

A

Major Project Report on

Face-Mask Detection Using Python

Submitted in partial fulfillment

of the requirement of the degree of

BACHELOR OF TECHNOLOGY



SUBMITTED BY:

Akhilesh Joshi(18etccs005)

Yash Joshi(18etccs98)

GUIDED BY:

Aditya Maheshwari
(Assistant Professor)

SUBMITTED TO:

Head of department
of Science



TECHNO INDIA NJR
INSTITUTE OF TECHNOLOGY

Computer Science Engineering

TECHNO INDIA NJR INSTITUTE OF TECHNOLOGY ,UDAIPUR

TECHNO INDIA NJR INSTITUTE OF TECHNOLOGY, UDAIPUR



Department of Computer Science

Certificate

This is to certify that this project report “**Face Mask Detection using Python**” is the work of “**Akhilesh Joshi(18etccs005), Yash Joshi(18etccs098)**” who have carried out the project work under my supervision. I approve this project for submission of the Bachelor of Technology in the **Department of Computer Science and Engineering, Techno India NJR Institute of Technology**, affiliated to Rajasthan Technical University, Kota.

Mr. Aditya Maheshwari

Assistant Professor

ACKNOWLEDGEMENT

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many Faculties. We would like to extend my sincere thanks to all of them. We are highly indebted to **Mr. Aditya Maheshwari (Assistant Professor), TINJRIT** for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

We would like to express my gratitude towards my parents & members for their kind cooperation and encouragement which helped us in completion of this project. Last but not least. Many thanks go to the head of the project, Mr. Aditya Maheshwari who has invested his full effort in guiding the team in achieving the goal. We have to appreciate the guidance given by other supervisors as well as the panels especially in our project presentation that has improved our presentation skills thanks to their comment and advice.

We would like to express our special gratitude and thanks to all above mentioned people for giving us such attention and time. Our thanks and appreciation also go to our colleague in developing the project and people who have willingly helped us out with their abilities.

Place : Udaipur
Date:

Akhilesh Joshi(18etccs005)
Yash Joshi(18etccs098)

DECLARATION

We hereby declare that the project work **Face-Mask Detection Using Python** submitted, is a record of an original work done by us under the guidance of **Mr. Aditya Masheshwari** Assistant Professor, Department of Computer Science, **Techno India NJR, Udaipur** and this project work is submitted in the partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology. The results embodied in this report have not been submitted to any other University or Institute for the award of any degree or diploma.

Akhilesh Joshi
Yash Joshi

ABSTRACT

After the outbreak of the worldwide pandemic COVID-19, there arises a severe need for protection mechanisms, face masks being the primary one. According to the World Health Organization, the corona virus COVID-19 pandemic is causing a global health epidemic, and the most successful safety measure is wearing a face mask in public places. Convolutional Neural Networks (CNNs) have developed themselves as a dominant class of image recognition models. The aim of this project is to examine and test machine learning capabilities for detecting and recognizing face masks worn by people in any given video or picture or in real time. This project develops a real-time automatic Face detection and recognition system. It can be used as an entry management device by registering an organization's employees or students with their faces, and then recognizing individuals when they approach or leave the premises by recording their photographs with faces. Based on the performance and accuracy of our model, the result of the binary classifier will be indicated showing a green rectangle superimposed around the section of the face indicating that the person at the camera is wearing a mask, or a red rectangle indicating that the person on camera is not wearing a mask along with face identification of the person

TABLE OF CONTENTS

Contents no.	Page
CHAPTER 1: INTRODUCTION	8-9
1.1 Project Detail	8
1.2 Project Scope	8
1.3 Hardware and Software Used	9
CHAPTER 2: TECHNICAL FEASIBILITY	10-13
2.1 Technology Description	
2.1.1 Python	10
2.1.2 Tensor flow, Keras	11
2.1.3 Open CV	12
2.1.4 Use Cases	13
CHAPTER 3: SYSTEM ANALYSIS AND DESIGN	14-15
3.1 Analysis	14
3.2 Architecture Diagram	15
CHAPTER 4: TRAINING AND TESTING MODEL	16-19
4.1 Training Model	16
4.2 Testing Model	17
4.3 Training and Testing accuracy data	19
CHAPTER 5: FINAL TESTING	19-27
CHAPTER 6: FUTURE SCOPE	28
10.1 Future Scope	28
CHAPTER 7: CONCLUSION	29

List of Figures

Figure No.	Figure Name	Page No.
Figure 2.1	Modules	10
Figure 3.2.1	Architecture Diagram	15
Figure 4.1.1	Training Model	16
Figure 4.1.2	Training Loss and Accuracy	17
Figure 4.2.1	Testing Model	18
Figure 4.3.1	Training and accuracy data	19
Figure 6.1.1	Screenshot 1	26
Figure 6.1.2	Screenshot 2	26
Figure 6.1.3	Screenshot 3	27

CHAPTER 1:

INTRODUCTION

1.1 Project Detail

Prior to the coronavirus disease 2019 (covid-19) pandemic, there was no concrete evidence supporting the use of community masks to reduce the spread of respiratory infections. Masks are primarily intended to prevent the wearer from spreading the viral droplets (source control). Covid-19 and other respiratory infections spread primarily through inhalation of respiratory aerosols produced by coughing, sneezing, talking, or breathing. The virus propagates and migrates down the respiratory tract and may lead to pneumonia, acute respiratory distress syndrome (ARDS) and even death. Face mask detection is a subset of object recognition that uses image processing algorithms. Deep Learning models have been used in the majority of past research. After correctly recognizing the face in the picture or video, the CNN-based approach evaluates if the face has been disguised. It is also capable of identifying a moving face and a mask in a video as a surveillance job performance to identify face masks in public spaces. The face mask detector didn't use any morphed masked images dataset. The model is accurate, and since the MobileNetV2 architecture is used, it's also computationally efficient and thus making it easier to deploy the model to embedded systems (Raspberry Pi, Google Coral, etc.).

