learning Techniques: A Systematic Review”, applied sciences, 2021.[4]**

CT images has been proposed as an essential substitute tool for COVID-19 detection. Here the CT images of patients are collected from China National Center for Bioinformation (CNCB) and COVIDNet-CT repository from github. They used transfer learning method to improve the performance of CNN on covid 19 testing using CT images. The Grad-Cam visualization techniques is established to explore and understand the covid affected area in the CT image. To address the issue, researchers must develop appropriate ML/DL techniques capable of producing better results with small datasets.

**4.Wentao Zhao , Wei Jiang1 & Xinguo Qiu ,"Deep learning for COVID‑19 detection based on CT images", scientific reports,2021.[24]**

## CT images have been proposed as an important alternative tool for COVID-19 detection. The CT images of patients are obtained from the China National Center for Bioinformation (CNCB) and the COVIDNet-CT github repository. They used the transfer learning method to improve CNN's performance on covid 19 testing using CT images. Grad-Cam visualization techniques are used to explore and comprehend the covid affected area in a CT image.

## 5.Joy Iong-Zong Chen, "Design of Accurate Classification of COVID-19 Disease in X-Ray Images Using Deep Learning Approach", Journal of ISMAC,2021. [9]

Using x-ray and the Histogram-Oriented Gradients (HOG) methodology, this study developed an accurate classification method for performing a reliable detection of COVID-19 viral patterns. In this study, 10 fold cross-validation with confusion metrics can be used to detect various diseases caused by lung infection, such as Pneumonia corona virus positivity or negativity. CNN is a deep learning algorithm that can be used in the processing of medical images to support accurate and timely decision-making. CNN training is then used to interpret new medicinal images using pattern recognition. The proposed CNN method achieves high detection accuracy in a quick and efficient manner. The study also used limited datasets from various sources to analyse system robustness by responding to real-world scenarios.

**6.Boran Sekeroglu and Ilker Ozsahin,"Detection of COVID-19 from Chest X-Ray Images Using Convolutional Neural Networks", Original research,2020.[6]**

Cohen provided a total of 225 COVID-19 chest x-ray images. It also included 131 male patients and 64 female patients. Because this is the first publicly available COVID-19 x-ray image collection, and it was created in a short period of time, the dataset used in this study does not have complete metadata. Several categorized experiments were carried out to evaluate the convnet's efficiency on the considered image. Convnet experiments, statistical measurement experiments, and transfer learning experiments were the three types of experiments.

**7.Mohammad Rahimzadeh , Abolfazl Attar,"A modified deep convolutional neural network for detecting COVID-19 and pneumonia from chest X-ray images based on the concatenation of Xception and ResNet50V2 ", Informatics in medicine ,2020. [11]**

## In this paper, they used introduced training techniques to train several deep convolutional networks for classifying x-ray images into three classes: normal, pneumonia, and covid-19. This paper makes use of two open source datasets: kaggle (14,8633 images) and github (222 images). They proposed a neural network that is a hybrid of the Xception and Resnet networks. By combining multiple features extracted by two robust networks, this network achieved the highest accuracy. The network is trained in 8 stages in a row.

## 8.Moutaz Alazab, Albara Awajan , Abdelwadood Mesleh, Ajith Abraham, Vansh Jatana, Salah Alhyari, "COVID-19 Prediction and Detection Using Deep Learning", ijcisim, 2020. [12]

## To forecast the number of COVID-19 confirmations, three forecasting methods were used: the prophet algorithm (PA), the autoregressive integrated moving average (ARIMA) model, and the long short-term memory neural network (LSTM). The COVID-19 detector, which is based on CNN, outperformed in terms of precision, recall, and F-measure. When using augmentation, the COVID-19 detector produced better results. As the time between training and validation became shorter, the training process improved.

## 9.Parag Chatterjee Mainak Biswas, and Arnab Kumar Das, "Specialized covid-19 detection techniques with machine learning", Journal of Physics: Conference Series,2021. [13]

## VGG19, MobileNet, Inception, Xception, and Inception ResNet were the convolutional neural network architectures used in this study. Following several experiments, a parameter known as layer cut-off was defined as the number of un-trainable layers beginning at the bottom of CNN. It employs the transfer learning technique to detect the COVID-19 disease automatically. The benefits of transfer learning include improved results for detecting abnormalities in medical image datasets.

## 10.Pillalamarry Mahesh, Yakkala Gnana Prathyusha, Botlagunta Sahithi, S Nagendram,"Covid-19 Detection from Chest X-Ray using Convolution Neural Networks", ICMAICT, 2020. [15]

There are several methods for diagnosing COVID-19, but they are both expensive and time-consuming. By using a chest x-ray, we can save money and time. However, expert radiotherapists are required to diagnose x-rays. As a result, we created a model that can detect COVID and non-COVID X-rays automatically. Deep Learning algorithms are currently producing the best results in disease classification. Furthermore, features learned by pre-trained Convolution Neural Network (CNN) models on large-scale datasets are extremely useful in image classification tasks. We train and test our model to determine whether the images are COVID or normal. We use analysis to determine the best CNN model for the job. The accuracy metrics are used to validate the model's classification.

**11.S. Tabik , A. Gómez-Ríos, J. L. Martín-Rodríguez, I. Sevillano-García, M. Rey-Area, D. Charte,E. Guirado, J. L. Suárez, J. Luengo, vM. A. Valero-González, P. García-Villanova,E. Olmedo-Sánchez, and F. Herrera, "COVIDGR Dataset and COVID-SDNet Methodology for Predicting COVID-19 Based on Chest X-Ray Images", IEEE Journal of biomedical and health informatics, 2020.[20]**

COVIDGR, a dataset containing 426 positive and 426 negative PA CXR views, was created. CNN is the algorithm used in this case. They proposed the COVID Smart Data based Network (COVID-SDNet) methodology, which increases the generalizability of COVID-classification models. The title of this paper is "Descriptive analysis of dental X-ray images using various practical methods."

**CHAPTER 3**

**SYSTEM ANALYSIS**

**3.1 EXISTING SYSTEM:**

Existing systems employ a variety of tests including swab tests, nasal aspirates, sputum tests and manual analysis of chest x-ray images. Considering that these methods take more time to perform and require additional processes, it becomes necessary to have an automated solution. In emergency situations, this method is too slow to predict.

**DISADVANTAGES:**

* Several hundred swab tests and x-ray images must be routinely processed manually in hospitals due to the large number of patients.
* It takes too long to process cases, which can lead to an increase in covid cases.

**3.2 PROPOSED SYSTEM:**

There are 192 countries that are affected in the covid-19 pandemic and 150 million cases reported worldwide. A deep learning system will increase both speed and accuracy in order to overcome the fallbacks in the current system. Based on chest x-ray images and convolutional neural networks, we developed an automated prediction of covid-19. Covid-19 can be detected most effectively by chest x-ray.

**ADVANTAGES:**

* An X-ray image is a much more accessible and cost-effective test than the others.
* The portable X-Ray machines also allow for testing inside isolation wards, reducing the need for additional protective equipment (PPE).

**3.3 FEASIBILITY STUDY:**

* ECONOMICAL
* TECHNICAL
* SOCIAL

**3.3.1. ECONOMICAL FEASIBILITY:**

Here we have done our project estimation by COCOMO model based on total Lines of Code (LOC) which required to develop our project.

Total Lines of code (LOC) = 9,983.

|  |  |
| --- | --- |
| **MODULES** | **CATEGORY** |
| Inception | Organic |
| Exception | Organic |
| Resnet50 | Organic |
| VGG16 | Organic |

**Inception:** (LOC=2737)

Estimation of development effort = a(KLOC)b PM

= 2.4(0.2737)1.05 PM

**=**0.68PM

Estimation of Development time = c(effort)d

=2.5 (0.68 )0.38.

=0.64Months

**Exception:** (LOC=2367)

Estimation of development effort = a(KLOC)b PM

= 2.4(0.2367)^1.05 PM

**=**0.54PM

Estimation of Development time = c(effort)d

=2.5(0.54)0.38.

=1.12Months

**Resnet50:** (LOC=2589)

Estimation of development effort = a(KLOC)b PM

= 2.4(0.2589)^1.05 PM

**=**0.60PM

Estimation of Development time = c(effort)d

=2.5(0.60)0.38.

=1.17Months

**VGG16:** (LOC=2290)

Estimation of development effort = a(KLOC)b PM

= 2.4(0.2290)^1.05 PM

**=**0.53PM

Estimation of Development time = c(effort)d

=2.5(0.53)0.38.

=1.11Months

Total Effort= 0.68+0.54+0.60+0.53=2.49 PM

Total Development Time=0.64+1.12+1.17+1.11=4.59 months

**3.3.2. TECHNICAL FEASIBILITY:**

**Python:**

Python is used here for predictive modeling because Python-based frameworks give us results faster and also help in the planning of the next steps based on the results.

**CNN:**

The purpose of using convolutional neural network in our project is because it performs prediction on identification efficiently. Immense datasets are applied to CNNs, it is even considered that larger the data, greater the accuracy will result.

**Deep Learning:**

Deep learning techniques can help in determining COVID-19 virus with Chest X-ray Images. Due to the high availability of large-scale annotated image datasets, great success has been achieved using convolutional neural network for image analysis and classification. We use deep learning with python as it solves complex problems and also it is platform independent.

**3.3.3 SOCIAL FEASIBILITY:**

Fast and timely detection of COVID positive patients is necessary to avoid spreading the disease and keeping it in control. This project has been done to detect the COVID positive patients from Chest X-Ray images in a simple and inexpensive way. As our project is inexpensive and it is user-friendly, it is socially feasible.

**3.4 HARDWARE ENVIRONMENT:**

* Processor - I5
* Speed - 3 GHz
* RAM - 4GB (min)
* Hard Disk - 500 GB

**3.5 SOFTWARE ENVIRONMENT:**

* Operating system :   windows 10 (64 bit)
* Programming Language :    python 3.7
* Tools               :    anaconda (jupyter note book IDE)

**CHAPTER 4**

 **SYSTEM DESIGN**

**4.1 ER DIAGRAM:**

ER diagram illustrates the logical structure of a database by defining entities, their attributes, and their relationships.

**ENTITY:**

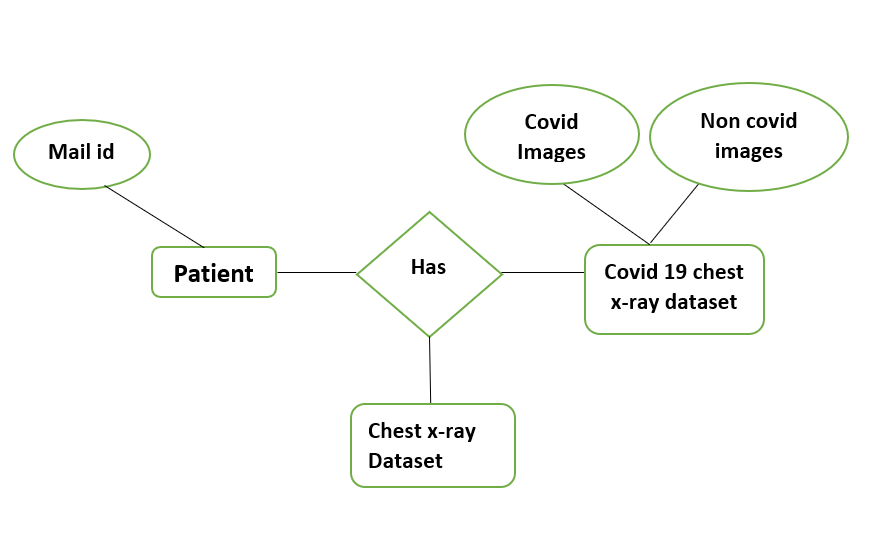
Entities are represented by means of rectangles. Rectangles are named with the entity set they represent. The entities used here are patient, Chest x-ray dataset and Covid-19 chest x-ray dataset.

