A PROJECT REPORT ON

Classification of Covid-19 Images with Severity Scoring

SUBMITTED TO MIT SCHOOL OF ENGINEERING
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE

BACHELOR OF TECHNOLOGY
(Computer Science & Engineering)

BY

Akanksha Prasad	MITU21BTCS0061
Aarya Ranjit	MITU21BTCS0008
Aarya Kulkarni	MITU21BTCS0009
Harshal Jagtap	MITU21BTCS0235

Under The Guidance of

Mr. Pratik Kamble

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
MIT School of Engineering
MIT Art, Design and Technology University
Rajbaug Campus, Loni-Kalbhor, Pune 412201

2022-23

MIT SCHOOL OF ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
MIT ART, DESIGN AND TECHNOLOGY UNIVERSITY,
RAJBAUG CAMPUS, LONI-KALBHOR, PUNE 412201

CERTIFICATE

This is to certify that the Project Entitled

Classification of Covid-19 Images with Severity Scoring

Submitted by

Akanksha Prasad	MITU21BTCS0061
Aarya Ranjit	MITU21BTCS0008
Aarya Kulkarni	MITU21BTCS0009
Harshal Jagtap	MITU21BTCS0235

is a bonafide work carried out by them under the supervision of Mr. Pratik Kamble and it is submitted towards the partial fulfillment of the requirement of MIT ADT University, Pune for the award of the degree of Bachelor of Technology (Com-puter Science and Engineering).

Mr. Pratik Kamble	Prof. HOD Name	Dr. Rajneeshkaur Sachdeo
Internal Guide	H.O.D	Director
Department of CSE	Department of CSE	MIT SoE

On Company Letterhead/seal

CERTIFICATE

This is to certify that the Project Entitled

BE PROJECT TITLE
Classification of Covid-19 Images with Severity Scoring

Submitted by

Akanksha Prasad	MITU21BTCS0061
Aarya Ranjit	MITU21BTCS0008
Aarya Kulkarni	MITU21BTCS0009
Harshal Jagtap	MITU21BTCS0235

is a bonafide work carried out by them (with the Sponsorship from ————) under the supervision of Mr/Ms/Miss. .... and has been completed successfully.

Mr. Guide Name
Designation:
External Guide:
Seal and Stamp of the company

DECLARATION

We, the team members

Akanksha Prasad	MITU21BTCS0061
Aarya Ranjit	MITU21BTCS0008
Aarya Kulkarni	MITU21BTCS0009
Harshal Jagtap	MITU21BTCS0235

Hereby declare that the project work incorporated in the present project entitled **Classification of Covid-19 Images with Severity Scoring** is original work. This work (in part or in full) has not beensubmitted to any University for the award or a Degree or a Diploma.We have properly acknowledged the material collected from secondary sources wherever required.Wesolely own the responsibility for the originality of the entire content.

Date: Name & Signature of the Team Members

Akanksha Prasad

Aarya Ranjit

Aarya Kulkarni

Harshal Jagtap

Name and Signature of Guide

Seal/Stamp of the college
Place:
Date:

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
MIT SCHOOL OF ENGINEERING
RAJBAUG, LONI KALBHOR
PUNE – 412201

EXAMINER’S APPROVAL CERTIFICATE

The project report entitled submitted by Student 1 (MITU17BTCS0008), Student 2 (MITU17BTCS0009), Student 3 (MITU17BTCS0061) and Student 4(MITU21BTCS0235) in partial fulfillment for the award of the degree of Bachelor of Technology (Computer Science & Engineering) during the academic year 2022-23, of MIT-ADT University, MIT School of Engineering, Pune, is hereby approved.

Examiners

Examiner 1 Name and Signature :

Examiner 2 Name and Signature:

Acknowledgments

It gives us great pleasure in presenting the project report on **'Classification of Covid-19 Images with Severity Scoring'**

We would like to take this opportunity to thank my internal guide ***Prof. Pratik Kamble*** *for giving me all the help and guidance I needed. I am really grateful to them for their kind support. Their valuable suggestions were very helpful.*

We are also grateful to ***Prof. HOD Name****, Head of Computer Science & Engineering indispensable support, suggestions.*

In the end our special thanks to ***Other Person Name*** *for providing various resources such as laboratory with all needed software platforms,continuous Internet connection, for Our Project.*

Akanksha Prasad
Aarya Ranjit
Aarya Kulkarni
Harshal Jagtap
(B.Tech. Computer Science & Engineering)

Abstract

Since the outbreak of Covid-19 on 11th March 2020,a lot of research were done for exact, quick and point of care testing with quick outcomes for detection ,isolation and diagnosis.

Initially we used RT-PCR tests for detecting Covid-19 ,but there was a risk of transmission of the virus while collecting the samples. To predict the accurate severity of Covid-19 and its effects on Human lungs ,to understand the condition of lungs and diagnosis methods medical experts need an accurate precision.

We make use of image classification techniques convolutional networks to analyze raw images of COVID-19 affected human lungs in this project for the detection of pulmonary abnormalities in the early stage, Chest X-Ray is not exact. So, we are using CNN model especially for identifying the stages of lung abnormalities. We present normal and covid with the scoring systems which cause the severity in lungs of COVID-19 patients every day. This will be accurate for predicting the stages of COVID-19 for early treatment and also to protect the uninfected population.

The goal of this project is to classify Chest X-ray images symptoms by brixia scores using Convolutional neural networks. The idea is we can input an image, define a weight matrix, and the input is convolved to extract features from the image without losing the information about its spatial arrangement.

Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

1.1 RELEVANCE

- The project aims to develop content in the Covid-19 category. To control, manage and prevent pandemic from affecting people through contact. Also to help for effective diagnosis methods according to severity. These days the clinical facilities such as the intensive care units and the mechanical ventilators are very limited due to which it becomes quite important to classify the patients according to their severity levels.

1.2 MOTIVATION OF THE PROJECT

- To reduce the dependencies of Doctors for detection and diagnosis of Covid-19.
- To make use of classification models for detecting abnormality in lungs.

1.3 PROBLEM STATEMENT

- Using a Convolutional Neural Network, we analyse images (X-Rays of Chests) and identify whether the patient is COVID-19 positive or not.

1.4 OBJECTIVES

- The goal of this project is to classify Chest X-ray images symptoms by brixia scores using Convolutional neural networks.With positive and negative cases a severity can be determined from Chest X -Rays.
-

1.5 SCOPE

- The idea of implementating modern technologies such as AI and ML in healthcare industries is rapidly growing in popularity.
- **Basic idea**:-The goal of this project is to classify Chest X-Ray images as normal(healthy) or Covid cases using a Convolutional Neural Networks. CNN is a type of Deep learning algorithm that performs very well in image classification tasks.
- **Simple training model**:- The VGG-16 training model allows us to train the algorithm with a simpler and easier-to-implement code compared to other training models while giving us approx. 99% accuracy.