This system can therefore be used in real-time applications which require face-mask detection for safety purposes due to the outbreak of Covid-19. This project can be integrated with embedded systems for application in airports, railway stations, offices, schools, and public places to ensure that public safety guidelines are followed.

1.2 Scope

The scope of the project is to scan for people without masks so that they can be fined or impose a penalty for doing so. Further this can be used in a variety of situations to ensure the maximum precaution from different diseases. This project can be integrated with embedded systems for application in airports, railway stations, offices, schools and public places to ensure that public safety guidelines are followed.

1.3 Hardware/Software Used

The hardware requirements Face-Mask Detection are -

- Processor: Intel core I5 Processor
- Hard Disk: 80 GB HDD
- Ram: 512 MB and above
- 2MP camera and above

The software specifications are –

- Operating System: Window 7 and above
- Python, Keras, Tensorflow, Open
- MobileNetV2
- Open CV

Chapter : 2

Technology Description

2.1. Purpose

This system will be used in Four User Modules which are Keras, Tensorflow, MobileNet and OpenCV. As all of these have different requirements the modules are designed to meet their needs and avoid any type of confusion. The Uses of all Four User Modules have been described below.

```
1 tensorflow==2.9.1
2 keras==2.9.0
3 imutils==0.5.4
4 numpy==1.22.4
5 opencv-python==4.5.5.64
6 matplotlib==3.5.2
7 scipy==1.8.1
8 sklearn==0.0
9 scikit-learn==1.1.1|
```

fig 2.1

2.1.1 Python

- Python is a general purpose, dynamic, high-level, and interpreted programming language. It supports an Object Oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures.
- Python is *easy to learn* yet powerful and versatile scripting language, which makes it attractive for Application Development.
- Python's syntax and *dynamic typing* with its interpreted nature make it an ideal language for scripting and rapid application development.
- Python supports *multiple programming patterns*, including object-oriented, imperative, and functional or procedural programming styles.

- Python is not intended to work in a particular area, such as web programming. That is why it is known as a multipurpose programming language because it can be used with web, enterprise, 3D CAD, etc.
- We don't need to use data types to declare variables because it is *dynamically typed* so we can write `a=10` to assign an integer value in an integer variable.
- Python makes the development and debugging *fast* because there is no compilation step included in Python development, and the edit-test-debug cycle is very fast.

2.1.2 Tensorflow and Keras

- TensorFlow is an open-source software library used for dataflow programming beyond a range of tasks. It is a math library that is used for machine learning applications like neural networks.
- **TensorFlow** provides multiple APIs (Application Programming Interfaces). These can be classified into 2 major categories:
 - Low level API:
 - complete programming control
 - recommended for machine learning researchers
 - provides fine levels of control over the models
 - **TensorFlow Core** is the low level API of TensorFlow.
 - High level API:
 - built on top of **TensorFlow**
 - easier to learn and use than **TensorFlow**
 - make repetitive tasks easier and more consistent between different users.
- **Keras** is an open-source high-level Neural Network library, which is written in Python and is capable enough to run on Theano, TensorFlow, or CNTK. It was developed by one of the Google engineers, Francois Chollet. It is made user-friendly, extensible, and modular for facilitating faster experimentation with deep neural networks. It not only supports Convolutional Networks and Recurrent Networks individually but also their combination.

- **Keras** being a model-level library helps in developing deep learning models by offering high-level building blocks. All the low-level computations such as products of Tensor, convolutions, etc. are not handled by Keras itself, rather they depend on a specialized tensor manipulation library that is well optimized to serve as a backend engine. Keras has managed it so perfectly that instead of incorporating one single library of tensor and performing operations related to that particular library, it offers plugging of different backend engines into Keras.

2.1.3 OpenCV

- OpenCV is the huge open-source library for computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems.
- By using it, one can process images and videos to identify objects, faces, or even handwriting of a human. When integrated with various libraries, such as NumPy, python is capable of processing the OpenCV array structure for analysis.
- To Identify image pattern and its various features we use vector space and perform mathematical operations on these features.

2.1.4 Use Cases

A. Use cases of Tensorflow Module:

- Tensorflow is an interface for expressing machine learning algorithms.
- Tensorflow can utilize ML systems into fabrication.
- Tensorflow can be used in a bunch of areas of computer science such as text summarization, voice recognition, and information retrieval.
- The whole Sequential CNN architecture consists of several layers using Tensorflow in the backend.
- Tensorflow is used to reshape the data(image) in the image processing.

B. Use cases of Keras Module:

- Keras gives fundamental reflections and building units for creation and transportation of ML arrangements with high iteration velocity.
- It takes full advantage of the scalability and cross platform capabilities of TensorFlow.
- All the layers used in the CNN model are implemented using Keras.

C. Use case of OpenCV Module:

- OpenCV (Open Source Computer Vision Library), an open-source computer vision and ML software library.
- OpenCV is used to differentiate and recognize faces, recognize objects, group movements in recordings, trace progressive modules, follow eye gesture, track camera actions, expel red eyes from pictures taken utilizing flash, and find comparative pictures from an image database.

CHAPTER: 3

System Analysis and Design

3.1 Analysis

We use Convolutional Neural Network and Deep Learning for Real Time Detection and Recognition of Human Faces, which is simple face detection and recognition system is proposed in this paper which has the capability to recognize human faces in single as well as multiple face images in a database in real time with masks on or off the face. After pre-processing, face detection is performed by using CNNs architecture. Architecture layers of CNN are created using Keras Library in Python. Detected faces are augmented to make computation fast.

The proposed system contains the following modules:

A. Capture image

In this Module we are able to capture real time images. We do this by the help of Flutter and applying it to the Classifier Model.

B. Preprocessing Images

The input image is captured from a webcam or camera in the real-time world. The frames (images) from the dataset are loaded. Face images are cropped and resized after they have been loaded. Later, noise distortions in the images are suppressed.