**ATTRIBUTES:**

The attributes are represented in the form of ellipse. Every attribute is connected to its entity. The three attributes represented here are mail id, covid images and non-covid images.

**RELATIONSHIP:**

Relationships are represented in the form of diamond. A Relationship has is used here which contains three entities named patient, chest x-ray dataset and covid19 chest x-ray dataset. A Relationship which has two entities are called as binary relationship.

****

**FIG.4.1 Er Diagram Of Covid 19 Recognition Using Chest**

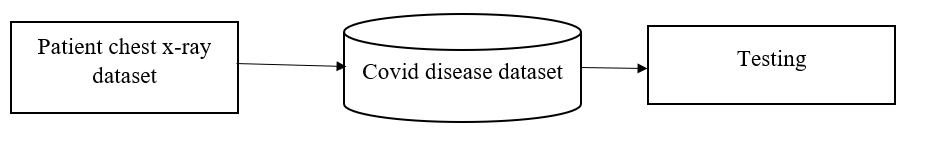
**X-Ray**

**4.2 DATA FLOW DIAGRAM:**

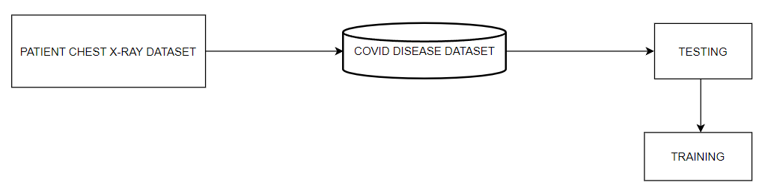
A data flow diagram shows how information flows through a system or process. These include data inputs and outputs, data stores, and the various processes through which the data moves. DFDs use standard symbols and notations to describe the relationships between entities. The four components of data flow diagrams are,

* External entity
* Process
* Data store
* Data flow

**LEVEL 0:**

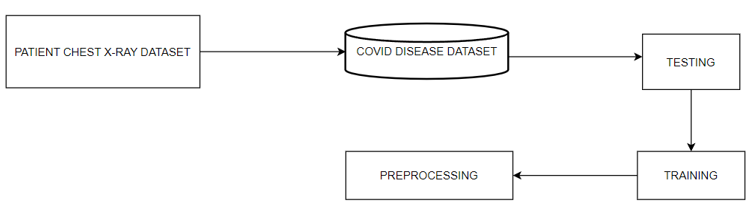
**FIG 4.2 DFD Level 0 Of Covid 19 Recognition Using Chest X-Ray**

**LEVEL 1:**



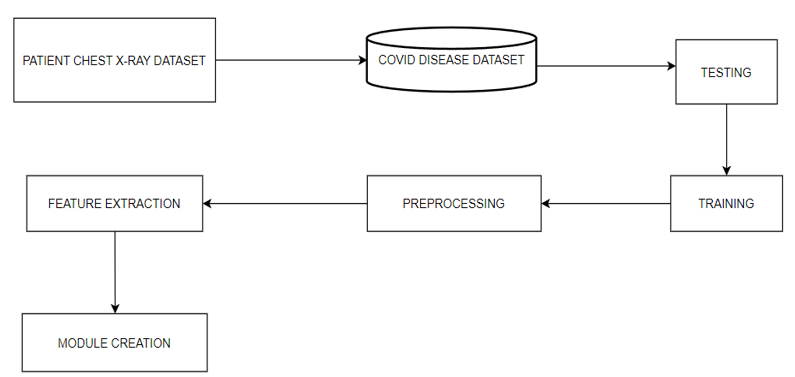
**FIG 4.3 DFD Level 1 Of Covid 19 Recognition Using Chest X-Ray**

**LEVEL 2:**



**FIG 4.4 DFD Level 2 Of Covid 19 Recognition Using Chest X-Ray**

**LEVEL 3:**



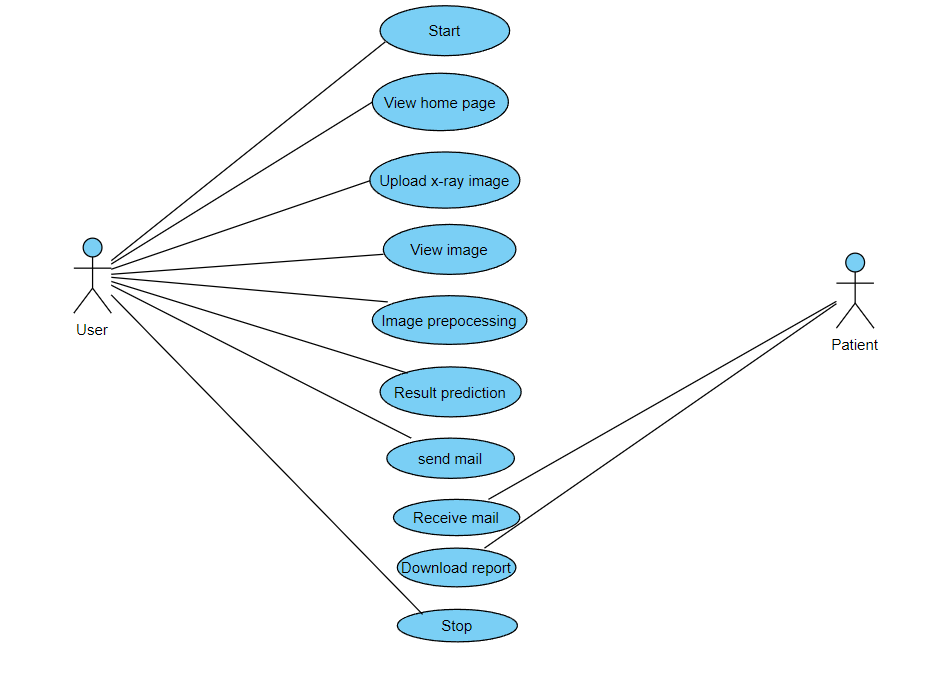
**FIG 4.5 DFD Level 3 Of Covid 19 Recognition Using Chest X-Ray**

**4.3 UML DIAGRAMS:**

**4.3.1.USE CASE:**

A use case represents a particular functionality of a system. Use case diagram is used to describe the relationships among the functionalities and their internal/external controllers. These controllers are known as actors.

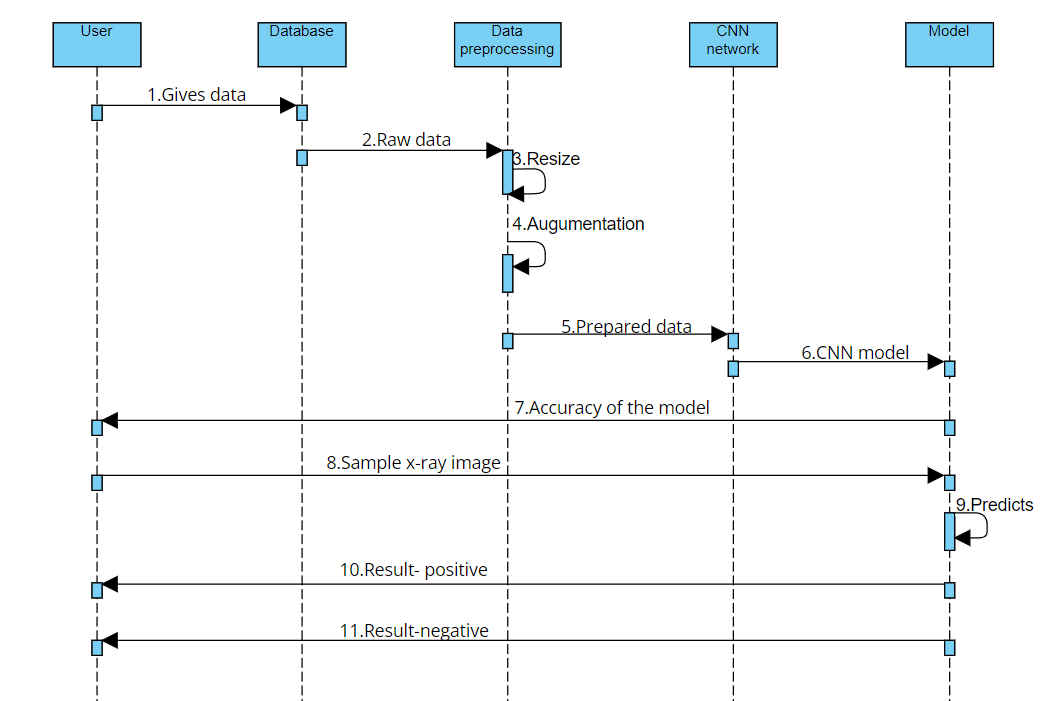
**Actors –** They interacts with the system**,** an actor can be a human being or an organization. The two actors used here are User and Patient. The user will view the home page and select the detect covid button. Then upload the x-ray image and click the upload image button. The image can be viewed by the user after uploading it into the system. After uploading the image will be preprocessed and the result will be predicted. The user can send the report to patient by clicking send mail button. The patient can download the report from the mail.



**FIG 4.7 Use case** **Diagram** **Of Covid 19 Recognition System**

**4.3.2. SEQUENCE DIAGRAM:**

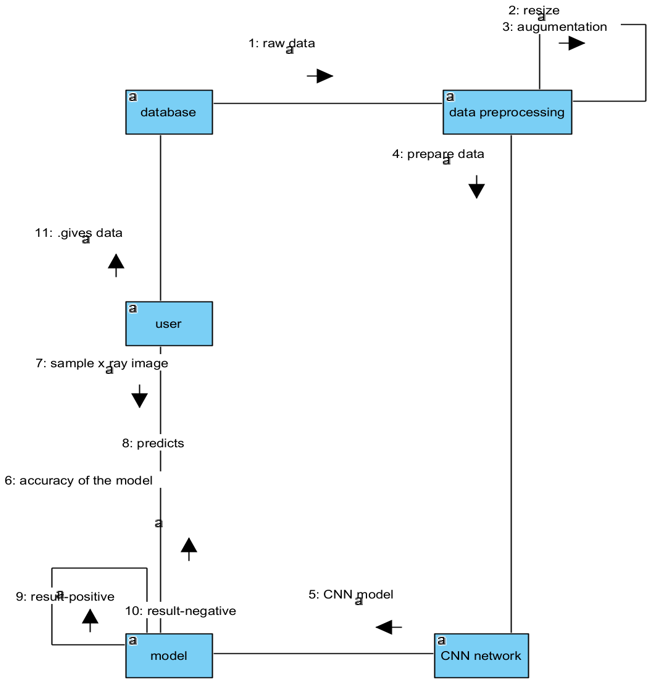
A sequence diagram is an interaction diagram. From the name, it is clear that the diagram deals with some sequences, which are the sequence of messages flowing from one object to another. The sequence diagram includes a group of objects which are represented by lifelines, and the messages they exchange over time during their interaction.