1.6 ORGANIZATION OF THE REPORT

1)Literature Survey
2)Software Requirements
3)Project and Its implementations
4)System Testing
5)Conclusion
6)References

CHAPTER 2

LITERATURE SURVEY

2.1RELATED WORK

There has been extensive research carried out across the globe on Image Recognition using Convolutional Neural Networks. Since the advent of the COVID-19 pandemic, efforts have been taken to use CNNs to help identify COVID-19 cases from chest X-Rays. Since COVID-19 affects the lungs most adversely, it is a good use case for CNNs to reduce the manual work of doctors in analysing chest X-Rays by themselves.

Numerous papers have been written on related projects and their execution. COVID-Net, an architecture based on transfer learning to classify X-Ray images into 4 categories: normal, bacterial infection, non-COVID and COVID- 19 viral infection, was used by Wang L, Wong A in their paper in 2020. But, their work contained data irregularities, and to account for that, Asmaa Abbas, Mohammed M. Abdelsamea & Mohamed Medhat Gaber wrote about using DeTraC https://link.springer.com/article/10.1007/s10489-020-01829-7

Their work is referenced by Neha Rajawat, Bharat Singh Hada, Mayank Meghawat, Soniya Lalwani & Rajesh Kumar in their work on the paper below
who also use COVID-Net to classify the X-Ray images.
https://link.springer.com/article/10.1007/s13369-022-06841-2

In all the above work, we found that the process of preparing datasets and setting up the CNN models was quite complex, and inaccessible to students or people with systems which may not have the power to run large and complex codes. Hence, we looked at simpler pre-processing methods and training models in the following paper by Agata Giełczyk, Anna Marciniak, Martyna Tarczewska, Zbigniew Lutowski
https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0265949
and the following Kaggle project and dataset by Manasij Haldar
https://www.kaggle.com/code/therockomanz/covid-19-detection-2/notebook#DenseNet169

2.1 COMPARISON OF EXISTING WORK

Most projects found online use models such as ResNet-18, DenseNet169, InceptionV3, DeTraC, etc. These models, though accurate in their working, are quite complex and challenging to run, especially on older hardware, which is usually found in hospitals and clinics. Hence, our project uses the VGG-16 model, which provides satisfactory accuracy (approx. 99%), while being relatively simple to implement.

Also, finding a publicly available dataset on Kaggle allows the training of the model to be implemented faster, as less pre-processing is required and more time could be given to the other aspects of the project.

2.3 Gap Organisation

Our main goal was to simplify the processes used by earlier work, while maintaining a satisfactory rate of accuracy. We have achieved this by using the VGG-16 and Inception models to process the X-Ray images and train our algorithm.

Using a publicly available dataset also allows our work to be easily peerreviewable, as other teams with different models may use the same dataset for testing purpose

CHAPTER 3

SOFTWARE REQUIREMENT SPECIFICATION

3.1 INTRODUCTION

3.2 PURPOSE AND SCOPE OF DOCUMENT

We could have used normal ANN but we are using CNN because both are unique in how they work mathematically, and this causes them to be better at solving specific problems. In general, CNN tends to be a more powerful and accurate way of solving classification problems. ANN is still dominant for problems where datasets are limited, and image inputs are not necessary. A CNN is a kind of network architecture for deep learning algorithms and is specifically used for image recognition and tasks that involve the processing of pixel data. There are other types of neural networks in deep learning, but for identifying and recognizing objects, CNNs are the network architecture of choice.

3.3 GENERAL DESCRIPTION

When we deploy our project to a web service we can take the input from user,then simply press submit button and the classifier functions accordingly and we determine whether the case ids positive and negative.

3.4 FUNCTIONAL REQUIREMENTS

Functioning of classifying models are required such as VGG16,Inception V3 and Sequential Model. a pre-trained model is a model created by some one else to solve a similar problem. Instead of building a model from scratch to solve a similar problem, you use the model trained on other problem as a starting point.

3.5 INTERFACE REQUIREMENTS

For FrontEnd,we require a library named as Flask, which will enable us to deploy our CNN Model, Flask is an API of Python that allows us to build up web-applications.We used some templates which are base.html and index.html, for our main page and taking inputs from the user for an image.

3.6 PERFORMANCE REQUIREMENTS

In the project first we create a CNN Model ,then we determine the positive and negative cases using the predict function defined for the model ‘Classifier’.Then we create another python notebook for deployment and put our saved model in the notebook through which the functions for predictions work on web service.

3.7 DESIGN CONSTRAINTS

For the software limitations are there that train and test data can be divided into 0.7 and 0.25 because the sum of train and test sum should be less than 1.The input shape which we give our model should be compatible with the dataset.

3.8 NON-FUNCTIONAL ATTRIBUTES

When we have a dataset in CNN we need to have it in a default output way,i.e,we need a categorized data that belongs to some certain classes,then only the model will be able to calassify the images

.

3.9 NON-FUNCTIONAL REQUIREMENTS

- Categorised Dataset
- The accuracy of the training and testing data should be at least 84% or 96%

.

3.10 OVERVIEW OF RESPONSIBILITIES OF DEVELOPER

1)Importing the necessary libraries.

Importing the libraries Sequential,VGG16,MaxPool,Conv2D,Flatten and Dense. In case if we don’t have some libraries we can install them using ‘pip-install’.

2)Preparing the dataset

For The output we need to have a proper dataset so we need to categorise our dataset and then we need to import it in our google collab.

3)Training the Models

We need to train certain models for classification, we need to resize image accordingly and also the data should be in proper annotated way (i.e. there should be some segmentation).

4)Creating the models.

We need to add some convolutional layers like Dense, Flatten and Conv2d and give input shape, epochs, batch size accordingly.

5)Classifying the images

After we are done making the models we can simply use a predict function to classify the images. So, we will consider the results and define the classes for Covid and Normal Images, i.e, for Covid class is 1 and for Normal class is 0.

6)Deployment of the project

We deploy the project using Flask library.We define the name of the app and import our model by which we have saved it to device.After that we create some templates like base.html and index.html for creating our web pages and to make it look systematic.

CHAPTER 4

PROJECT DESIGN AND IMPLEMENTATION

4.1 ARCHITECTURAL DIAGRAM

A description of the system architecture and architectural diagram needs to be presented.