C. Face Detection

In this method all black pixels in grayscale images were accumulated. They then deducted from the total number of white boxes. Finally, the outcome is compared to the given threshold, and if the criterion is met, the function considers it a hit.

D. Dataset

The proposed model has datasets captured from an individual's person. The dataset of faces is classified into with masks and without masks and is stored in different databases. Each folder consists of 40 to 60 images of an individual person respectively. The individual's face images should have images captured from different masks and different backgrounds so the accuracy of the training model increases. The dataset is integrated with Keras Library in Python. Larger the dataset, the more accurate the training model. So dataset images are directly congruent to accuracy of the training model.

D. Mask Detection

For Mask Detection, we use a sequential CNN model along with the inbuilt Keras Library in Python. The sequential CNN model is trained from a dataset of human faces with or without masks on the faces. It forms a logic from the pre-processed images like a human brain, then the model detects

the face along with the mask using feature extraction and feature selection. After identification of the mask along with the face of the person, it forwards to the prediction or identification stage.

E. Prediction(image)

In this Module prediction of a person takes place. Here the browsed image will be placed into the model and output will be shown as based on which label its get matched the most.

3.2 Face mask detection flow from the webcam

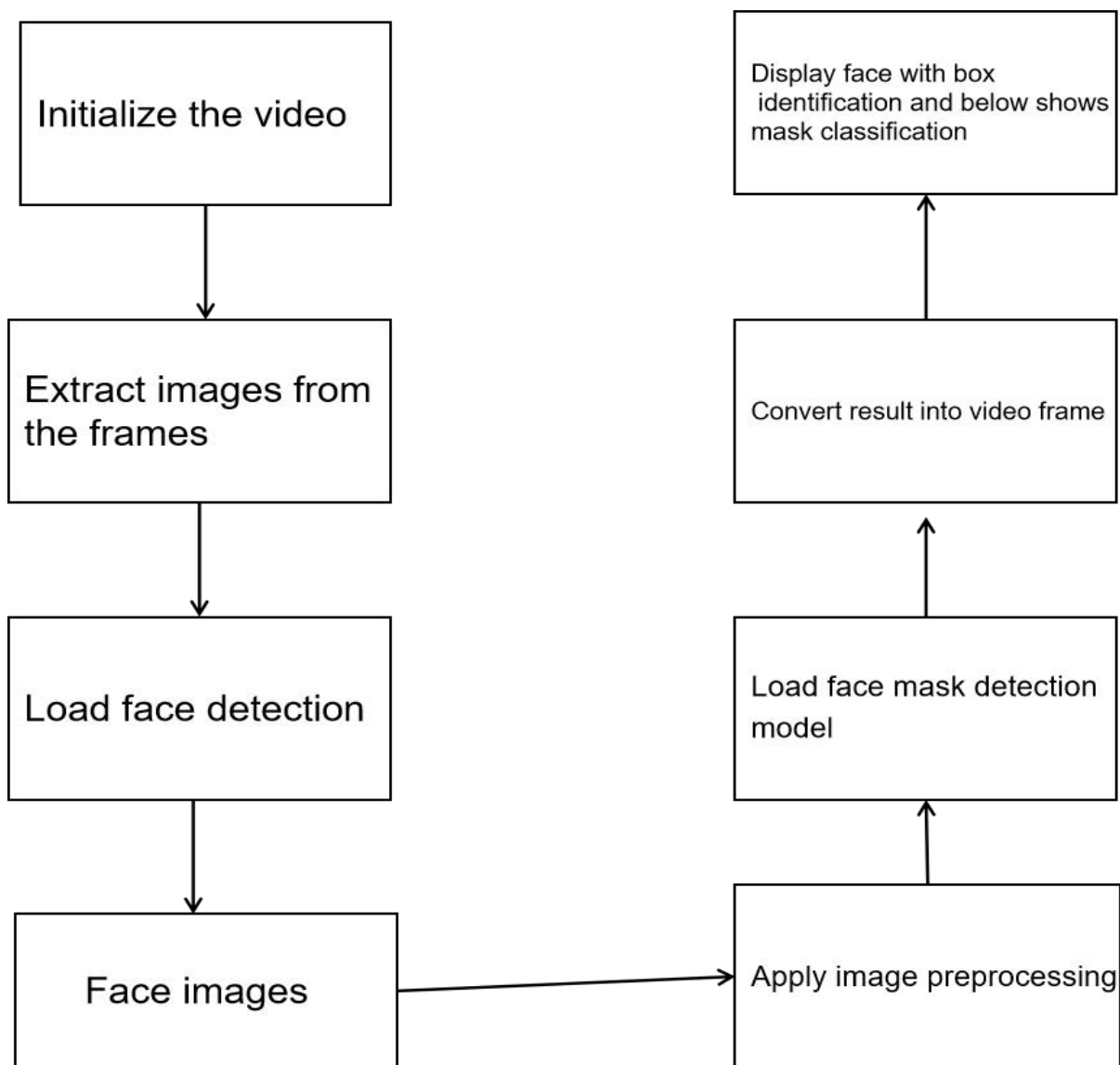


Fig 3.2.1
Architecture Diagram

CHAPTER: 4

Training and Testing Model

4.1 Training Model

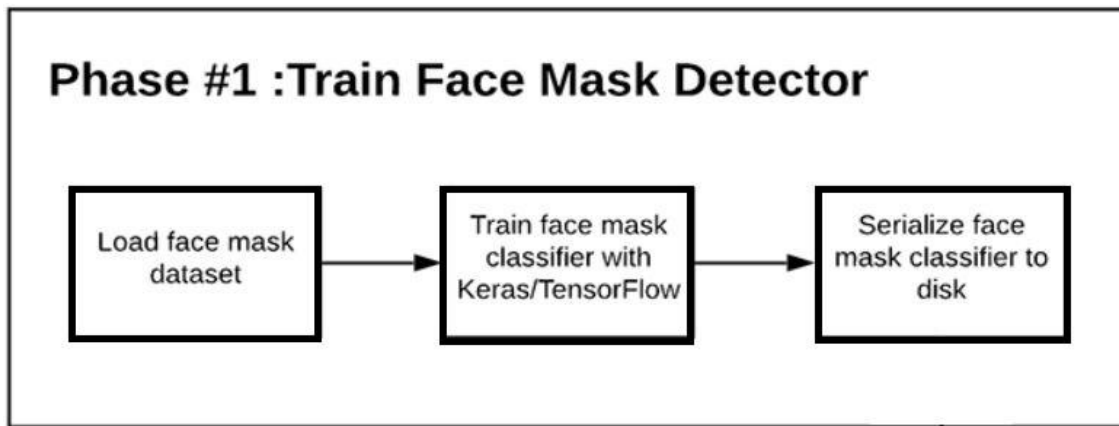


Fig 4.1.1

IN THIS PHASE:

- The pre-processed face images are directed to the CNN model for training.
- Based on the dataset given, a logic is formed in the CNN to categorize the faces according to their features.