****

**FIG 4.8 Sequence Diagram Of Covid 19 Recognition System**

**4.3.3. COLLABORATION DIAGRAM**

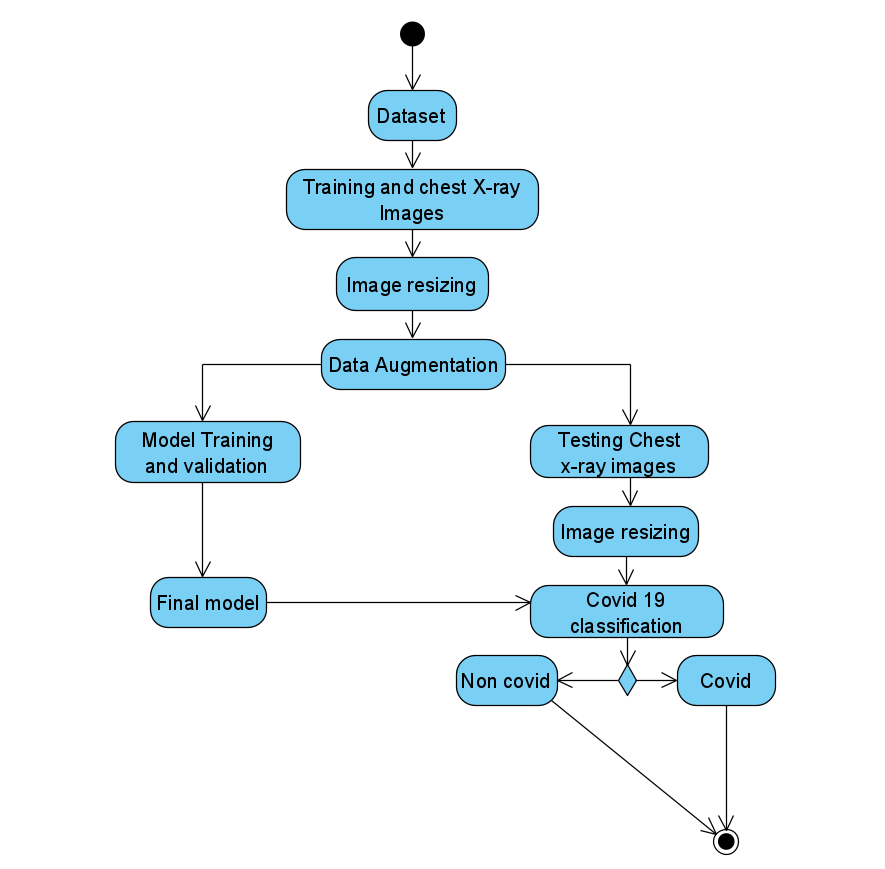
A collaboration diagram, also known as a communication diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language. Collaboration diagram is another form of interaction diagram. It represents the structural organization of a system and the messages sent/received. Structural organization consists of objects and links.

****

**FIG 4.9 Collaboration Diagram Of Covid 19 Recognition System**

**4.3.4. ACTIVITY DIAGRAM:**

Activity diagram represents the flow of activities from one to another. The activity can be described as an operation of the system. An activity diagram captures the dynamic behaviour of the system. It is also known as an object-oriented flowchart. Activity diagrams include swim lanes, branching, parallel flow, control nodes, expansion nodes, and object nodes.

****

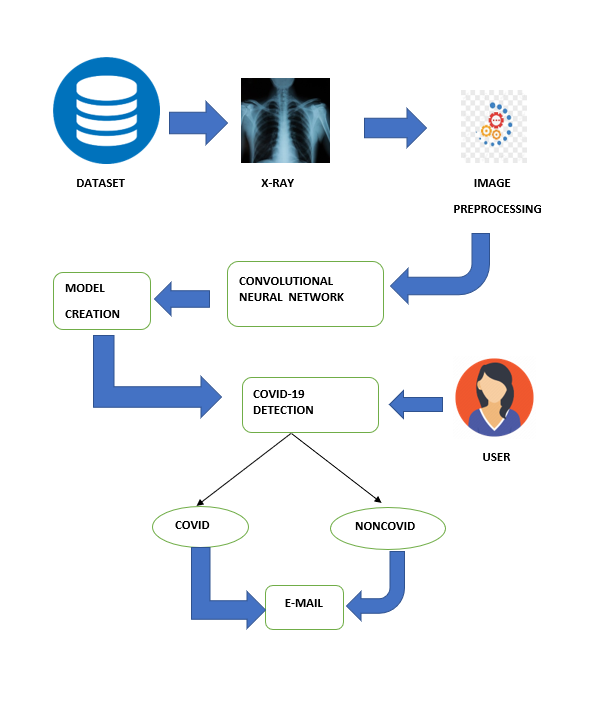
**FIG 4.10 Activity Diagram Of Covid 19 Recognition System**

**CHAPTER 5**

**SYSTEM ARCHITECTURE**

**5.1 SYSTEM OVERVIEW**

The system architecture is the basic structure of the system. The chest x-ray image is taken from the dataset and the image will be pre-processed using convolutional neural network. By using pre-processed image model is created. When the user uploads the chest x-ray image the result will be based on the given x-ray image. The report will be sent to the user mail id as covid or non-covid.

**FIG.5.1 System Architecture Of Covid 19 Recognition System**

**5.2 MODULE DESIGN SPECIFICATION**

The following are the list of modules of covid 19 recognition using chest x-ray:

* + - * + Image Acquisition
        + Data Pre-processing
        + Train the model
        + Evaluation of model
        + Send mail

**5.1.1.** **IMAGE ACQUISITION**

The acquisition of photographs is the initial step. The computer must learn by doing in order to create a classification model. To recognize an object, the computer must view a large number of photos. Deep learning models can also be trained with other forms of data, such as time series data and voice data. Images will be the relevant data for detecting covid. A chest X-ray image is one example of a possible image. This stage produces photos that will be used to train the model later. The project's data was obtained from an open source Github repository. Chest X-ray images (1000 images) were obtained from: <https://github.com/ieee8023/covid-chestxray-dataset>. On Chest X-ray, VGG16, ResNet 50, Inception V3, and Xception were trained.

**5.1.2. DATA PRE-PROCESSING**

In the deep learning process, image pre-processing is a very common and beneficial technique that not only increases the size of the original dataset but also enriches the information implicit in it. The dataset contains images of various sizes. They can be fed into deep learning models for training after being resized to a fixed size. The images are resized to 224 x 224 px, which is regarded as the ideal size for the ResNet50 model. As a result, the shape input tensor (224, 224, 3) is added to the pretrained ResNet50 model. A squared image with a fixed aspect ratio is scaled to roughly the same height and width. To filter the size of all input samples, the image filtering preprocessing technique is used. The X-ray images have been resized to 224 224. As previously stated, we used a powerful image enhancement method known as Dynamic Histogram Equalization (DHE) to improve the quality of images before they were fed into the CNN model. Histogram Equalization (HE), which refers to mapping from the initial narrow pixel levels to a larger extent and improves image enhancement, is a widely used technique in image processing. The HE technique is used to convert the grey levels of an image by using the cumulative effort function globally, but it always results in the loss of elaboration information in images, resulting in poor image quality. This popular image contrast enhancement method could improve image contrast in a variety of applications, including MRI, X-rays, and CT.

**5.1.3. TRAIN MODEL**

After the model has been built, the next step is to train it. Image Data Generator is designed to train models on modified images, such as those with different angles, flips, rotations, or shifts. We were successful in developing an image-recognition artificial convolutional neural network. Separate the dataset into two sections: training and testing. Finally, we will create and train the model using the training dataset. To learn CXR modality-specific feature representations, a custom CNN and a selection of pretrained CNN models are trained on a large-scale selection of CXRs. The acquired knowledge is then transferred and fine-tuned in order to classify covid and non-covid CXRs. We use the advantages of modality-specific knowledge transfer, iterative pruning, and ensemble strategies to reduce model complexity and improve the DL model's robustness, generalization, and inference capability. With a batch size of 32 images, the model is trained for 500 epochs. Eighty percent of the images were used to train the models, with the remaining twenty percent used to test the models' accuracy. After training, the accuracies achieved for the model are as follows:

**Table 5.1 Training model of covid 19 recognition using chest x-ray**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | INCEPTIONV3 | VGG16 | RESNET50 | XCEPTION |
| CHEST  X-RAY  IMAGES | 96% | 94% | 83% | 92% |

**5.1.4. EVALUATION OF MODEL**

Model testing is possible after the model has been trained. This step retrieves a test set of data. Because the model has never seen this set of data before, its true correctness will be confirmed. Finally, the model that has been saved can be used in the real world. This step is known as model evaluation. This means that the model can be used with new data. Several evaluation metrics, such as accuracy, precision, recall, and F1 score, are described. Four indices, True Positive, True Negative, False Positive, False Negative, are used to analyse and identify model performance based on model outputs. The True Positive indicates that the model has signed the chest X-ray images that have covid as covid as well. If the chest X-ray images do not show covid as well as the model predicts, this is a True Negative. The precision rate was always used to estimate how much the number of images that are truly covid accounted for in the total number of positive covid examples. That is, covid images must be identified in practical clinical diagnoses, so the precision rate is critical. Most of the time, the higher the precision rate, the lower the recall rate. As a result, the F1 score rate is widely regarded as an appropriate criterion.

**5.1.5 SEND MAIL**

After the evaluation of the model, the results can be sent to the user mail id. There will be a send mail option visible in the result page through which the result is sent to the user as a report of covid or non-covid.

**5.2. ALGORITHM:**

**5.2.1. CONVOLUTIONAL NEURAL NETWORK(CNN):**

Deep learning neural networks are subsets of convolution neural networks. Its primary applications are image classification and image analysis. CNN's goal is to simulate how the human brain analyses images. Convolution neural networks are made up of one or more Convolutional layers, followed by one or more fully connected layers. The CNN is made up of three layers: input, hidden, and output. The input layer is made up of arrays of pixels. The hidden layer is the most important layer because it is responsible for image computation. The hidden layer is made up of activation functions and biases. The output layer assists us in calculating the class score. CNNs have the advantage of being easier to train and providing high accuracies.

Multiple layers of artificial neurons make up convolutional neural networks. Artificial neurons are mathematical functions that calculate the weighted sum of multiple inputs and output an activation value, similar to biological neurons. When you feed an image into a ConvNet, each layer generates several activation functions, which are then passed on to the next layer. Typically, the first layer extracts basic features such as horizontal or diagonal edges. This output is forwarded to the next layer, which detects more complex features like corners or combinational edges. It can identify even more complex features as we move deeper into the network.

**INPUT LAYER:**

It is the layer where we provide input to our model. The total number of features in our data is equal to the number of neurons in this layer (number of pixels in the case of an image).