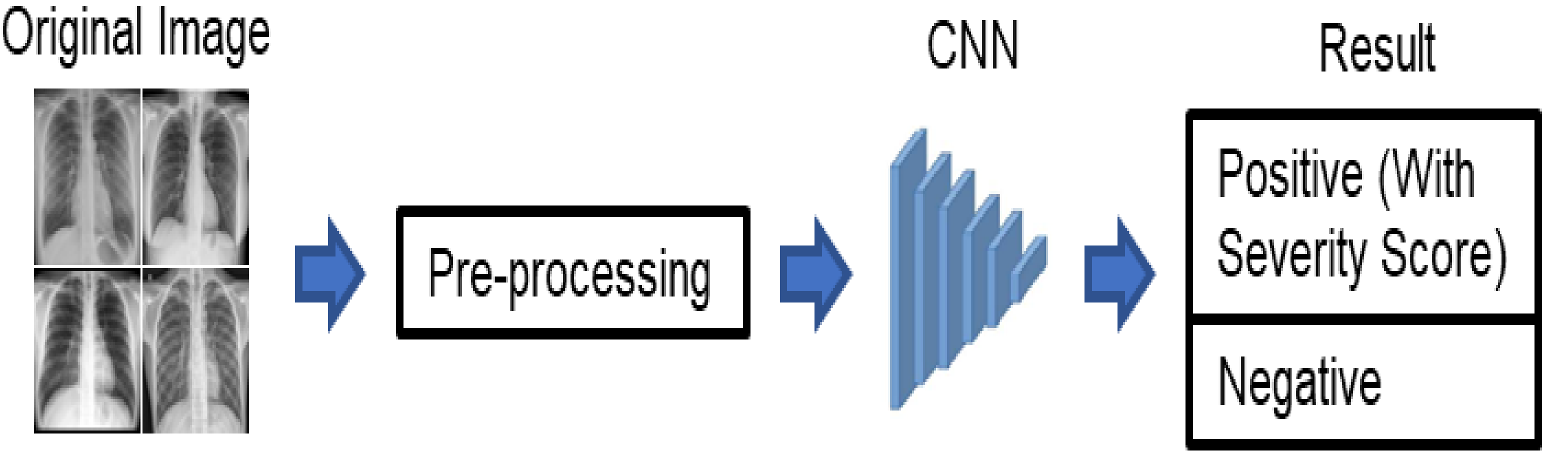


Figure 4.1: Architecture diagram

4.2 METHODOLOGY

After conducting research through sources like IEEE, Springer, Kaggle, etc. and looking at the current global situation with the COVID-19 pandemic, we formulated the problem statement. We decided that we would implement a CNN using the VGG-16 and InceptionV3 models to detect COVID-19 cases by analysing chest X-Ray images.

The processes followed in implementing our solution are as follows:

1)Importing the necessary libraries.

Importing the libraries Sequential,VGG16,MaxPool,Conv2D,Flatten and Dense. In case if we don't have some libraries we can install them using 'pip-install'.

2)Preparing the dataset

For The output we need to have a proper dataset so we need to categorise our dataset and then we need to import it in our google collab.

3)Training the Models

We need to train certain models for classification, we need to resize image accordingly and also the data should be in proper annotated way (i.e. there should be some segmentation).

4)Creating the models.

We need to add some convolutional layers like Dense, Flatten and Conv2d and give input shape, epochs, batch size accordingly.

5)Classifying the images

After we are done making the models we can simply use a predict function to classify the images.
So, we will consider the results and define the classes for Covid and Normal Images, i.e, for Covid class is 1 and for Normal class is 0.

6)Deployment of the project

We deploy the project using Flask library.We define the name of the app and import our model by which we have saved it to device.After that we create some templates like base.html and index.html for creating our web pages and to make it look systematic.

4.3 USAGE SCENARIO

This section provides various usage scenarios for the system to be developed.

4.3.1 User profiles

Our project will mainly be used by doctors in hospitals and clinics. Instead of having to collect nose swab samples, risking contamination and spending 24-72 hours in processing and delivering the result, doctors can simply take an X-Ray image of a chest, which takes just a few minutes, and run it through our algorithm. The whole process of COVID-19 detection is made safer and more efficient for them.

Patients whose X-Rays are run through the algorithm to detect COVID-19 will get their results in a fraction of the time taken by RT-PCR testing, while being cheaper too. This clearly benefits the end users, both doctors and patients.

4.3.2 Use-cases

All use-cases for the software are presented. Description of all main Use cases using use case template is to be provided.

Sr No.	Use Case	Description	Actors	Assumptions
1	Quick Detection of Covid-19	The CNN algorithm helps in quick detection of Covid-19	Doctors and Physicians	We have assumed some classes in dataset for showing results