- The trained model is capable of categorizing human faces based on with or without masks on it.
- Training model is done with the help of a sequential CNN model.


```

2022-05-28 14:27:52.644740: W tensorflow/core/framework/cpu_allocator_impl.cc:82] Allocation of 156905472 exceeds 10% of free system memory.
95/95 [=====] - 155s 2s/step - loss: 0.4022 - accuracy: 0.8556 - val_loss: 0.1352 - val_accuracy: 0.9909
Epoch 2/20
95/95 [=====] - 136s 1s/step - loss: 0.1500 - accuracy: 0.9651 - val_loss: 0.0671 - val_accuracy: 0.9935
Epoch 3/20
95/95 [=====] - 138s 1s/step - loss: 0.0969 - accuracy: 0.9753 - val_loss: 0.0485 - val_accuracy: 0.9922
Epoch 4/20
95/95 [=====] - 138s 1s/step - loss: 0.0789 - accuracy: 0.9782 - val_loss: 0.0424 - val_accuracy: 0.9922
Epoch 5/20
95/95 [=====] - 135s 1s/step - loss: 0.0737 - accuracy: 0.9815 - val_loss: 0.0368 - val_accuracy: 0.9909
Epoch 6/20
95/95 [=====] - 110s 1s/step - loss: 0.0532 - accuracy: 0.9868 - val_loss: 0.0334 - val_accuracy: 0.9935
Epoch 7/20
95/95 [=====] - 90s 946ms/step - loss: 0.0520 - accuracy: 0.9865 - val_loss: 0.0380 - val_accuracy: 0.9870
Epoch 8/20
95/95 [=====] - 89s 934ms/step - loss: 0.0527 - accuracy: 0.9858 - val_loss: 0.0298 - val_accuracy: 0.9922
Epoch 9/20
95/95 [=====] - 87s 917ms/step - loss: 0.0430 - accuracy: 0.9868 - val_loss: 0.0296 - val_accuracy: 0.9935
Epoch 10/20
95/95 [=====] - 88s 928ms/step - loss: 0.0382 - accuracy: 0.9891 - val_loss: 0.0290 - val_accuracy: 0.9935
Epoch 11/20
95/95 [=====] - 88s 927ms/step - loss: 0.0353 - accuracy: 0.9888 - val_loss: 0.0294 - val_accuracy: 0.9922
Epoch 12/20
95/95 [=====] - 87s 920ms/step - loss: 0.0241 - accuracy: 0.9927 - val_loss: 0.0235 - val_accuracy: 0.9935
[INFO] evaluating network...
24/24 [=====] - 16s 633ms/step

```

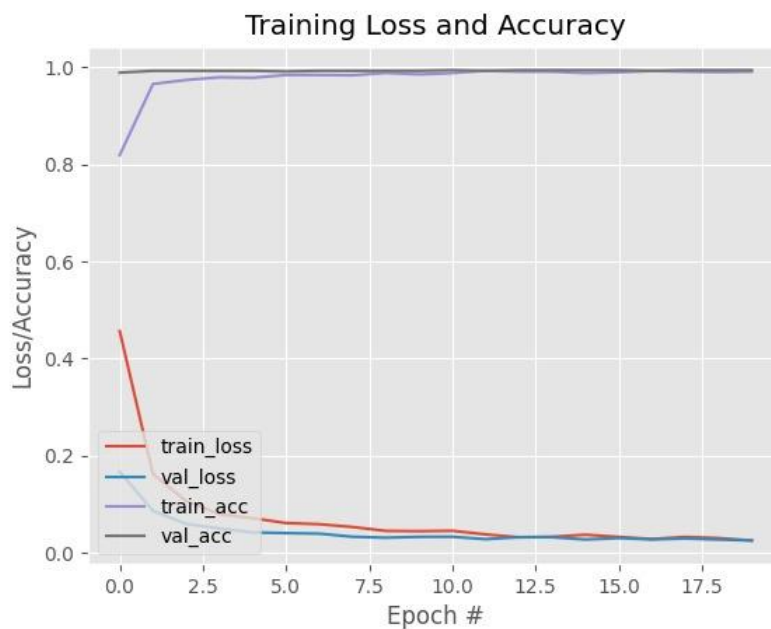


fig 4.1.2

- In fig 4.2 COVID-19 face mask detector training accuracy/loss curves demonstrate high accuracy and little signs of overfitting on the data. We're now ready to apply our knowledge of computer vision and deep learning using Python, OpenCV, and TensorFlow/Keras to perform face mask detection.