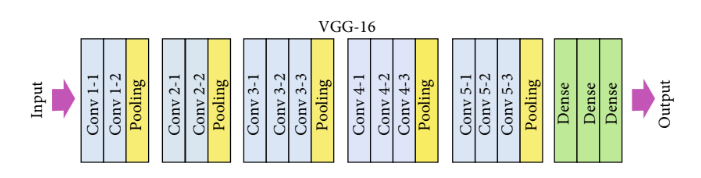
**HIDDEN LAYER:**

The hidden layer receives the input from the Input layer. Depending on our model and data size, there could be many hidden layers. Each hidden layer can have a different number of neurons, which should be greater than the number of features. The output of each layer is computed by matrix multiplication of the previous layer's output with learnable weights of that layer, followed by the addition of learnable biases and an activation function, which makes the network nonlinear.

**OUTPUT LAYER:**

The hidden layer output is then fed into a logistic function, such as sigmoid or softmax, which converts the output of each class into the probability score of each class.

**VGG:**

VGG stands for Visual Geometry Group, and it is a multi-layered deep Convolutional Neural Network (CNN) architecture. The "deep" refers to the number of layers, with VGG-16 or VGG-19 having 16 or 19 convolutional layers, respectively. VGGNet-16 can classify images into 1000 object categories and has 16 layers. The image input size for the model is 224 by 224. VGG16 is a CNN model developed at Oxford University by the VGG. AlexNet, the network's replacement, was founded in 2012. As shown in Figure 5.2.1, VGG16 has eight layers: three completely connected layers, five max-pooling layers, and one softmax layer. The architecture was created as part of the ImageNet competition. The width of the convolution blocks is set to a small integer. After each max-pooling operation, the width parameter is increased by two until it reaches 512. The image size for the VGG16 is 224 224 pixels. To keep the image's spatial resolution, spatial padding was used. The VGG16 network has been made open-source, allowing for similar operations to be carried out. The model can also be used for transfer learning because certain frameworks, such as Keras, provide pretrained weights that can be used to build custom models with minor changes. ****

**FIG 5.2 VGG16 Model**

**RESNET:**

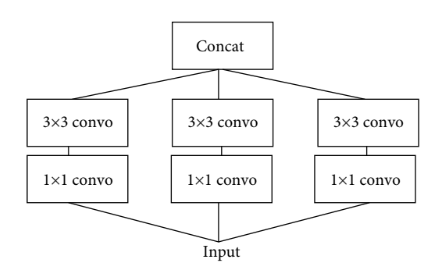
ResNet-50 is a 50-layer deep convolutional neural network. A pretrained version of the network trained on over a million images from the ImageNet database can be loaded. The pretrained network can classify images into 1000 different categories.

**INCEPTION-V3:**

Inception-V3 is a 48-layer deep convolutional neural network. A pretrained version of the network trained on over a million images from the ImageNet database can be loaded. For a wide range of images, the network has learned rich feature representations. The network's image input size is 299 by 299 pixels.

**XCEPTION:**

Xception is a 71-layer deep convolutional neural network. The ImageNet database can be used to load a pretrained version of the network trained on over a million images. The pretrained network can categorize images into 1000 different object categories. The Inception network has been superseded by the Xception network. Extreme inception is also known as Xception. The Xception network employs depth-wise separable convolution layers rather than traditional convolution layers. Xception includes mapping spatial and cross-channel correlations, which can be completely separated in CNN feature maps. For a wide range of images, the network has learned rich feature representations. The network's image input size is 299 by 299 pixels. To obtain cross-channel correlations in an input image, the input image is converted into spatial correlations within each output channel.

****

**FIG 5.3 Xception Model**

**CHAPTER 6**

**SYSTEM IMPLEMENTATION**

**6.1 CODING:**

**INCEPTION V3:**

from \_\_future\_\_ import print\_function, division

from builtins import range, input

from tensorflow.keras.layers import Input, Lambda, Dense, Flatten, AveragePooling2D, Dropout

from tensorflow.keras.models import Model, load\_model

from tensorflow.keras.applications import InceptionV3

from tensorflow.keras.applications.resnet50 import preprocess\_input

from tensorflow.keras.preprocessing import image

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from sklearn.metrics import confusion\_matrix, roc\_curve

import seaborn as sns

import numpy as np

import matplotlib.pyplot as plt

import cv2

from glob import glob

#define size to which images are to be resized

IMAGE\_SIZE = [224, 224]

# training config:

epochs = 500

batch\_size = 32

#define paths

covid\_path = 'data/chest/Chest\_COVID'

noncovid\_path = 'data/chest/Chest\_NonCOVID'

# Use glob to grab images from path .jpg or jpeg

covid\_files = glob(covid\_path + '/\*')

noncovid\_files = glob(noncovid\_path + '/\*')

# Preparing Labels

covid\_labels = [ ]

noncovid\_labels = [ ]

covid\_images=[ ]

noncovid\_images=[ ]

import cv2

for i in range(len(covid\_files)):

image = cv2.imread(covid\_files[i])

image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

image = cv2.resize(image,(224,224))

covid\_images.append(image)

covid\_labels.append('Chest\_COVID')

for i in range(len(noncovid\_files)):

image = cv2.imread(noncovid\_files[i])

image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

image = cv2.resize(image,(224,224))

noncovid\_images.append(image)

noncovid\_labels.append('Chest\_NonCOVID')

def plot\_images(images, title):

nrows, ncols = 5, 8

figsize = [10, 6]

fig, ax = plt.subplots(nrows=nrows, ncols=ncols, figsize=figsize, facecolor=(1, 1, 1))

for i, axi in enumerate(ax.flat):

axi.imshow(images[i])

axi.set\_axis\_off()

plt.suptitle(title, fontsize=24)

plt.tight\_layout(pad=0.2, rect=[0, 0, 1, 0.9])

plt.show()

plot\_images(covid\_images, 'Positive COVID-19 Chest X-ray')

plot\_images(noncovid\_images, 'Negative COVID-19 Chest X-ray')

# normalize to interval of [0,1]

covid\_images = np.array(covid\_images) / 255

noncovid\_images = np.array(noncovid\_images) / 255

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import LabelBinarizer

from tensorflow.keras.utils import to\_categorical

# split into training and testing

covid\_x\_train, covid\_x\_test, covid\_y\_train, covid\_y\_test = train\_test\_split(

covid\_images, covid\_labels, test\_size=0.2)

noncovid\_x\_train, noncovid\_x\_test, noncovid\_y\_train, noncovid\_y\_test = train\_test\_split(

noncovid\_images, noncovid\_labels, test\_size=0.2)

X\_train = np.concatenate((noncovid\_x\_train, covid\_x\_train), axis=0)

X\_test = np.concatenate((noncovid\_x\_test, covid\_x\_test), axis=0)

y\_train = np.concatenate((noncovid\_y\_train, covid\_y\_train), axis=0)

y\_test = np.concatenate((noncovid\_y\_test, covid\_y\_test), axis=0)

# make labels into categories - either 0 or 1

y\_train = LabelBinarizer().fit\_transform(y\_train)

y\_train = to\_categorical(y\_train)

y\_test = LabelBinarizer().fit\_transform(y\_test)

y\_test = to\_categorical(y\_test)

plot\_images(covid\_x\_train, 'X\_train')

plot\_images(covid\_x\_test, 'X\_test')

# y\_train and y\_test contain class lables 0 and 1 representing COVID and NonCOVID for X\_train and X\_test

inception = InceptionV3(weights="imagenet", include\_top=False,

input\_tensor=Input(shape=(224, 224, 3)))

outputs = inception.output

outputs = Flatten(name="flatten")(outputs)

outputs = Dropout(0.5)(outputs)

outputs = Dense(2, activation="softmax")(outputs)

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

for layer in inception.layers:

layer.trainable = False

model.compile(

loss='categorical\_crossentropy',

optimizer='adam',

metrics=['accuracy']

)

train\_aug = ImageDataGenerator(

rotation\_range=20,

width\_shift\_range=0.2,

height\_shift\_range=0.2,

horizontal\_flip=True

)

model.summary()

history = model.fit(train\_aug.flow(X\_train, y\_train, batch\_size=32),

validation\_data=(X\_test, y\_test),

validation\_steps=len(X\_test) / 32,

steps\_per\_epoch=len(X\_train) / 32,

epochs=500

model.save('inceptionv3\_chest.h5')

model.save\_weights('inceptionv3\_chest.hdf5')

model = load\_model('inceptionv3\_chest.h5')

y\_pred = model.predict(X\_test, batch\_size=batch\_size)

prediction=y\_pred[0:10]

for index, probability in enumerate(prediction):

if probability[1] > 0.5:

plt.title('%.2f' % (probability[1]\*100) + '% COVID')

else:

plt.title('%.2f' % ((1-probability[1])\*100) + '% NonCOVID')

plt.imshow(X\_test[index])

plt.show()

# Convert to Binary classes

y\_pred\_bin = np.argmax(y\_pred, axis=1)

y\_test\_bin = np.argmax(y\_test, axis=1)

fpr, tpr, thresholds = roc\_curve(y\_test\_bin, y\_pred\_bin)

plt.plot(fpr, tpr)

plt.xlim([0.0, 1.0])

plt.ylim([0.0, 1.0])

plt.rcParams['font.size'] = 12

plt.title('ROC curve for our model')

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.grid(True)

def plot\_confusion\_matrix(normalize):

classes = ['COVID','NonCOVID']

tick\_marks = [0.5,1.5]

cn = confusion\_matrix(y\_test\_bin, y\_pred\_bin,normalize=normalize)

sns.heatmap(cn,cmap='plasma',annot=True)

plt.xticks(tick\_marks, classes)

plt.yticks(tick\_marks, classes)

plt.title('Confusion Matrix')

plt.ylabel('True label')

plt.xlabel('Predicted label')

plt.show()

print('Confusion Matrix without Normalization')

plot\_confusion\_matrix(normalize=None)

print('Confusion Matrix with Normalized Values')

plot\_confusion\_matrix(normalize='true')

from sklearn.metrics import classification\_report

print(classification\_report(y\_test\_bin, y\_pred\_bin))

plt.figure(figsize=(10,10))

plt.plot(history.history['accuracy'])

plt.plot(history.history['val\_accuracy'])

plt.title('Model Accuracy')

plt.ylabel('Accuracy')

plt.xlabel('Epoch')

plt.legend(['Training', 'Testing'])

plt.savefig('inception\_chest\_accuracy.png')

plt.show()

plt.figure(figsize=(10,10))

plt.plot(history.history['loss'])

plt.plot(history.history['val\_loss'])

plt.title('Model Loss')

plt.ylabel('Loss')

plt.xlabel('Epoch')

plt.legend(['Training', 'Testing'])

plt.savefig('inception\_chest\_loss.png')

plt.show()