Table 4.1: Use Cases

4.3.3 Use Case View

Use Case Diagram. Example is given below

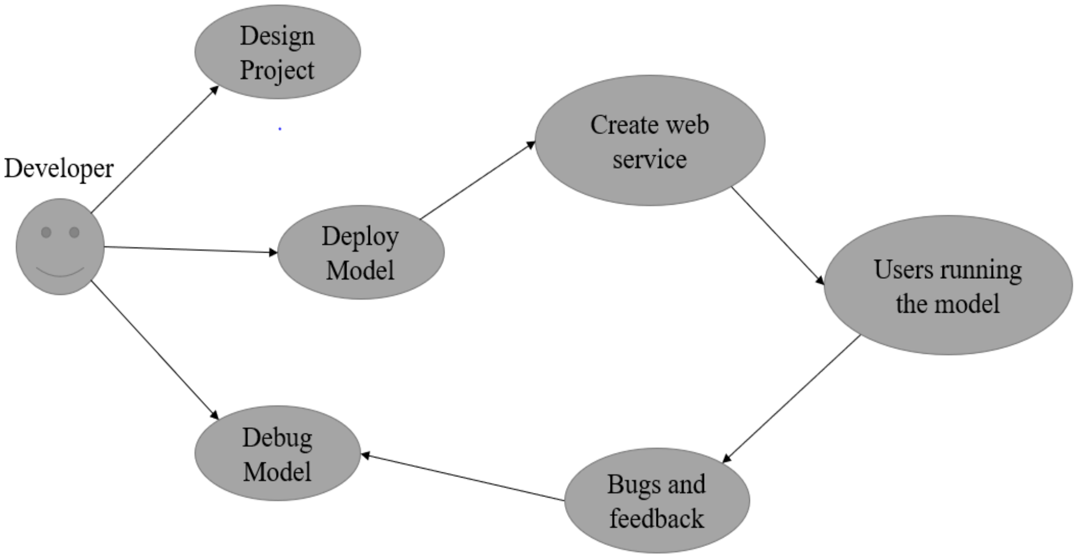


Figure 4.2: Use case diagram

4.4 DATA MODEL AND DESCRIPTION

4.4.1 Data Description

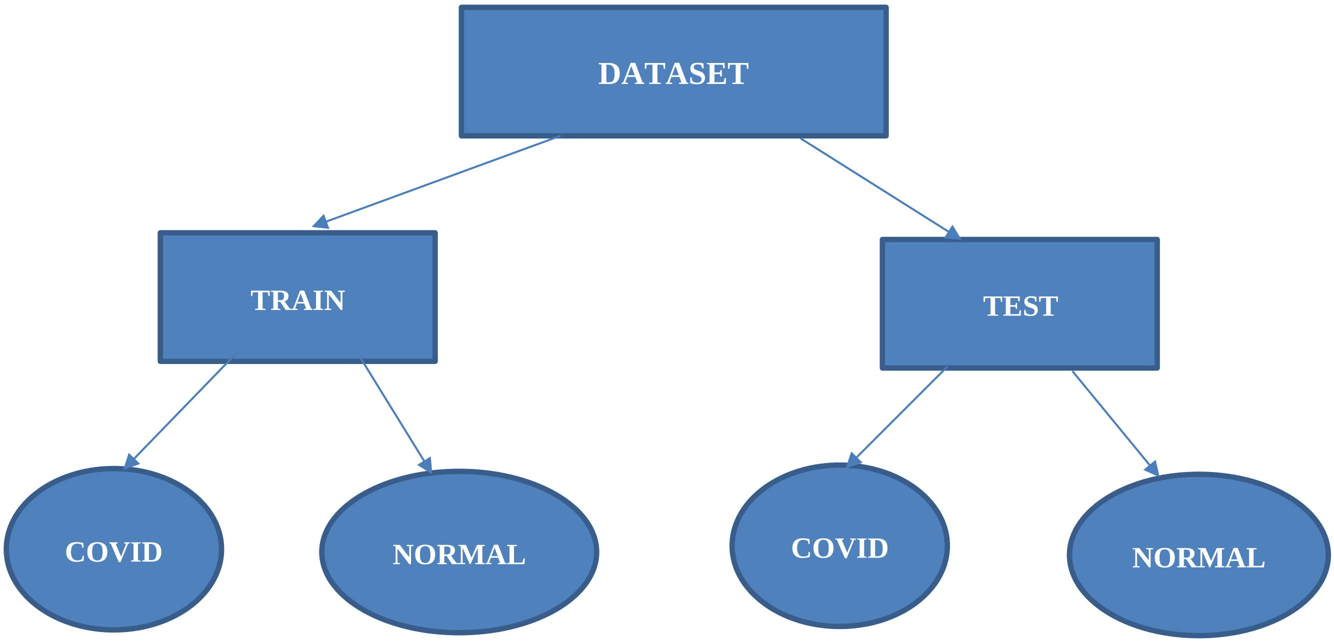
It is a simple directory structure branched into test and train and further branched into the respective 3 classes which contains the images. It contains around 137 cleaned images of COVID-19 and 317 in total containing Viral Pneumonia and Normal Chest X-Rays structured into the test and train directories.
The dataset consists 251 training images which belongs to 3 classes,66 training images which belongs to 3 classes.The three classes are Covid,Normal and Pneumonia,out of which we are only using Covid and Normal Images.

4.4.2 Data objects and Relationships

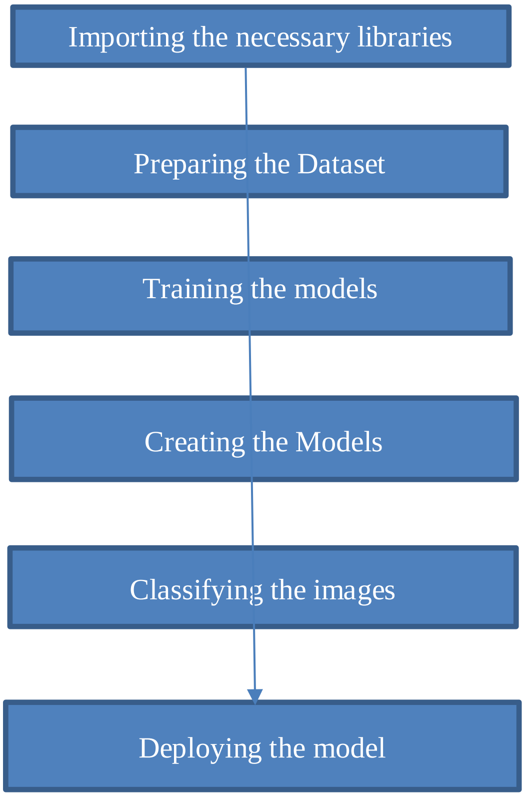
Dataset			
Train		Test	
Covid	Normal	Covid	Normal
111	70	26	20

The Train Dataset has two attributes Covid and Normal and has 181 images and Test Datsaset has two attributes Covid and Normal and has 46 images .

4.4.3 Data Flow Diagram



4.4.4 Activity Diagram:

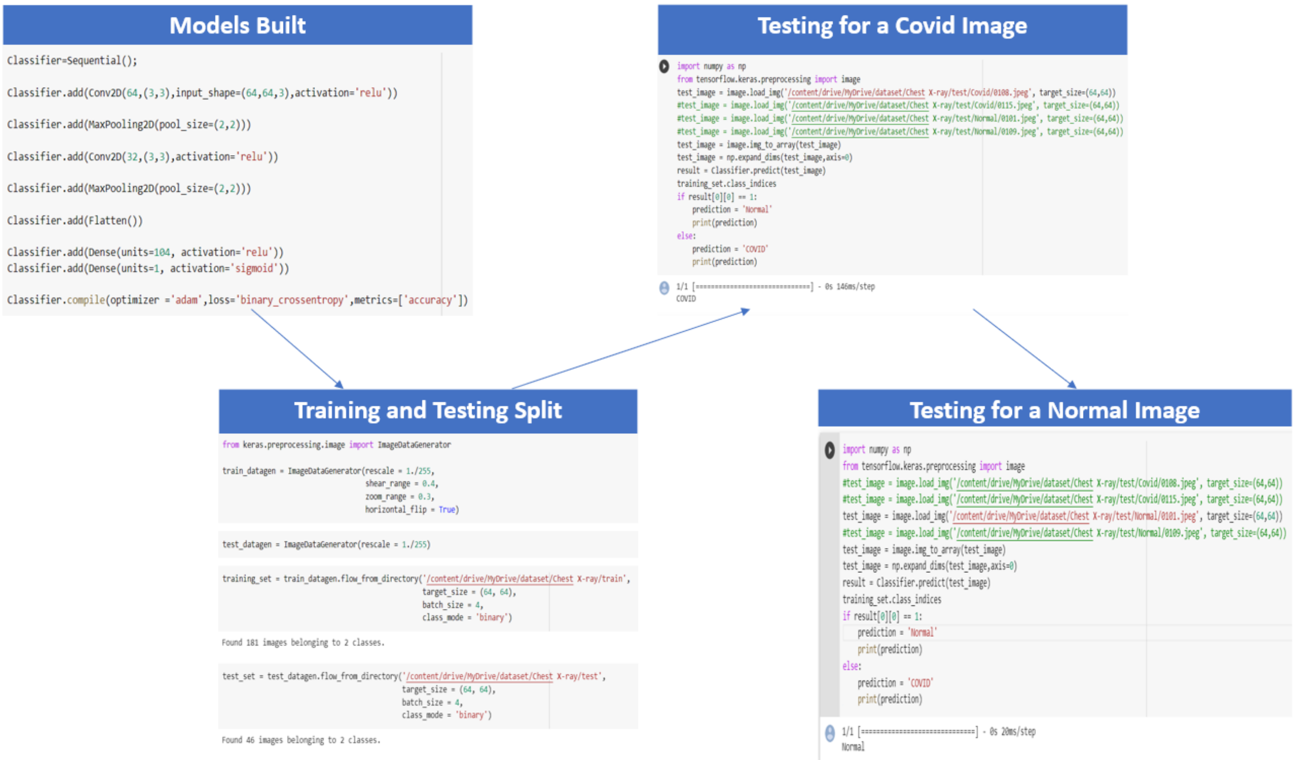


4.5 CLASS DIAGRAM

Class diagrams, Interaction Diagrams, Algorithms. Description of each component description required.