4.2 Testing Model

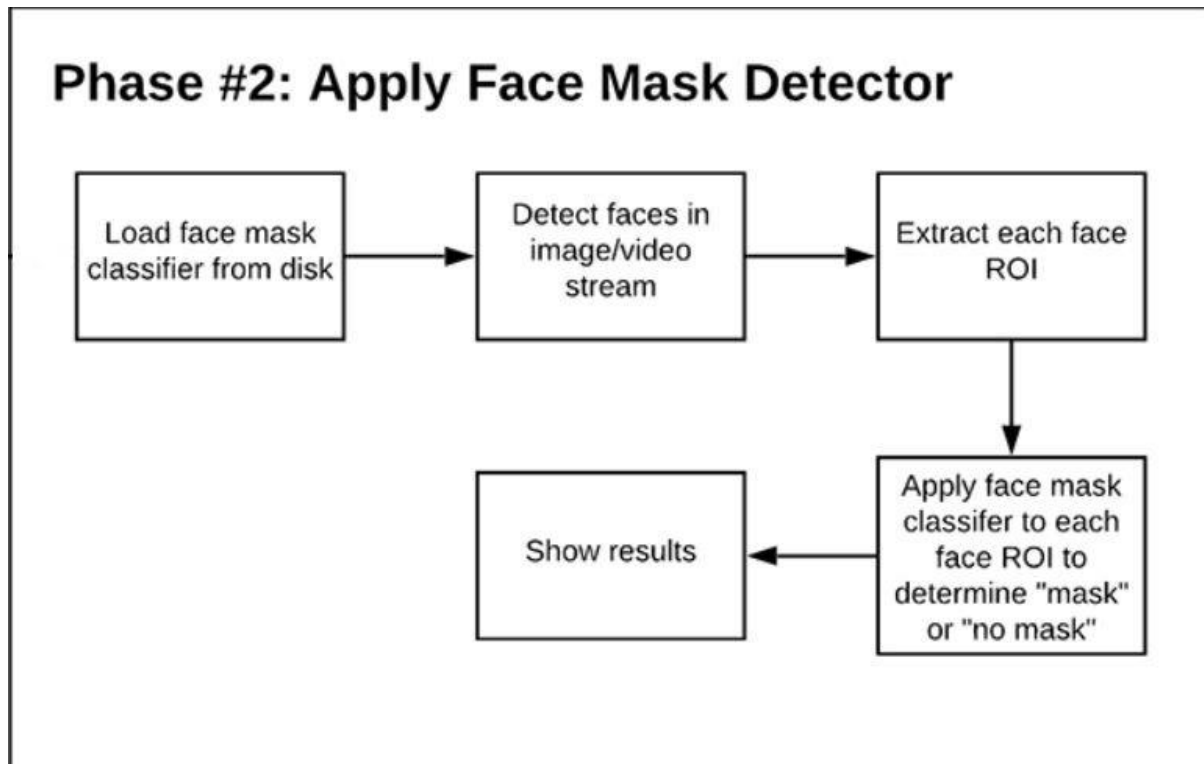


Fig 4.2.1

IN THIS PHASE:

- In this phase, when a person comes in front of a webcam, the image is captured and predicted by the CNN model according to the logic learned by the sequential model.
- The image undergoes pre-processing.
- These pre-processed images and the saved CNN model are then loaded. Based on the algorithm interpreted by the system, it predicts and detects human faces according to a trained model.
- In fig 4.3, INIT_LR, EPOCHS & BS are the three factors through which by adjusting we can achieve maximum accuracy.

```
INIT_LR = 1e-4  
EPOCHS = 20  
BS = 32
```

fig 4.3

4.3 Training and accuracy data

	precision	recall	f1-score	support
with_mask	0.99	0.99	0.99	383
without_mask	0.99	0.99	0.99	384
accuracy			0.99	767
macro avg	0.99	0.99	0.99	767
weighted avg	0.99	0.99	0.99	767

fig 4.3.1

CHAPTER 5:

Final Testing

5.1 Testing Methodology

Companies rely on software more than ever to provide and manage information with strategic and operational importance and to provide key decision support. Rising customer expectations for fault-free, requirements-exact software have increased awareness of the importance of software testing as a critical activity.

We begin the testing process by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used. The process verifies that the application meets the requirements specified in the system requirements document and is bug free. At the end of each testing day, we

prepare a summary of completed and failed tests. Applications are not allowed to launch until all identified problems are fixed. A report is prepared at the end of testing to show exactly what was tested and to list the final outcomes.

Our software testing methodology is applied in three distinct phases: unit testing, system testing, and acceptance were testing.

Unit Testing: The programmers conduct unit testing during the development phase. Programmers can test their specific functionality individually or with other units. However, unit testing is designed to test small pieces of functionality rather than the system as a whole. This allows the programmers to conduct the first round of testing to eliminate bugs before they reach the testing staff. In unit testing the analyst tests the programs making up a system.

For this reason, unit testing is sometimes called program testing. Unit testing gives stress on the modules independently of one another, to find errors. This helps the tester in detecting errors in coding and logic that are contained within that module alone. The errors resulting from the interaction between modules are initially avoided.

For example, a hotel information system consists of modules to handle reservations; guest checking and checkout; restaurant, room service and miscellaneous charges; convention activities; and accounts receivable billing. For each, it provides the ability to enter, modify or retrieve data and respond to different types of inquiries or print reports. The test cases needed for unit testing should exercise each condition and option.

Unit testing can be performed from the bottom up, starting with smallest and lowest-level modules and proceeding one at a time. For each module in bottom-up testing a short program is used to execute the module and provides the needed data, so that the module is asked to perform the way it will when embedded within the larger system.

System Testing: The objective of system testing is to ensure that all individual programs are working as expected, that the programs link together to meet the requirements specified and to ensure that the computer system and the associated clerical and other procedures work together.

The initial phase of system testing is the responsibility of the analyst who determines what conditions are to be tested, generates test data, produces a schedule of expected results, runs

the tests and compares the computer produced results with the expected results with the expected results.