**XCEPTION:**

from \_\_future\_\_ import print\_function, division

from builtins import range, input

from tensorflow.keras.layers import Input, Lambda, Dense, Flatten, AveragePooling2D, Dropout

from tensorflow.keras.models import Model, load\_model

from tensorflow.keras.applications import Xception

from tensorflow.keras.applications.resnet50 import preprocess\_input

from tensorflow.keras.preprocessing import image

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from sklearn.metrics import confusion\_matrix, roc\_curve

import seaborn as sns

import numpy as np

import matplotlib.pyplot as plt

import cv2

from glob import glob

#define size to which images are to be resized

IMAGE\_SIZE = [224, 224]

epochs = 500

batch\_size = 32

covid\_path = 'data/chest/Chest\_COVID'

noncovid\_path = 'data/chest/Chest\_NonCOVID'

covid\_files = glob(covid\_path + '/\*')

noncovid\_files = glob(noncovid\_path + '/\*')

covid\_labels = []

noncovid\_labels = []

covid\_images=[]

noncovid\_images=[]

import cv2

for i in range(len(covid\_files)):

image = cv2.imread(covid\_files[i])

image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

image = cv2.resize(image,(224,224))

covid\_images.append(image)

covid\_labels.append('Chest\_COVID')

for i in range(len(noncovid\_files)):

image = cv2.imread(noncovid\_files[i])

image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

image = cv2.resize(image,(224,224))

noncovid\_images.append(image)

noncovid\_labels.append('Chest\_NonCOVID')

def plot\_images(images, title):

nrows, ncols = 5, 8

figsize = [10, 6]

fig, ax = plt.subplots(nrows=nrows, ncols=ncols, figsize=figsize, facecolor=(1, 1, 1))

for i, axi in enumerate(ax.flat):

axi.imshow(images[i])

axi.set\_axis\_off()

plt.suptitle(title, fontsize=24)

plt.tight\_layout(pad=0.2, rect=[0, 0, 1, 0.9])

plt.show()

plot\_images(covid\_images, 'Positive COVID-19 Chest X-ray')

plot\_images(noncovid\_images, 'Negative COVID-19 Chest X-ray')

covid\_images = np.array(covid\_images) / 255

noncovid\_images = np.array(noncovid\_images) / 255

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import LabelBinarizer

from tensorflow.keras.utils import to\_categorical

# split into training and testing

covid\_x\_train, covid\_x\_test, covid\_y\_train, covid\_y\_test = train\_test\_split(

covid\_images, covid\_labels, test\_size=0.2)

noncovid\_x\_train, noncovid\_x\_test, noncovid\_y\_train, noncovid\_y\_test = train\_test\_split(

noncovid\_images, noncovid\_labels, test\_size=0.2)

X\_train = np.concatenate((noncovid\_x\_train, covid\_x\_train), axis=0)

X\_test = np.concatenate((noncovid\_x\_test, covid\_x\_test), axis=0)

y\_train = np.concatenate((noncovid\_y\_train, covid\_y\_train), axis=0)

y\_test = np.concatenate((noncovid\_y\_test, covid\_y\_test), axis=0)

# make labels into categories - either 0 or 1

y\_train = LabelBinarizer().fit\_transform(y\_train)

y\_train = to\_categorical(y\_train)

y\_test = LabelBinarizer().fit\_transform(y\_test)

y\_test = to\_categorical(y\_test)

plot\_images(covid\_x\_train, 'X\_train')

plot\_images(covid\_x\_test, 'X\_test')

xception = Xception(weights="imagenet", include\_top=False,

input\_tensor=Input(shape=(224, 224, 3)))

outputs = xception.output

outputs = Flatten(name="flatten")(outputs)

outputs = Dropout(0.5)(outputs)

outputs = Dense(2, activation="softmax")(outputs)

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

for layer in xception.layers:

layer.trainable = False

model.compile(

loss='categorical\_crossentropy',

optimizer='adam',

metrics=['accuracy']

)

train\_aug = ImageDataGenerator(

rotation\_range=20,

width\_shift\_range=0.2,

height\_shift\_range=0.2,

horizontal\_flip=True

)

history = model.fit(train\_aug.flow(X\_train, y\_train, batch\_size=32),

validation\_data=(X\_test, y\_test),

validation\_steps=len(X\_test) / 32,

steps\_per\_epoch=len(X\_train) / 32,

epochs=500)

model.save('xception\_chest.h5')

model.save\_weights('xceptionweights\_chest.hdf5')

model = load\_model('xception\_chest.h5')

prediction=y\_pred[0:10]

for index, probability in enumerate(prediction):

if probability[1] > 0.5:

plt.title('%.2f' % (probability[1]\*100) + '% COVID')

else:

plt.title('%.2f' % ((1-probability[1])\*100) + '% NonCOVID')

plt.imshow(X\_test[index])

plt.show()

y\_pred\_bin = np.argmax(y\_pred, axis=1)

y\_test\_bin = np.argmax(y\_test, axis=1)

fpr, tpr, thresholds = roc\_curve(y\_test\_bin, y\_pred\_bin)

plt.plot(fpr, tpr)

plt.xlim([0.0, 1.0])

plt.ylim([0.0, 1.0])

plt.rcParams['font.size'] = 12

plt.title('ROC curve for our model')

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.grid(True)

def plot\_confusion\_matrix(normalize):

classes = ['COVID','NonCOVID']

tick\_marks = [0.5,1.5]

cn = confusion\_matrix(y\_test\_bin, y\_pred\_bin,normalize=normalize)

sns.heatmap(cn,cmap='plasma',annot=True)

plt.xticks(tick\_marks, classes)

plt.yticks(tick\_marks, classes)

plt.title('Confusion Matrix')

plt.ylabel('True label')

plt.xlabel('Predicted label')

plt.show()

print('Confusion Matrix without Normalization')

plot\_confusion\_matrix(normalize=None)

print('Confusion Matrix with Normalized Values')

plot\_confusion\_matrix(normalize='true')

from sklearn.metrics import classification\_report

print(classification\_report(y\_test\_bin, y\_pred\_bin))

plt.figure(figsize=(10,10))

plt.plot(history.history['accuracy'])

plt.plot(history.history['val\_accuracy'])

plt.title('Model Accuracy')

plt.ylabel('Accuracy')

plt.xlabel('Epoch')

plt.legend(['Training', 'Testing'])

plt.savefig('xception\_chest\_accuracy.png')

plt.show()

plt.figure(figsize=(10,10))

plt.plot(history.history['loss'])

plt.plot(history.history['val\_loss'])

plt.title('Model Loss')

plt.ylabel('Loss')

plt.xlabel('Epoch')

plt.legend(['Training', 'Testing'])

plt.savefig('xception\_chest\_loss.png')

plt.show()

**RESNET:**

from \_\_future\_\_ import print\_function, division

from builtins import range, input

from tensorflow.keras.layers import Input, Lambda, Dense, Flatten, AveragePooling2D, Dropout

from tensorflow.keras.models import Model, load\_model

from tensorflow.keras.applications.resnet50 import ResNet50

from tensorflow.keras.applications.resnet50 import preprocess\_input

from tensorflow.keras.preprocessing import image

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from sklearn.metrics import confusion\_matrix, roc\_curve

import seaborn as sns

import numpy as np

import matplotlib.pyplot as plt

import cv2

from glob import glob

IMAGE\_SIZE = [224, 224]

epochs = 500

batch\_size = 32

covid\_path = 'data/chest/Chest\_COVID'

noncovid\_path = 'data/chest/Chest\_NonCOVID'

covid\_files = glob(covid\_path + '/\*')

noncovid\_files = glob(noncovid\_path + '/\*')

covid\_labels = []

noncovid\_labels = []

covid\_images=[]

noncovid\_images=[]

import cv2

for i in range(len(covid\_files)):

image = cv2.imread(covid\_files[i])

image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

image = cv2.resize(image,(224,224))

covid\_images.append(image)

covid\_labels.append('Chest\_COVID')

for i in range(len(noncovid\_files)):

image = cv2.imread(noncovid\_files[i])

image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

image = cv2.resize(image,(224,224))

noncovid\_images.append(image)

noncovid\_labels.append('Chest\_NonCOVID')

def plot\_images(images, title):

nrows, ncols = 5, 8

figsize = [10, 6]

fig, ax = plt.subplots(nrows=nrows, ncols=ncols, figsize=figsize, facecolor=(1, 1, 1))

for i, axi in enumerate(ax.flat):

axi.imshow(images[i])

axi.set\_axis\_off()

plt.suptitle(title, fontsize=24)

plt.tight\_layout(pad=0.2, rect=[0, 0, 1, 0.9])

plt.show()

plot\_images(covid\_images, 'Positive COVID-19 Chest X-ray')

plot\_images(noncovid\_images, 'Negative COVID-19 Chest X-ray')

covid\_images = np.array(covid\_images) / 255

noncovid\_images = np.array(noncovid\_images) / 255

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import LabelBinarizer

from tensorflow.keras.utils import to\_categorical

# split into training and testing

covid\_x\_train, covid\_x\_test, covid\_y\_train, covid\_y\_test = train\_test\_split(

covid\_images, covid\_labels, test\_size=0.2)

noncovid\_x\_train, noncovid\_x\_test, noncovid\_y\_train, noncovid\_y\_test = train\_test\_split(

noncovid\_images, noncovid\_labels, test\_size=0.2)

X\_train = np.concatenate((noncovid\_x\_train, covid\_x\_train), axis=0)

X\_test = np.concatenate((noncovid\_x\_test, covid\_x\_test), axis=0)

y\_train = np.concatenate((noncovid\_y\_train, covid\_y\_train), axis=0)

y\_test = np.concatenate((noncovid\_y\_test, covid\_y\_test), axis=0)

# make labels into categories - either 0 or 1

y\_train = LabelBinarizer().fit\_transform(y\_train)

y\_train = to\_categorical(y\_train)

y\_test = LabelBinarizer().fit\_transform(y\_test)

y\_test = to\_categorical(y\_test)

plot\_images(covid\_x\_train, 'X\_train')

plot\_images(covid\_x\_test, 'X\_test')

res = ResNet50(weights="imagenet", include\_top=False,

input\_tensor=Input(shape=(224, 224, 3)))

outputs = res.output

outputs = Flatten(name="flatten")(outputs)

outputs = Dropout(0.5)(outputs)

outputs = Dense(2, activation="softmax")(outputs)

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

for layer in res.layers:

layer.trainable = False

model.compile(

loss='categorical\_crossentropy',

optimizer='adam',

metrics=['accuracy']

)

train\_aug = ImageDataGenerator(

rotation\_range=20,

width\_shift\_range=0.2,

height\_shift\_range=0.2,

horizontal\_flip=True

)

history = model.fit(train\_aug.flow(X\_train, y\_train, batch\_size=32),

validation\_data=(X\_test, y\_test),

validation\_steps=len(X\_test) / 32,

steps\_per\_epoch=len(X\_train) / 32,

epochs=500)

model.save('resnet\_chest.h5')

model.save\_weights('resnetweights\_chest.hdf5')

model = load\_model('resnet\_chest.h5')

y\_pred = model.predict(X\_test, batch\_size=batch\_size)

prediction=y\_pred[0:10]

for index, probability in enumerate(prediction):

if probability[1] > 0.5:

plt.title('%.2f' % (probability[1]\*100) + '% COVID')

else:

plt.title('%.2f' % ((1-probability[1])\*100) + '% NonCOVID')

plt.imshow(X\_test[index])

plt.show()

y\_pred\_bin = np.argmax(y\_pred, axis=1)

y\_test\_bin = np.argmax(y\_test, axis=1)

fpr, tpr, thresholds = roc\_curve(y\_test\_bin, y\_pred\_bin)

plt.plot(fpr, tpr)

plt.xlim([0.0, 1.0])

plt.ylim([0.0, 1.0])

plt.rcParams['font.size'] = 12

plt.title('ROC curve for our model')