4.5.1 Class Diagram

For Backend



Models Built

```
Classifier=Sequential();

Classifier.add(Conv2D(64,(3,3),input_shape=(64,64,3),activation='relu'))

Classifier.add(MaxPooling2D(pool_size=(2,2)))

Classifier.add(Conv2D(32,(3,3),activation='relu'))

Classifier.add(MaxPooling2D(pool_size=(2,2)))

Classifier.add(Flatten())

Classifier.add(Dense(units=104, activation='relu'))
Classifier.add(Dense(units=1, activation='sigmoid'))

Classifier.compile(optimizer ='adam',loss='binary_crossentropy',metrics=['accuracy'])
```

Testing for a Covid Image

```
import numpy as np
from tensorflow.keras.preprocessing import image
test_image = image.load_img('/content/drive/MyDrive/dataset/Chest X-ray/test/Covid/0108.jpeg', target_size=(64,64))
#test_image = image.load_img('/content/drive/MyDrive/dataset/Chest X-ray/test/Covid/0115.jpeg', target_size=(64,64))
#test_image = image.load_img('/content/drive/MyDrive/dataset/Chest X-ray/test/Normal/0101.jpeg', target_size=(64,64))
#test_image = image.load_img('/content/drive/MyDrive/dataset/Chest X-ray/test/Normal/0109.jpeg', target_size=(64,64))
test_image = image.img_to_array(test_image)
test_image = np.expand_dims(test_image,axis=0)
result = Classifier.predict(test_image)
training_set.class_indices
if result[0][0] == 1:
    prediction = 'Normal'
    print(prediction)
else:
    prediction = 'COVID'
    print(prediction)
```

```
1/1 [==============================] - 0s 146ms/step
COVID
```

Training and Testing Split

```
from keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.4,
                                   zoom_range = 0.3,
                                   horizontal_flip = True)
```

```
test_datagen = ImageDataGenerator(rescale = 1./255)
```

```
training_set = train_datagen.flow_from_directory('/content/drive/MyDrive/dataset/Chest X-ray/train',
                                                 target_size = (64, 64),
                                                 batch_size = 4,
                                                 class_mode = 'binary')
```

```
Found 181 images belonging to 2 classes.
```

```
test_set = test_datagen.flow_from_directory('/content/drive/MyDrive/dataset/Chest X-ray/test',
                                            target_size = (64, 64),
                                            batch_size = 4,
                                            class_mode = 'binary')
```

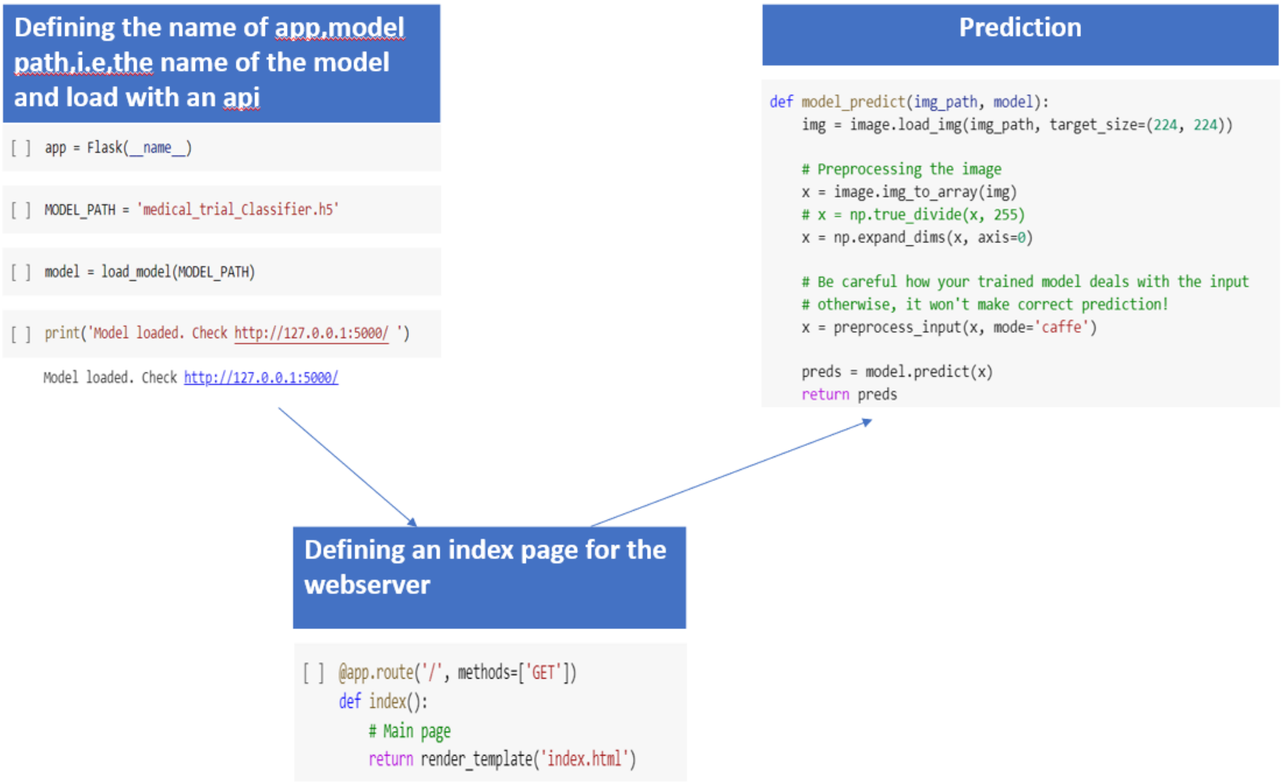
```
Found 46 images belonging to 2 classes.
```

Testing for a Normal Image

```
import numpy as np
from tensorflow.keras.preprocessing import image
#test_image = image.load_img('/content/drive/MyDrive/dataset/Chest X-ray/test/Covid/0108.jpeg', target_size=(64,64))
#test_image = image.load_img('/content/drive/MyDrive/dataset/Chest X-ray/test/Covid/0115.jpeg', target_size=(64,64))
test_image = image.load_img('/content/drive/MyDrive/dataset/Chest X-ray/test/Normal/0101.jpeg', target_size=(64,64))
#test_image = image.load_img('/content/drive/MyDrive/dataset/Chest X-ray/test/Normal/0109.jpeg', target_size=(64,64))
test_image = image.img_to_array(test_image)
test_image = np.expand_dims(test_image,axis=0)
result = Classifier.predict(test_image)
training_set.class_indices
if result[0][0] == 1:
    prediction = 'Normal'
    print(prediction)
else:
    prediction = 'COVID'
    print(prediction)
```

```
1/1 [==============================] - 0s 20ms/step
Normal
```

For Frontend:-



Defining the name of app,model path,i.e,the name of the model and load with an api

```
[ ]  app = Flask(__name__)
```

```
[ ]  MODEL_PATH = 'medical_trial_Classifier.h5'
```

```
[ ]  model = load_model(MODEL_PATH)
```

```
[ ]  print('Model loaded. Check http://127.0.0.1:5000/ ')

     Model loaded. Check http://127.0.0.1:5000/
```

Prediction

```
def model_predict(img_path, model):
    img = image.load_img(img_path, target_size=(224, 224))

    # Preprocessing the image
    x = image.img_to_array(img)
    # x = np.true_divide(x, 255)
    x = np.expand_dims(x, axis=0)

    # Be careful how your trained model deals with the input
    # otherwise, it won't make correct prediction!
    x = preprocess_input(x, mode='caffe')

    preds = model.predict(x)
    return preds
```

Defining an index page for the webserver

```
[ ]  @app.route('/', methods=['GET'])
     def index():
         # Main page
         return render_template('index.html')
```

Figure 4.4: Class Diagram

CHAPTER 5

SYSTEM TESTING

In System Testing when we use Image Data Generator,using Image Data Generator we make training data and testing data for system testing we use only the testing images (Covid and Normal).