The analyst may also be involved in procedures testing. When the analyst is satisfied that the system is working properly, he hands it over to the users for testing. The importance of system testing by the user must be stressed. Ultimately it is the user must verify the system and give the go-ahead.

During testing, the system is used experimentally to ensure that the software does not fail, i.e., that it will run according to its specifications and in the way users expect it to. Special test data is input for processing (test plan) and the results are examined to locate unexpected results.

A limited number of users may also be allowed to use the system so analysts can see whether they try to use it in unexpected ways. It is preferably to find these surprises before the organization implements the system and depends on it. In many organizations, testing is performed by persons other than those who write the original programs. Using persons who do not know how certain parts were designed or programmed ensures more complete and unbiased testing and more reliable software.

The system is tested as a complete, integrated system. System testing first occurs in the development environment but eventually is conducted in the production environment. Functionality and performance testing are designed to catch bugs in the system, unexpected results, or other ways in which the system does not meet the stated requirements.

The testers create detailed scenarios to test the strength and limits of the system, trying to break it if possible. Editorial reviews not only correct typographical and grammatical errors, but also improve the system's overall usability by ensuring that on-screen language is clear and helpful to users. Accessibility reviews ensure that the system is accessible to users with disabilities.

System testing consists of the following five steps:

- i. Program testing
- ii. String testing

- iii. System testing
- iv. System documentation
- v. User acceptance testing

Program Testing

A program represents the logical elements of a system. For a program to run satisfactorily, it must compile and test data correctly and tie in properly with other programs. It is the responsibility of a programmer to have an error free program. At

The time of testing the system, there exist two types of errors that should be checked. These errors are syntax and logic.

A syntax error is a program statement that violates one or more rules of the language in which it is written. An improperly defined field dimension or omitted key words are common syntax errors. These errors are shown through error messages generated by the computer. A logic error, on the other hand, deals with incorrect data fields out of range items, and invalid combinations.

Since the logical errors are not detected by the compiler, the programmer must examine the output carefully to detect them. When a program is tested, the actual output is compared with the expected output. When there is a discrepancy, the sequence of the instructions must be traced to determine the problem. The process is facilitated by breaking the program down into self- contained portions, each of which can be checked at certain key points.

String Testing

Programs are invariably related to one another and interact in a total system. Each program is tested to see whether it conforms to related programs in the system. Each part of the system is tested against the entire module with both test and live data before the whole system is ready to be tested.

System Testing

System testing is designed to uncover weaknesses that were not found in earlier tests. This includes forced system failure and validation of total system as it will be implemented by its user in the operational environment. Under this testing, generally we

Take low volumes of transactions based on live data. This volume is increased until the maximum level for each transaction type is reached.

The total system is also tested for recovery and fallback after various major failures to ensure that no data are lost during the emergency.

All this is done with the old system still in operation. When we see that the proposed system is successful in the test, the old system is discontinued.

System Documentation

All design and test documentation should be well prepared and kept in the library for future reference. The library is the central location for maintenance of the new system.

User Acceptance Testing

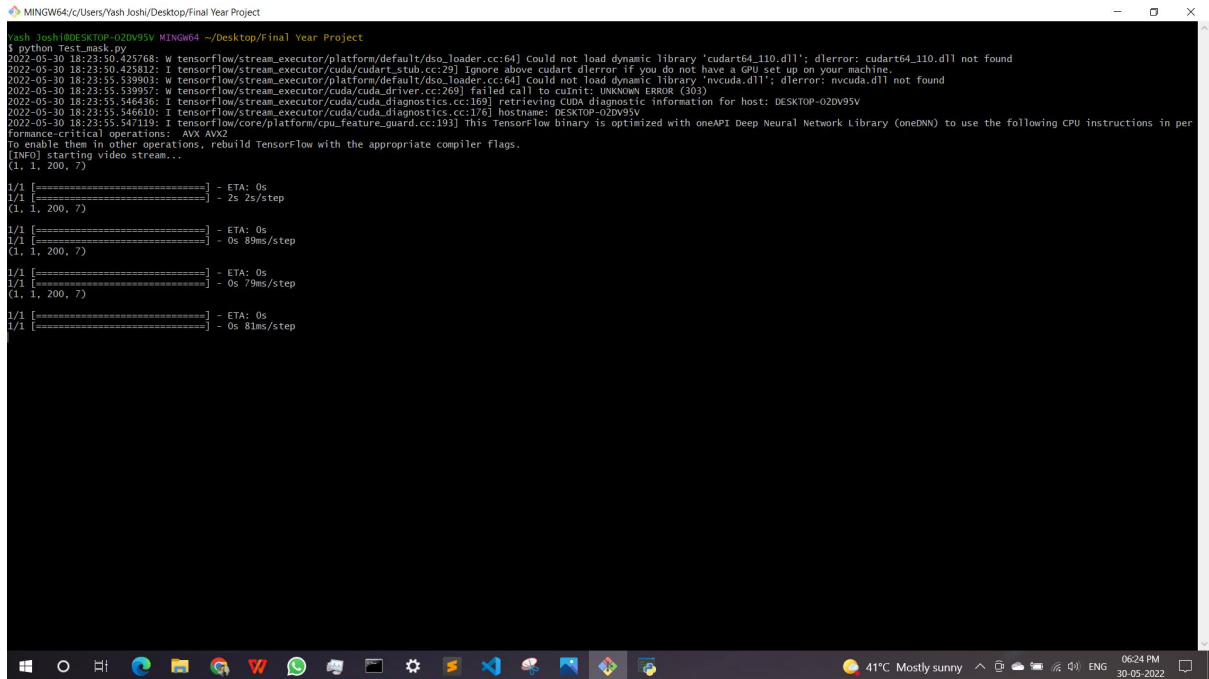
An acceptance test has the objective of selling the user on the validity and reliability of the system. It verifies that the system's procedures operate to system specifications and that the integrity of important data is maintained. Performance of an acceptance test is actually the user's show. User motivation is very important for the successful performance of the system. After that a comprehensive test report is prepared. This report shows the system's tolerance, performance range, error rate and accuracy.