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.grid(True)

def plot\_confusion\_matrix(normalize):

classes = ['COVID','NonCOVID']

tick\_marks = [0.5,1.5]

cn = confusion\_matrix(y\_test\_bin, y\_pred\_bin,normalize=normalize)

sns.heatmap(cn,cmap='plasma',annot=True)

plt.xticks(tick\_marks, classes)

plt.yticks(tick\_marks, classes)

plt.title('Confusion Matrix')

plt.ylabel('True label')

plt.xlabel('Predicted label')

plt.show()

print('Confusion Matrix without Normalization')

plot\_confusion\_matrix(normalize=None)

print('Confusion Matrix with Normalized Values')

plot\_confusion\_matrix(normalize='true')

from sklearn.metrics import classification\_report

print(classification\_report(y\_test\_bin, y\_pred\_bin))

plt.figure(figsize=(10,10))

plt.plot(history.history['accuracy'])

plt.plot(history.history['val\_accuracy'])

plt.title('Model Accuracy')

plt.ylabel('Accuracy')

plt.xlabel('Epoch')

plt.legend(['Training', 'Testing'])

plt.savefig('resnet\_chest\_accuracy.png')

plt.show()

plt.figure(figsize=(10,10))

plt.plot(history.history['loss'])

plt.plot(history.history['val\_loss']

plt.title('Model Loss')

plt.ylabel('Loss')

plt.xlabel('Epoch')

plt.legend(['Training', 'Testing'])

plt.savefig('resnet\_chest\_loss.png')

plt.show()

**VGG16:**

from \_\_future\_\_ import print\_function, division

from builtins import range, input

from tensorflow.keras.layers import Input, Lambda, Dense, Flatten, GlobalAveragePooling2D, Dropout

from tensorflow.keras.models import Model, load\_model

from tensorflow.keras.applications import VGG19

from tensorflow.keras.applications.vgg16 import preprocess\_input

from tensorflow.keras.preprocessing import image

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from sklearn.metrics import confusion\_matrix, roc\_curve

import seaborn as sns

import numpy as np

import matplotlib.pyplot as plt

from glob import glob

import pandas as pd

IMAGE\_SIZE = [224, 224] # feel free to change depending on dataset

epochs = 500

batch\_size = 32

covid\_path = 'data/chest/Chest\_COVID'

noncovid\_path = 'data/chest/Chest\_NonCOVID'

covid\_files = glob(covid\_path + '/\*')

noncovid\_files = glob(noncovid\_path + '/\*')

covid\_labels = [ ]

noncovid\_labels = [ ]

covid\_images = [ ]

noncovid\_images = [ ]

import cv2

for i in range(len(covid\_files)):

image = cv2.imread(covid\_files[i])

image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

image = cv2.resize(image,(224,224))

covid\_images.append(image)

covid\_labels.append('Chest\_COVID')

for i in range(len(noncovid\_files)):

image = cv2.imread(noncovid\_files[i])

image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

image = cv2.resize(image,(224,224))

noncovid\_images.append(image)

noncovid\_labels.append('Chest\_NonCOVID')

def plot\_images(images, title):

nrows, ncols = 5, 8

figsize = [10, 6]

fig, ax = plt.subplots(nrows=nrows, ncols=ncols, figsize=figsize, facecolor=(1, 1, 1))

for i, axi in enumerate(ax.flat):

axi.imshow(images[i])

axi.set\_axis\_off()

plt.suptitle(title, fontsize=24)

plt.tight\_layout(pad=0.2, rect=[0, 0, 1, 0.9])

plt.show()

plot\_images(covid\_images, 'Positive COVID-19 Chest X-ray')

plot\_images(noncovid\_images, 'Negative COVID-19 Chest X-ray')

covid\_images = np.array(covid\_images) / 255

noncovid\_images = np.array(noncovid\_images) / 255

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import LabelBinarizer

from tensorflow.keras.utils import to\_categorical

# split into training and testing

covid\_x\_train, covid\_x\_test, covid\_y\_train, covid\_y\_test = train\_test\_split(

covid\_images, covid\_labels, test\_size=0.2)

noncovid\_x\_train, noncovid\_x\_test, noncovid\_y\_train, noncovid\_y\_test = train\_test\_split(

noncovid\_images, noncovid\_labels, test\_size=0.2)

X\_train = np.concatenate((noncovid\_x\_train, covid\_x\_train), axis=0)

X\_test = np.concatenate((noncovid\_x\_test, covid\_x\_test), axis=0)

y\_train = np.concatenate((noncovid\_y\_train, covid\_y\_train), axis=0)

y\_test = np.concatenate((noncovid\_y\_test, covid\_y\_test), axis=0)

y\_train = LabelBinarizer().fit\_transform(y\_train)

y\_train = to\_categorical(y\_train)

y\_test = LabelBinarizer().fit\_transform(y\_test)

y\_test = to\_categorical(y\_test)

plot\_images(covid\_x\_train, 'X\_train')

plot\_images(covid\_x\_test, 'X\_test')

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(2, 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']

)

train\_aug = ImageDataGenerator(

rotation\_range=20,

width\_shift\_range=0.2,

height\_shift\_range=0.2,

horizontal\_flip=True

)

history = model.fit(train\_aug.flow(X\_train, y\_train, batch\_size=32),

validation\_data=(X\_test, y\_test),

validation\_steps=len(X\_test) / 32,

steps\_per\_epoch=len(X\_train) / 32,

epochs=500)

model.save('vgg\_chest.h5')

model.save\_weights('vggweights\_chest.hdf5')

model = load\_model('vgg\_chest.h5')

y\_pred = model.predict(X\_test, batch\_size=batch\_size)

prediction=y\_pred[0:10]

for index, probability in enumerate(prediction):

if probability[1] > 0.5:

plt.title('%.2f' % (probability[1]\*100) + '% COVID')

else:

plt.title('%.2f' % ((1-probability[1])\*100) + '% NonCOVID')

plt.imshow(X\_test[index])

plt.show()

y\_pred\_bin = np.argmax(y\_pred, axis=1)

y\_test\_bin = np.argmax(y\_test, axis=1)

fpr, tpr, thresholds = roc\_curve(y\_test\_bin, y\_pred\_bin)

plt.plot(fpr, tpr)

plt.xlim([0.0, 1.0])

plt.ylim([0.0, 1.0])

plt.rcParams['font.size'] = 12

plt.title('ROC curve for our model')

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.grid(True)

def plot\_confusion\_matrix(normalize):

classes = ['COVID','NonCOVID']

tick\_marks = [0.5,1.5]

cn = confusion\_matrix(y\_test\_bin, y\_pred\_bin,normalize=normalize)

sns.heatmap(cn,cmap='plasma',annot=True)

plt.xticks(tick\_marks, classes)

plt.yticks(tick\_marks, classes)

plt.title('Confusion Matrix')

plt.ylabel('True label')

plt.xlabel('Predicted label')

plt.show()

print('Confusion Matrix without Normalization')

plot\_confusion\_matrix(normalize=None)

print('Confusion Matrix with Normalized Values')

plot\_confusion\_matrix(normalize='true')

from sklearn.metrics import classification\_report

print(classification\_report(y\_test\_bin, y\_pred\_bin))

plt.figure(figsize=(10,10))

plt.plot(history.history['accuracy'])

plt.plot(history.history['val\_accuracy'])

plt.title('Model Accuracy')

plt.ylabel('Accuracy')

plt.xlabel('Epoch')

plt.legend(['Training', 'Testing'])

plt.savefig('vgg\_chest\_accuracy.png')

plt.show()

plt.figure(figsize=(10,10))

plt.plot(history.history['loss'])

plt.plot(history.history['val\_loss'])

plt.title('Model Loss')

plt.ylabel('Loss')

plt.xlabel('Epoch')

plt.legend(['Training', 'Testing'])

plt.savefig('vgg\_chest\_loss.png')

plt.show()

**CHAPTER 7**

**PERFORMANCE ANALYSIS**

**7.1 PERFORMANCE ANALYSIS**

There are several methods to evaluate a model’s performance. Accuracy, precision, recall, and F-score are the measures considered to estimate chest X-ray images.

**PRECISION:**

Precision is defined as the ratio of correctly predicted positive cases. This metric is a measure of exactness, which is calculated as the percentage of positive predictions of COVID-19 that were true positives divided by the number of predicted positives.

Precision =TP/TP + FP

**RECALL:**

The ratio of accurately detected positive cases is the recall. This metric is a measure of completeness, which is calculated as the percentage of positives that were correctly identified as true positives divided by the number of actual positives.

Recall = TP/TP + FN

**F1-SCORE:**

The F1-score is the harmonic mean of precision and recall. This is a combination of precision and recall that provides a significant measure for a test dataset that includes an imbalanced class.

F1 score = 2 x [Precision x Recall/Precision + Recall]

**ACCURACY:**

The percentage of correct predictions among the total number of predictions is called accuracy. This metric measures the percentage of correctly identified cases relative to the entire dataset. The CNN algorithm performs better if the accuracy is higher. Accuracy is a significant measure for a test dataset that includes a balanced class. Accuracy =TP + TN / [TP + FP + TN + FN

**CLASSIFICATION REPORT FOR CHEST X-RAY IMAGES**

**Table 7.1 Precision, Recall And F1-Score Of Xception**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-score | Support |
| 0 | 0.75 | 0.91 | 0.82 | 87 |
| 1 | 0.90 | 0.74 | 0.82 | 101 |
| accuracy |  |  | 0.82 | 188 |
| Macro avg | 0.83 | 0.83 | 0.82 | 188 |
| Weighted avg | 0.83 | 0.82 | 0.82 | 188 |

**Table 7.2 Precision, Recall And F1-Score Of Resnet50**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-score | Support |
| 0 | 0.96 | 0.86 | 0.91 | 87 |
| 1 | 0.89 | 0.97 | 0.93 | 101 |
| accuracy |  |  | 0.92 | 188 |
| Macro avg | 0.93 | 0.92 | 0.92 | 188 |
| Weighted avg | 0.92 | 0.92 | 0.92 | 188 |

**Table 7.3 Precision, Recall And F1-Score Of Inception**

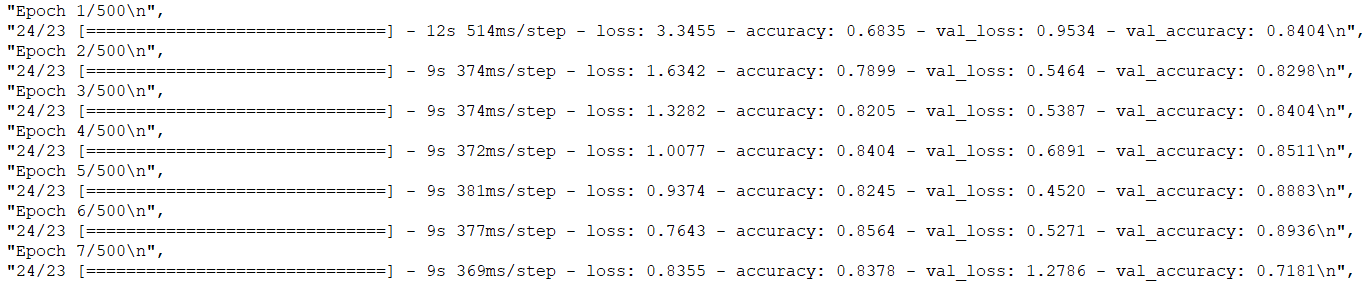
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-score | Support |
| 0 | 0.88 | 0.99 | 0.93 | 87 |
| 1 | 0.99 | 0.88 | 0.93 | 101 |
| accuracy |  |  | 0.93 | 188 |
| Macro avg | 0.93 | 0.93 | 0.93 | 188 |
| Weighted avg | 0.94 | 0.93 | 0.93 | 188 |