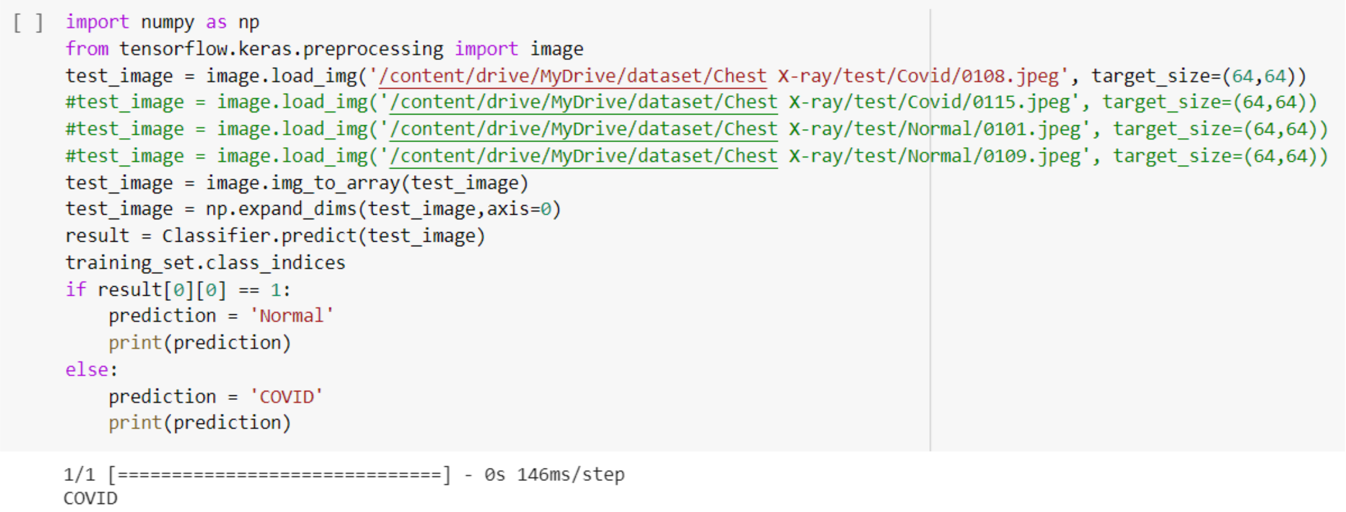
1)After making the training and testing data we use fit generator attribute for fitting the training and testing data .

2)Under that we define steps per epoch,epochs and target size of the testing set.(Note:Testing sets are used for validation sets)

3)We load an image from the dataset as test_image ,we convert it to numpy array and expand its dimensions.

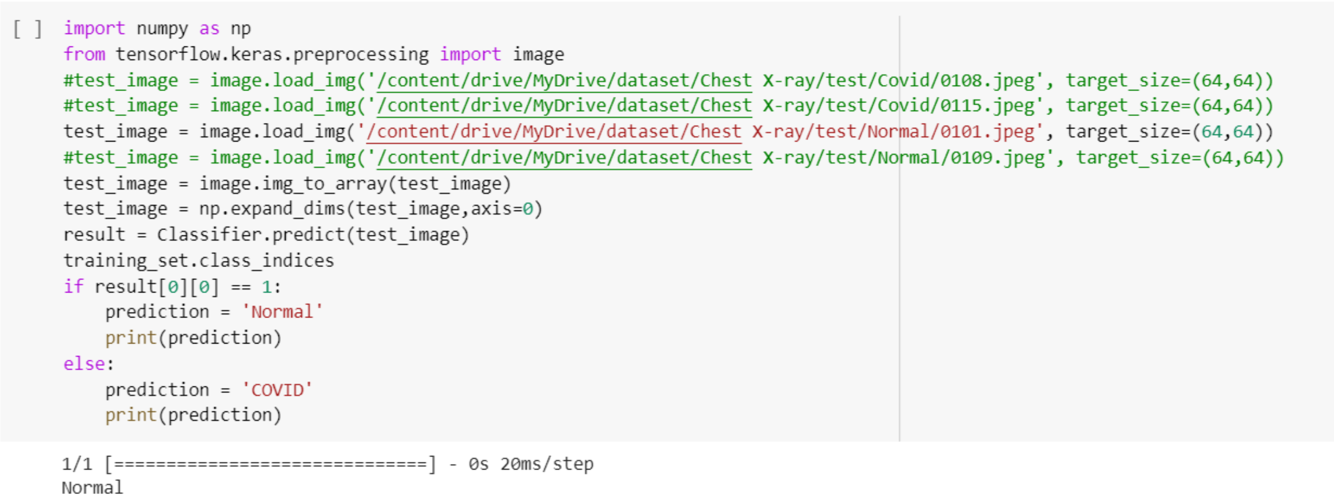
4)After that we predict the results according to the classes ,i.e, for Covid:1 and Normal:0

5)When we do this the backend looks like the following



```
[ ]  import numpy as np
     from tensorflow.keras.preprocessing import image
     test_image = image.load_img('/content/drive/MyDrive/dataset/Chest X-ray/test/Covid/0108.jpeg', target_size=(64,64))
     #test_image = image.load_img('/content/drive/MyDrive/dataset/Chest X-ray/test/Covid/0115.jpeg', target_size=(64,64))
     #test_image = image.load_img('/content/drive/MyDrive/dataset/Chest X-ray/test/Normal/0101.jpeg', target_size=(64,64))
     #test_image = image.load_img('/content/drive/MyDrive/dataset/Chest X-ray/test/Normal/0109.jpeg', target_size=(64,64))
     test_image = image.img_to_array(test_image)
     test_image = np.expand_dims(test_image,axis=0)
     result = Classifier.predict(test_image)
     training_set.class_indices
     if result[0][0] == 1:
         prediction = 'Normal'
         print(prediction)
     else:
         prediction = 'COVID'
         print(prediction)
```

```
1/1 [==============================] - 0s 146ms/step
COVID
```