Table 6.1 Test Report with test data

TEST REPORT WITH TEST DATA		
(To be filled by System Analyst/Programmer)		
Project Name : Face Mask Detection Using Python		
S No.	Testing Parameter	Observations
A.	INTERFACE TESTING 1) User-friendliness 2) Consistent menus	OK NA
B.	CONTROL FLOW TESTING 1) IF-THEN-ELSE 2) DO WHILE 3) CASE-SWITCH	OK OK OK
C.	VALIDATION TESTING 1) Check for improper or inconsistent typing 2) Check for erroneous initialization or default values 3) Check for incorrect variable names 4) Check for inconsistent Data Types 5) Check for relational/arithmetic operators	OK OK OK OK OK
D.	DATA INTEGRITY/SECURITY TESTING 1) Data Insertion/ Deletion/ Updating 2) Boundary condition (Underflow, Overflow Exception) 3) Check for unauthorized access of data 4) Check for data availability	OK OK OK OK

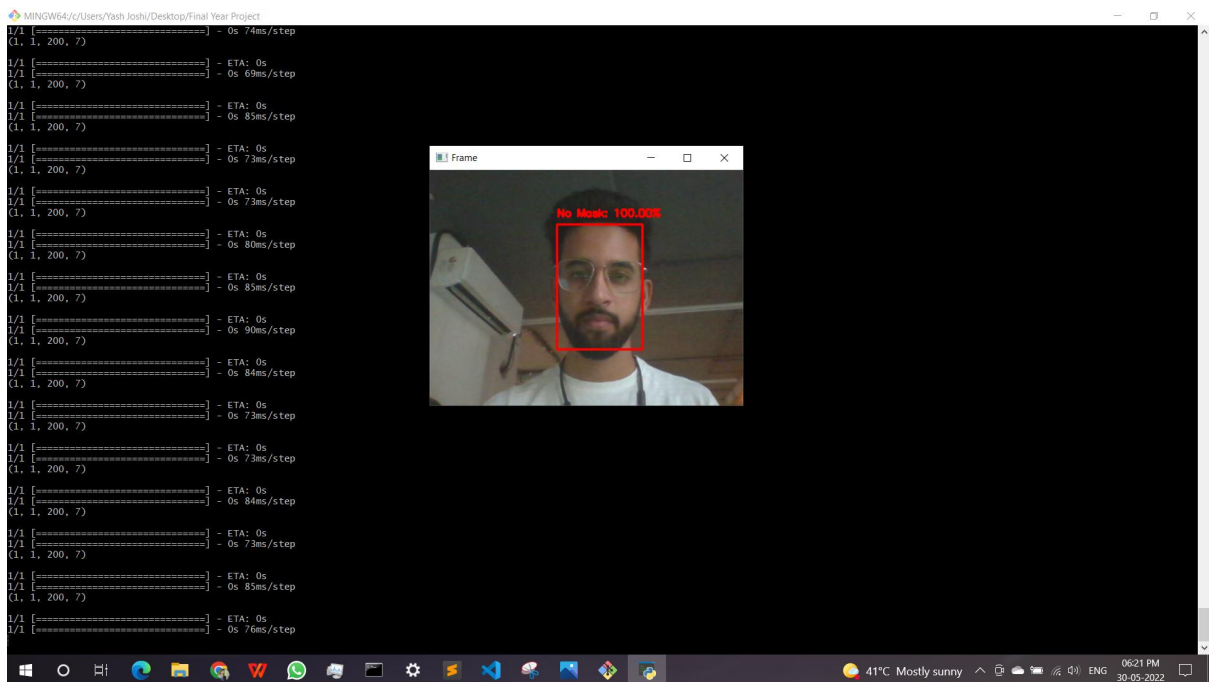
E.	<p>EFFICIENCY TESTING</p> <ol style="list-style-type: none"> 1) Throughput of the system 2) Response time of the system 3) Online disk storage required by the system 4) Primary memory required by the system 	<p>OK</p> <p>OK</p> <p>OK</p> <p>OK</p>
F.	<p>ERROR HANDLING ROUTINES</p> <ol style="list-style-type: none"> 1) Error description are intelligent/ understandable 2) Error recovery is smooth 3) All error handling routines are tested and executed at least once 	<p>OK</p> <p>OK</p> <p>OK</p>

Chapter – 6 : Screenshots



```
MINGW64~/c:/Users/Yash Joshi/Desktop/Final Year Project
yash joshi@DESKTOP-02DV95V MINGW64 ~/Desktop/Final Year Project
$ python Test_mask.py
2022-05-30 18:23:50.425768: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cudart64_110.dll'; dlderror: cudart64_110.dll not found
2022-05-30 18:23:50.425812: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlderror if you do not have a GPU set up on your machine.
2022-05-30 18:23:55.539903: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'nvcuda.dll'; dlderror: nvcuda.dll not found
2022-05-30 18:23:55.539957: W tensorflow/stream_executor/cuda/cuda_driver.cc:269] failed call to cuInit: UNKNOWN ERROR (303)
2022-05-30 18:23:55.546636: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: DESKTOP-02DV95V
2022-05-30 18:23:55.546610: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: DESKTOP-02DV95V
2022-05-30 18:23:55.547119: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in per
formance-critical operations: AVX AVX2
to enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
[INFO] starting video stream...
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 2s 2s/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 89ms/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 79ms/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 81ms/step
(I, 1, 200, 7)
```

fig 6.1.1



```

I/A [=====] - ETA: 0s
I/A [=====] - 0s 74ms/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 69ms/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 85ms/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 73ms/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 73ms/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 80ms/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 85ms/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 90ms/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 84ms/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 73ms/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 73ms/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 84ms/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 73ms/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 85ms/step
(I, 1, 200, 7)
I/A [=====] - ETA: 0s
I/A [=====] - 0s 76ms/step
(I, 1, 200, 7)

Frame
No Mask: 100.00%
```