**Table 7.4 Precision, Recall And F1-Score Of Vgg16**

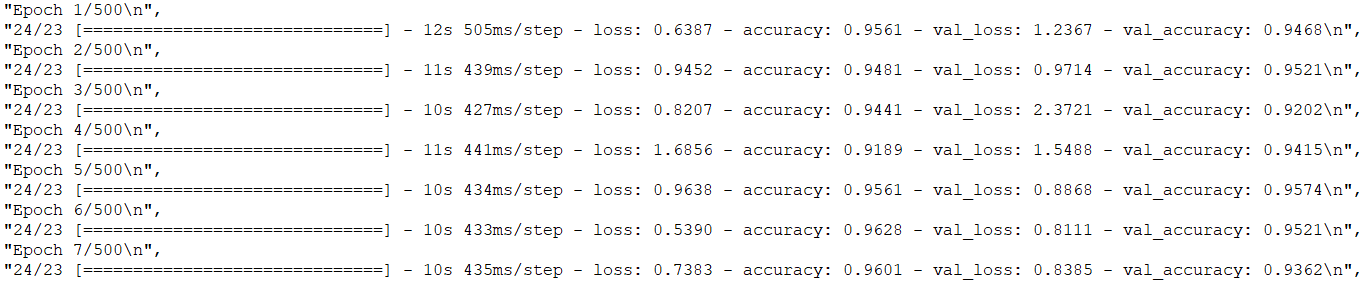
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-score | Support |
| 0 | 0.92 | 0.99 | 0.96 | 87 |
| 1 | 0.99 | 0.93 | 0.96 | 101 |
| accuracy |  |  | 0.96 | 188 |
| Macro avg | 0.96 | 0.96 | 0.96 | 188 |
| Weighted avg | 0.96 | 0.96 | 0.96 | 188 |

**7.2 RESULT AND DISCUSSION:**

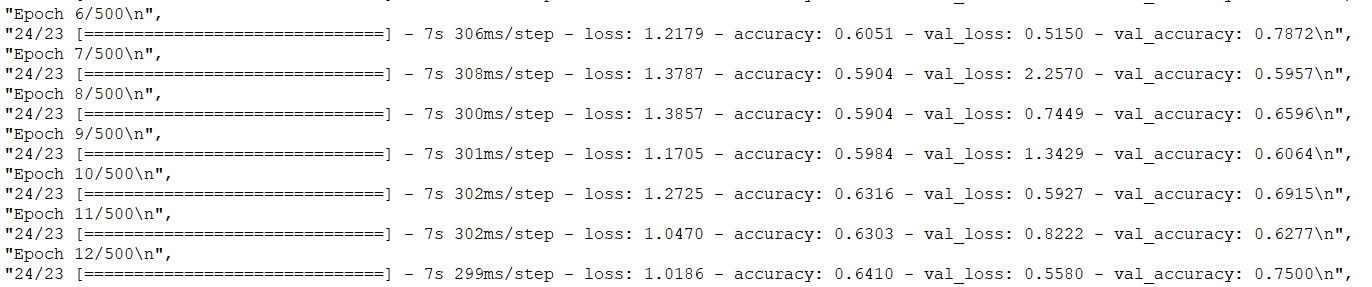
The COVID-19 is highly contagious, so controlling its transmission path effectively is crucial. The proposed work uses different deep learning algorithms (Xception, Inception, VGG, Resnet) to identify covid 19 from x-ray images. The training and testing phases are preceded by a preprocessing phase that includes data augmentation. In order to make an accurate prediction, evaluation metrics such as Precision,Recall,F1-score are used. Research findings indicated that Convolutional Neural Networks have the latent to detect respiratory diseases with the best accuracy, although a large amount images are needed, achieves an accuracy of 95% train accuracy and 98% of validation accuracy. By improving the networks, we can achieve 100% accuracy. The Accuracy metrics of Convolutional Neural Networks (CNNs) are shown below.



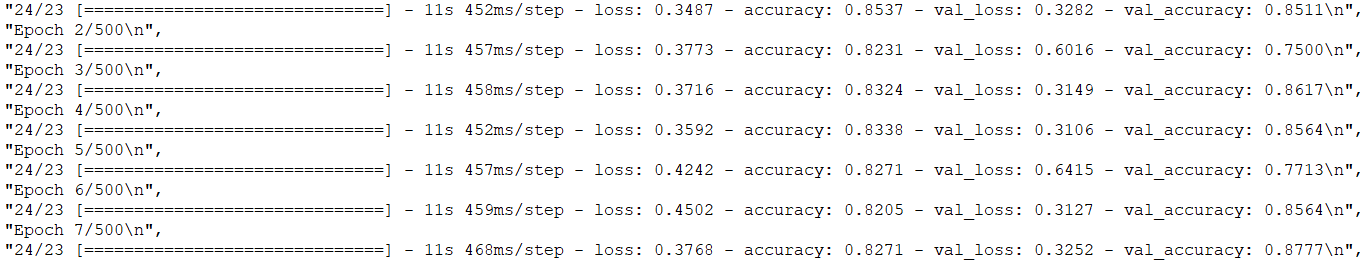
**FIG 8.1 Inception training model**

****

**FIG 8.2 Xception training model**

****

**FIG 8.3 Resnet training model**

****

**FIG 8.4 VGG training model**

The above figures describes the Test loss, test accuracy, train loss, train accuracy obtained after testing.

**CHAPTER 8**

**CONCLUSION**

**8.1 CONCLUSION AND FUTURE ENHANCEMENT:**

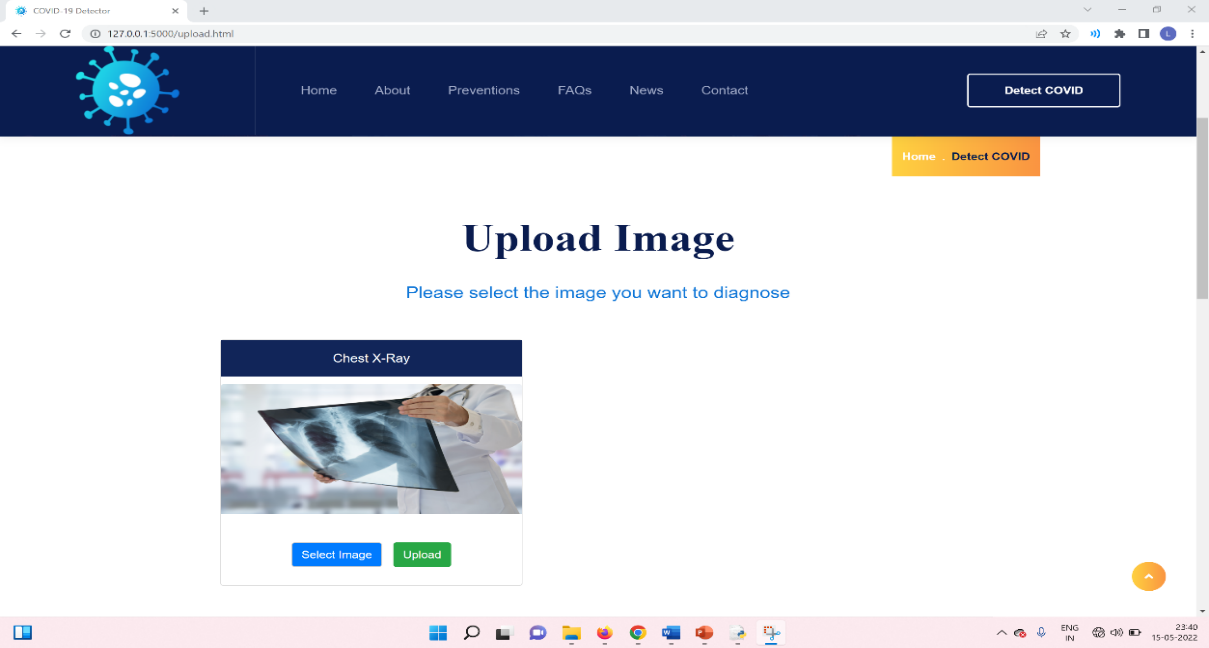
COVID-19 is a disease caused by the SARS-CoV-2 virus, which was found in Wuhan, China, in December 2019.  It is very contagious and has spread rapidly over the world.  COVID-19 is most commonly associated with respiratory symptoms that resemble a cold, flu, or pneumonia. Despite the fact that vaccines were launched at the beginning of 2021, there is a pressing need for quick and precise instruments to increase the healthcare system's efficiency. Our method successfully separated photos into two categories: COVID-19 positive and COVID-19 negative. Early detection of the novel coronavirus is critical to preventing the virus from spreading to others. We develop a deep transfer learning system that analyses chest X-ray pictures from patients with COVID-19 and patients without COVID-19 to detect the condition automatically. Doctors have limited time due to the enormous number of patients treated outside or in emergencies, and computer-aided analysis could save lives through early screening and proper therapy. This is especially useful in a pandemic, when existing health resources do not meet the severity of the disease or the necessity for preventive measures. The next step for this approach is to create a web page or a mobile app that the general public may utilise. We can also use GradCam tools to see the COVID-infected areas visually for improved outcomes. This model demonstrates that Convolutional Neural Networks can work wonders in the medical field as well.