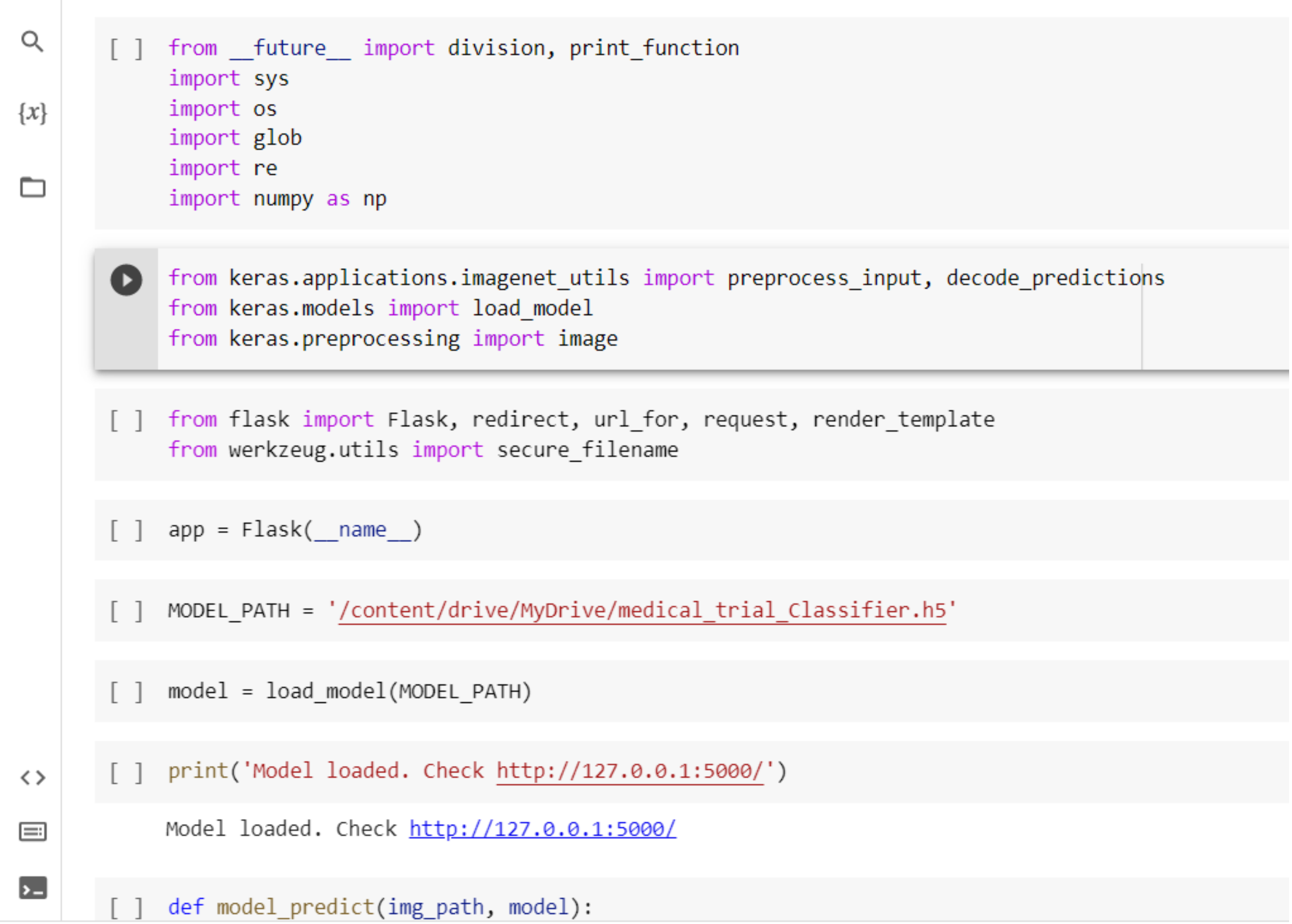


```
[ ] import numpy as np
    from tensorflow.keras.preprocessing import image
    #test_image = image.load_img('/content/drive/MyDrive/dataset/Chest X-ray/test/Covid/0108.jpeg', target_size=(64,64))
    #test_image = image.load_img('/content/drive/MyDrive/dataset/Chest X-ray/test/Covid/0115.jpeg', target_size=(64,64))
    test_image = image.load_img('/content/drive/MyDrive/dataset/Chest X-ray/test/Normal/0101.jpeg', target_size=(64,64))
    #test_image = image.load_img('/content/drive/MyDrive/dataset/Chest X-ray/test/Normal/0109.jpeg', target_size=(64,64))
    test_image = image.img_to_array(test_image)
    test_image = np.expand_dims(test_image,axis=0)
    result = Classifier.predict(test_image)
    training_set.class_indices
    if result[0][0] == 1:
        prediction = 'Normal'
        print(prediction)
    else:
        prediction = 'COVID'
        print(prediction)
```

```
1/1 [==============================] - 0s 20ms/step
Normal
```

6)So since we have classes for Covid and Normal as 1 &0 ,we have assumed that if it is 1 then it will be Covid case else it will be Normal.

7)When we deploy our project we import libraries,define app name and make two templates base.html and index.html.



```
[ ] from __future__ import division, print_function
    import sys
    import os
    import glob
    import re
    import numpy as np
```

```
from keras.applications.imagenet_utils import preprocess_input, decode_predictions
from keras.models import load_model
from keras.preprocessing import image
```

```
[ ] from flask import Flask, redirect, url_for, request, render_template
    from werkzeug.utils import secure_filename
```

```
[ ] app = Flask(__name__)
```

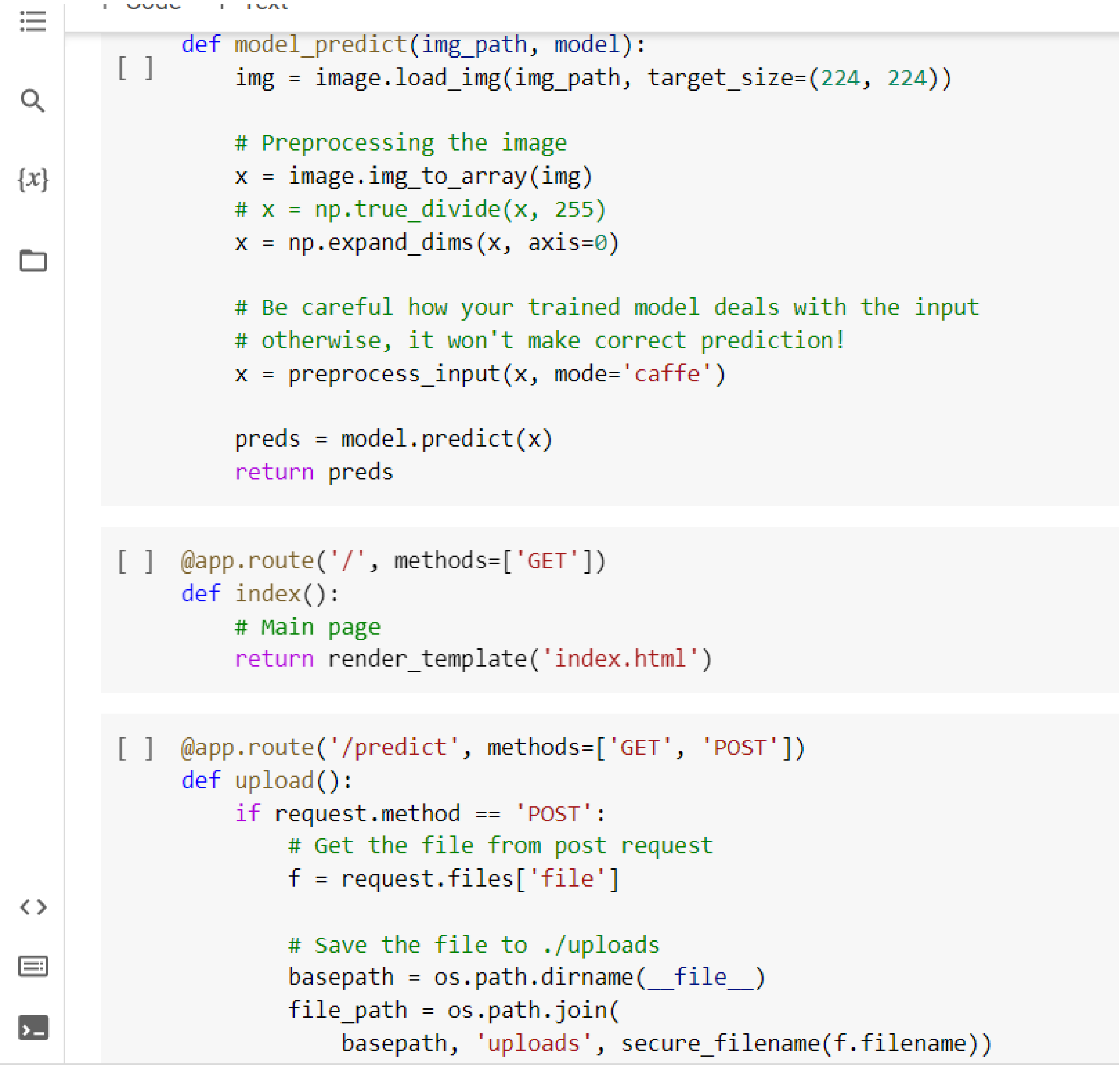
```
[ ] MODEL_PATH = '/content/drive/MyDrive/medical_trial_Classifier.h5'
```

```
[ ] model = load_model(MODEL_PATH)
```

```
[ ] print('Model loaded. Check http://127.0.0.1:5000/')
```

```
Model loaded. Check http://127.0.0.1:5000/
```