fig 6.1.2

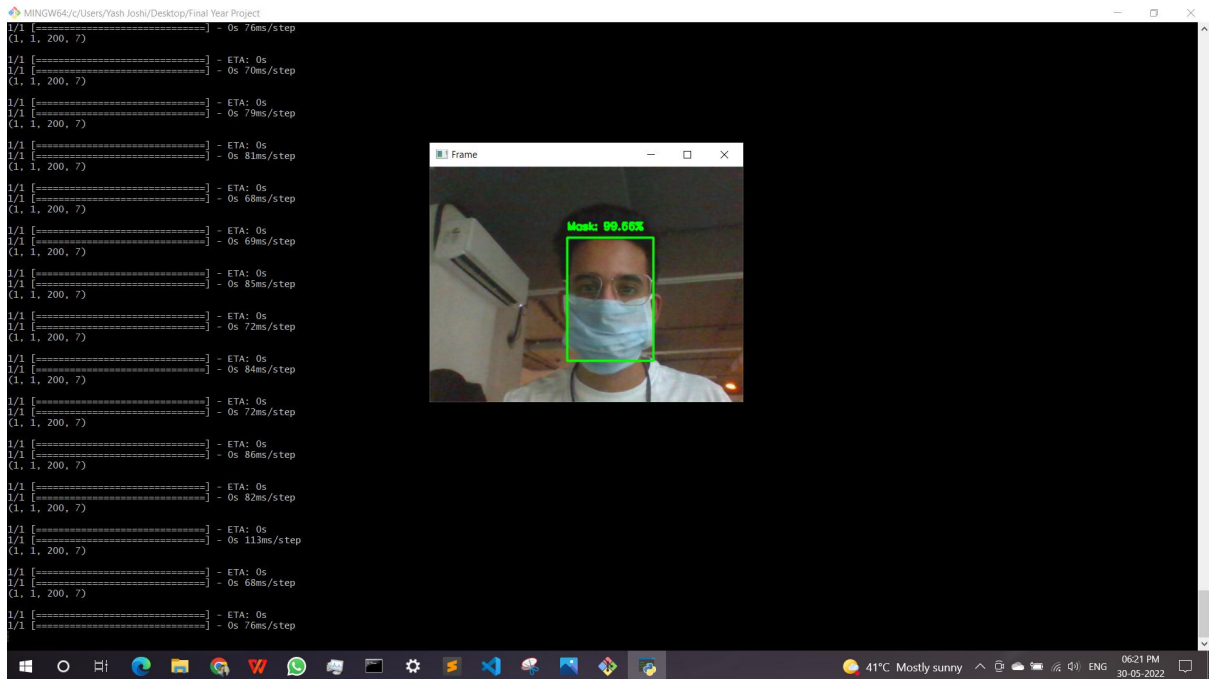


fig 6.1.3

- In fig 6.1.1 python Test_Mask.py command was entered and it started warming up the camera.
- A window will pop up showing real time mask availability and accuracy percentage.
- In fig 6.1.2 a face is inside a red box it means the person has not worn the mask.
- In fig 6.1.3 a face is inside a green box it means the person has worn the mask.

CHAPTER 6:

FUTURE SCOPE AND CONCLUSION

6.1 Future Scope

- Firstly, the proposed technique can be integrated into any high-resolution video surveillance devices and not limited to mask detection only.
- Secondly, the model can be extended to detect facial landmarks with a facemask for biometric purposes.
- Our proposed system can detect and recognize human faces in the real-time world.
- Compared to the traditional face detection and recognition system, the face detection and recognition based on CNN model along with the use of Python libraries has shorter detection and recognition time and stronger robustness, which can reduce the miss rate and error rate.
- It can still guarantee a high test rate in a sophisticated atmosphere, and the speed of detection can meet the real time requirement, and achieve good effect.
- The proposed CNN model shows greater accuracy and prediction for detecting and recognising human faces.

6.2 Conclusion

The proposed system to classify face mask detection using COVID-19 precaution both in images and videos using convolution neural network. Extensive experimentation on the datasets and the performance evaluation of the proposed methods are exhibited. Further, we made a successful attempt to preserve inter and intra class variations of face mask detection. We studied the different classifiers like OpenCV and Tensorflow. The project is developed as a prototype to detect masks for the people. The work is designed to provide a safety system for the people in order to avoid COVID-19. We proposed continuous monitoring of the people's conditions and storing the people's data in the server using the Deep learning concept. In order to investigate the performance, the proposed method and extensive experimentation is conducted on various Image datasets. We conducted experimentation under varying number of training and testing percentage for 10 random trials. The proposed system works well effectively for grayscale as well as for the colour image with masks on it or without masks on it.

CHAPTER - 7

REFERENCES

[1] International Journal of Engineering Research & Technology (IJERT) ISSN: 2278-0181 Published by, www.ijert.org ICCIDT - 2021 Conference Proceedings

[2] The Face Mask Detection Technology for Image Analysis in the Covid-19 Surveillance System To cite this article: G K Jakir Hussain et al 2021 J. Phys.: Conf. Ser. 1916 012084

[3] Face mask detection using deep learning: An approach to reduce risk of Coronavirus spread: [Shilpa Sethi](#), [Mamta Kathuria](#), [Trilok Kaushik](#)

[4] Facial Mask Detection Using Depthwise Separable Convolutional Neural Network Model During COVID-19 Pandemic: [Muhammad Zubair Asghar](#), [Fahad R. Albogamy](#), [Mabrook S. Al-Rakhami](#), [Junaid Asghar](#), Mohd Khairil Rahmat, Muhammad Mansoor Alam, Adidah Lajis and Haidawati Mohamad Nasir