**APPENDICES**

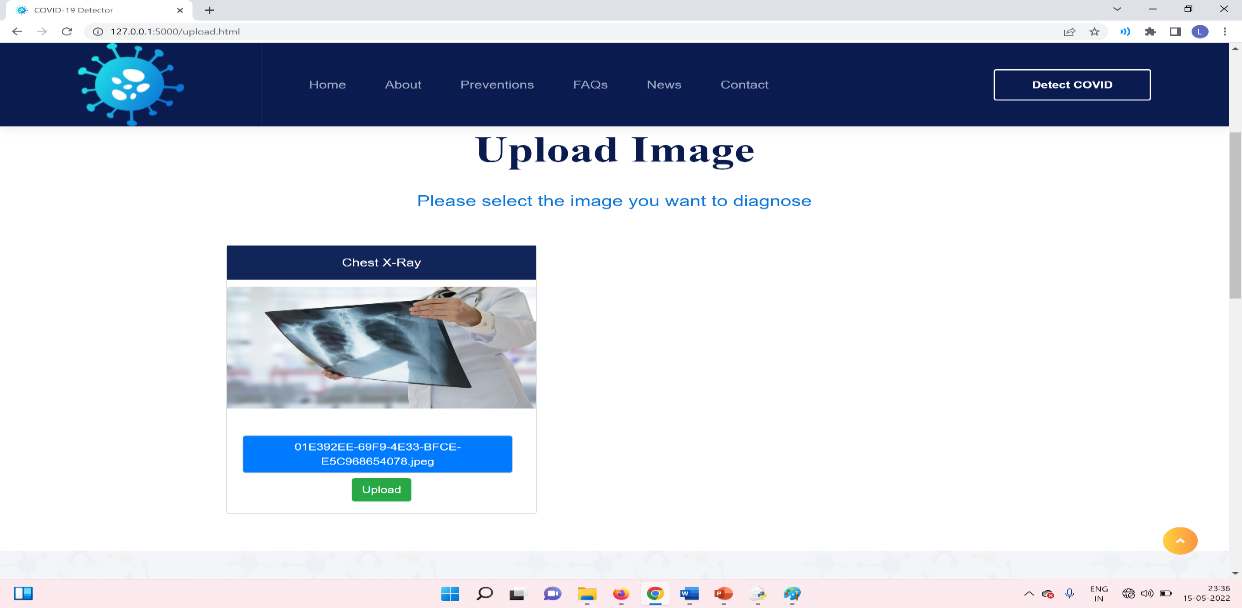
**A1.SAMPLE SCREENSHOTS:**

****

**FIG A.1 Home Page**



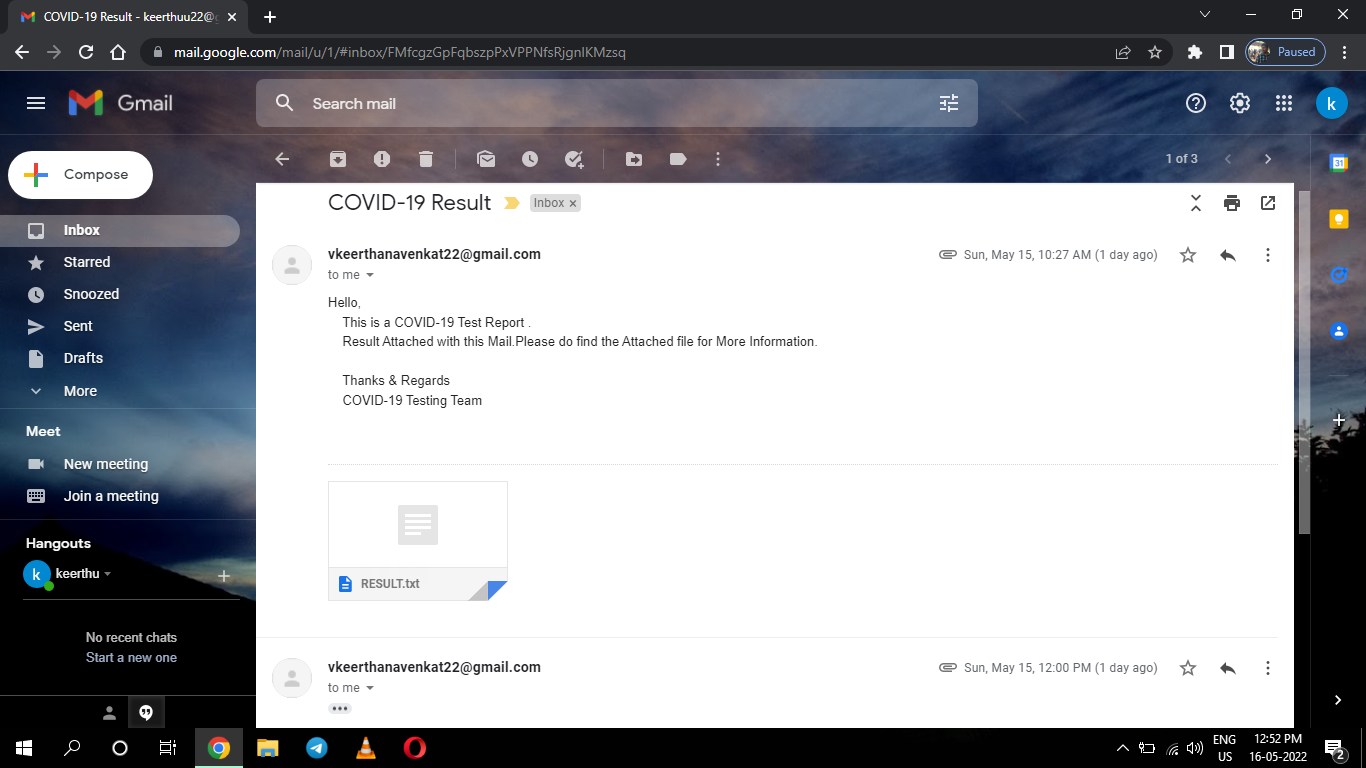
**FIG A.2 Upload Image**



**FIG A.3 View Uploaded Image**

****

**FIG A.4 Result Page**



**FIG A.5 Received Report Via Email**

**REFERENCES:**

[1] A. Abbas, M. M. Abdelsamea, and M. M. Gaber, “Classification of COVID-19 in chest x-ray images using DeTraC deep convolutional neural network.” Springer: Applied Intelligence, vol. 51, 2020.

[2]Aditya Saxena, Shamsheer Pal Singh, “Deep Learning Approach for the Detection of COVID-19 from Chest X-Ray Images using Convolutional Neural Networks”, arXiv: Electrical Engineering and Systems Science -Image and Video Processing,2022.

[3][Ameer Sardar Kwekha-Rashid](https://link.springer.com/article/10.1007/s13204-021-01868-7#auth-Ameer_Sardar-Kwekha_Rashid), [Heamn N. Abduljabbar](https://link.springer.com/article/10.1007/s13204-021-01868-7#auth-Heamn_N_-Abduljabbar) & [Bilal Alhayani](https://link.springer.com/article/10.1007/s13204-021-01868-7#auth-Bilal-Alhayani) ”Coronavirus disease (COVID-19) cases analysis using machine-learning applications”, Springer: Applied nano science,2021.

[4] Amir Rehman, Muhammad Azhar Iqbal, Irfan Ahmed , "COVID-19 Detection Empowered with Machine Learning and Deep Learning Techniques: A Systematic Review" , mdpi: Applied Science, volume 11 2021.

# [5] Soumya Ranjan Nayak, Deepak Ranjan Nayak, Utkarsh Sinha, Vaibhav Arora, Ram Bilas Pachori”,Application of deep learning techniques for detection of COVID-19 cases using chest X-ray images: A comprehensive study”, Science direct: Journal of Biomedical Signal Process Control, volume 64,2020.

[6] Boran Sekeroglu and Ilker Ozsahin,"Detection of COVID-19 from Chest X-Ray Images Using Convolutional Neural Networks", Original research, volume 25,2020.

[7] Chuansheng Zheng, Xianbo Deng, Qiang Fu, Qiang Zhou, Jiapei Feng, Hui Ma, Wenyu Liu, Xinggang Wang, “Deep Learning-Based Detection for COVID-19 from Chest CT Using Weak Label”, IEEE Transactions on Medical Imaging, 2020.

# [8] <https://www.w3schools.com/python/python_reference.asp> Python reference

[9] Joy Iong-Zong Chen, "Design of Accurate Classification of COVID-19 Disease in X-Ray Images Using Deep Learning Approach", Journal of ISMAC,volume 3,2021

[10] Md. Rezaul Karim,Till DÃűhmen,Dietrich Rebholz-Schuhmann,Stefan Decker,Michael Cochez,Oya Beyan, “DeepCOVIDExplainer: Explainable COVID-19 predictions based on chest X-ray images,” ResearchGate: Molecular Biology,2020.

[11] Mohammad Rahimzadeh , Abolfazl Attar, "A modified deep convolutional neural network for detecting COVID-19 and pneumonia from chest X-ray images based on the concatenation of Xception and ResNet50V2 ", Informatics in medicine ,volume 19,2020.

[12] Moutaz Alazab, Albara Awajan , Abdelwadood Mesleh, Ajith Abraham, Vansh Jatana, Salah Alhyari, "COVID-19 Prediction and Detection Using Deep Learning", International Journal of Computer Information Systems and Industrial Management Applications. ,Volume 12,2020.

[13] Parag Chatterjee Mainak Biswas, and Arnab Kumar Das,"Specialized covid-19 detection techniques with machine learning", Journal of Physics: Conference Series,2021.

[14] Parnian Afshar, Shahin Heidarian, Farnoosh Naderkhani, Anastasia Oikonomou, Konstantinos N. Plataniotis, Arash Mohammadi ,"COVID-CAPS: A Capsule Network-based Framework for Identification of COVID-19 cases from X-ray Images", Journal of computer vision and pattern recognition,2020.

[15] Pillalamarry Mahesh, Yakkala Gnana Prathyusha, Botlagunta Sahithi, S Nagendram,"Covid-19 Detection from Chest X-Ray using Convolution Neural Networks", Journal of Physics: Conference Series, Volume 1804, International Conference of Modern Applications on Information and Communication Technology (ICMAICT),2020.

[16] S. Asif, Y. Wenhui, H. Jin, Y. Tao, and S. J. M. Jinhai, “ Classification of Covid-19 from Chest X-ray Images Using Deep Convolutional Neural Networks”, MedRxiv, Cold Spring, NY, USA, 2020.

[17] [Shamima Akter](https://sciprofiles.com/profile/1576596), [F. M. Javed Mehedi Shamrat](https://sciprofiles.com/profile/1757506) and [Sovon Chakraborty](https://sciprofiles.com/profile/1833660),Asif Karim, Sami Azam “COVID-19 Detection Using Deep Learning Algorithm on Chest X-ray Image”,Biology,2021.

## [18] Deep Learning From Scratch: Building with Python from First Principles by Seth Weidman published by O`Reilley

[19] S.V.Kogilavani , J. Prabhu, R. Sandhiya, M. Sandeep Kumar, UmaShankar Subramaniam, Alagar Karthick ,M. Muhibbullah and Sharmila Banu Sheik Imam,"COVID-19 Detection Based on Lung Ct Scan Using Deep Learning Techniques",Hindawi, Computational and Mathematical Methods in Medicine ,Volume 2022, 2022.

[20] S. Tabik , A. Gómez-Ríos, J. L. Martín-Rodríguez, I. Sevillano-García, M. Rey-Area, D. Charte,E. Guirado, J. L. Suárez, J. Luengo, vM. A. Valero-González, P. García-Villanova,E. Olmedo-Sánchez, and F. Herrera, "COVIDGR Dataset and COVID-SDNet Methodology for Predicting COVID-19 Based on Chest X-Ray Images", IEEE Journal of biomedical and health informatics, 2020.

[21] Shuai Wang, Bo Kang, Jinlu Ma, Xianjun Zeng, Mingming Xiao,Jia Guo, Mengjiao Cai, Jingyi Yang, Yaodong Li, Xiangfei Meng, "A deep learning algorithm using CT images to screen for Corona virus disease (COVID-19)", Imaging informatics and artificial intelligence,2021.

[22] Tulin Ozturk, Muhammed Talo,Eylul Azra, Yildirim ,Ulas BaranBaloglu ,Ozal Yildirim, U.Rajendra Acharya "Automated detection of COVID-19 cases using deep neural networks with X-ray images", computers in biology and medicine,volume 121,2020

[23] Y. Song,” Deep Learning Enables Accurate Diagnosis of Novel Coronavirus (COVID-19) with CT Images”, MedRxiv, IEEE /ACM Transactions on computational biology and bio informatics, vol. 19, 2020.

[24] Wentao Zhao , Wei Jiang1 & Xinguo Qiu ,"Deep learning for COVID‑19 detection based on CT images", scientific reports,2021.

[25] Wassim Zouch, Dhouha Sagga, Amira Echtioui, Rafik Khemakhem, Mohamed Ghorbel, Chokri Mhiri & Ahmed Ben Hamida, "Detection of COVID-19 from CT and Chest X-ray Images Using Deep Learning Models", Journal of biomedical engineering,2022.