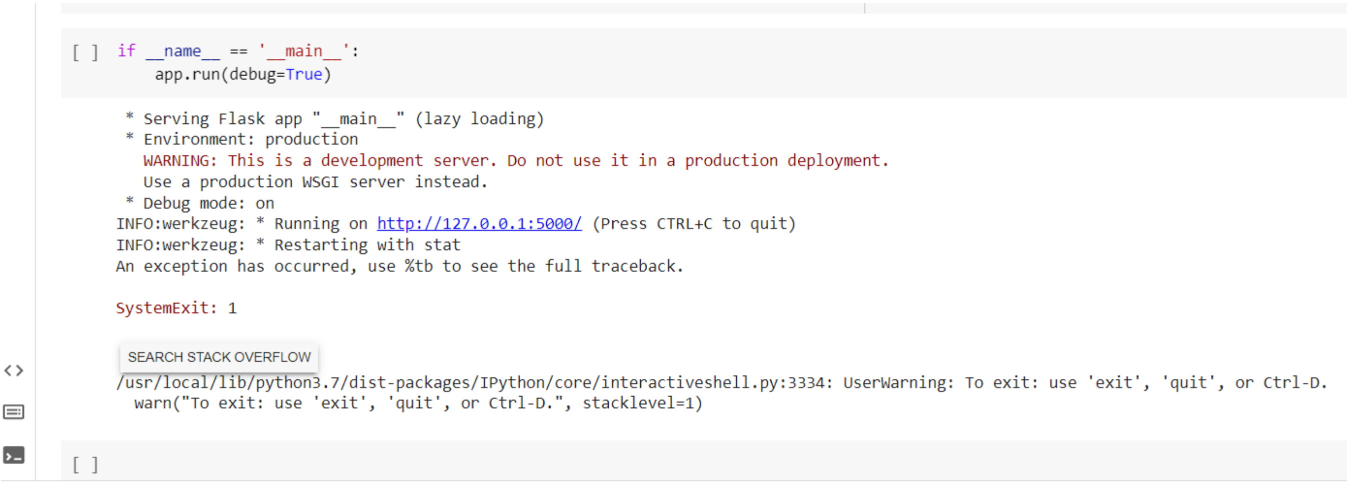
```
[ ] def model_predict(img_path, model):
```



```
def model_predict(img_path, model):
    img = image.load_img(img_path, target_size=(224, 224))

    # Preprocessing the image
    x = image.img_to_array(img)
    # x = np.true_divide(x, 255)
    x = np.expand_dims(x, axis=0)

    # Be careful how your trained model deals with the input
    # otherwise, it won't make correct prediction!
    x = preprocess_input(x, mode='caffe')

    preds = model.predict(x)
    return preds
```

```
@app.route('/', methods=['GET'])
def index():
    # Main page
    return render_template('index.html')
```

```
@app.route('/predict', methods=['GET', 'POST'])
def upload():
    if request.method == 'POST':
        # Get the file from post request
        f = request.files['file']

        # Save the file to ./uploads
        basepath = os.path.dirname(__file__)
        file_path = os.path.join(
            basepath, 'uploads', secure_filename(f.filename))
```



```
if __name__ == '__main__':
    app.run(debug=True)
```

```
 * Serving Flask app "__main__" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
INFO:werkzeug: * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
INFO:werkzeug: * Restarting with stat
An exception has occurred, use %tb to see the full traceback.

SystemExit: 1
```

```
/usr/local/lib/python3.7/dist-packages/IPython/core/interactiveshell.py:3334: UserWarning: To exit: use 'exit', 'quit', or Ctrl-D.
  warn("To exit: use 'exit', 'quit', or Ctrl-D.", stacklevel=1)
```

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

Owing to the high availability of medically-oriented image datasets, great success can be achieved using convolutional neural networks (CNNs) in the recognition and classification of these images.

Since previous research has shown CNNs to perform as well as the best clinicians in diagnostic tasks, they caused great excitement among researchers. The CNN algorithm can be improved in the future by adding a severity score to the COVID-19 positive cases, by analysing the images for other conditions like pneumonia, etc.

The COVID-19 pandemic has been felt across all industries, sparing little and forever changing the way we live, socialize, and work. Many of us have switched to remote work or telecommuting, while others have faced prolonged lay-offs and job losses. In one way or another, we’ve all felt the devastating blow of businesses being shuttered from economic instability.There may be such many research still going on in every country.

In Covid 19 pandemic our lives had become very difficult,i.e,the use of RT-PCR tests also couldn’t help because even when we use to have this test due to contact we couldn’t detect our exact result.The project aims at reducing the difficult efforts of doctors ,and also to help people for exact detection of Covid-19. To make use of Artificial Intelligence to solve real world problems and make them easier.

CHAPTER 7

REFERENCES

[1]https://courses.analyticsvidhya.com/courses/take/convolutional-neuralnetworks-cnn-from-scratch/texts/10846540-more-projects-with-cnn

[2] https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0265949

[3]https://www.analyticsvidhya.com/blog/2020/08/top-4-pre-trained-modelsfor-image-classification-with-python-code/

[4] https://link.springer.com/article/10.1007/s10489-020-01829-7

[5] https://link.springer.com/article/10.1007/s13369-022-06841-2

[6] https://drive.google.com/file/d/1s2C9DbUfOnadne9rEW60eHSRbTjYV9ab/view?usp=sharing