Real-Time Face Mask Detection Using Machine Learning

BY

MAHEDI HASSAN HASIB

ID: 173-15-10328

JASIA AFRIN

ID: 181-15-10779

AND

MD. ASHIKUR RAHMAN

ID: 173-15-10353

This Report Presented in Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Computer Science and Engineering.

Supervised By

Mr. Ahmed Al Marouf
Senior Lecturer
Department of CSE
Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

January 2022

APPROVAL

This Project/internship titled **“Real-Time Face Mask Detection Using Machine Learning”**, submitted by Mahedi Hassan Hasib&Jasia Afrin&Ashikur Rahman,ID No: 173-15-10328,181-15-10779,173-15-10353 to the Department of Computer Science and Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on 04 January 2022.

BOARD OF EXAMINERS

Chairman

Dr. S.M Aminul Haque (SMAH)
Associate Professor and Associate Head
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Internal Examiner

Raja Tariqul Hasan Tusher (THT)
Senior Lecturer
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Internal Examiner

Md. Sazzadur Ahamed (SZ)
Senior Lecturer
Department of Computer Science and Engineering
Faculty of Science & Information Technology

External Examiner

Dr. Shamim H Ripon
Professor
Department of Computer Science and Engineering
East-West University

DECLARATION

We hereby declare that this project has been done by us under the guidance supervision of Dr. Fizar Ahmed, Assistant Professor, Department of CSE Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere for the award of any degree or diploma.

Supervised by:

Mr. Ahmed Al Marouf

Senior Lecturer Department of CSE

Daffodil International University

Co-Supervised by:

T Siddique

Shah Md. Tanvir Siddiquee

Assistant Professor

Computer Science & Engineering

Daffodil International University

ACKNOWLEDGEMENT

First and initially, we offer our greatest appreciation and gratitude to almighty ALLAH for His heavenly gift, which enabled us to complete the previous year's endeavor
We are highly grateful and wish our profound indebtedness to “**Mr. Ahmed Al Marouf**”, **Senior Lecturer,** Department of CSE Daffodil International University, Dhaka, Bangladesh. Profound

We completed this project due to the knowledge and dynamic ideas he provided. His unwavering perseverance, intellectual instruction, consistent encouragement, steady and passionate supervision, constructive judgment, critical direction, and reading and correcting several poor drafts

All stages have made it possible to complete this project.

We would like to express our heartiest gratitude to **Dr. Tauhid Bhuiyan, Professor and Head,** Department of **Computer Science and Engineering**, Daffodil International University, for his kind assistance to complete our project and also to other faculty members and the staff of **Computer Science and Engineering** department of Daffodil International University.

We would like to thank all of our Daffodil International University research mates
for participating in this discussion while completing their course work.

ABSTRACT

“Real-Time Face Mask Detection Using Machine Learning”, is a Machine Learning-based project in real-time. Covid 19 has a huge impact on our modern daily life. We have been suffering from the Covid pandemic since late 2019. Our regular life became much tougher than ever.

We have a Wearing mask, one of the major rules we have to maintain in this pandemic situation. We have created a machine learning project that will detect immediately if a person wearing a mask or not. The project can be a crying need for this pandemic situation. The project has been built with python with the help of library functions like TensorFlow, OpenCV, Keras, and many more.

This system recognition works very fast to detect masks with the value of percentages in the OpenCV camera view. After running the project, the webcam will automatically open itself and start detecting a face with a mask or not a mask in it.

This project can be used in any government office, bank, hospital, educational institution, Closed Circuit Camera, and corporate offices as well. This system is very fast to recognize masks in a fast way of time.

TABLE OF CONTENTS

LIST OF FIGURES:

LIST OF TABLES

TABLE **PAGE-NO**

CHAPTER 1

INTRODUCTION

1.1 Introduction

Our project is an ML project based on Python. The organization is a face mask detection system based on an object recognition system. The system is an automatic face mask detection system with real-time face mask detection technology. The system automatically launches the webcam and starts detecting the mask in the user. The system is very easy to use. Our project is built with on machine learning. The project has had a huge impact on machine learning python-based library functions.

1.2 Motivation

The project is based on python and implemented on machine learning. The project is built on OpenCV, Keras, TensorFlow libraries. After running the project webcam will be open up automatically and start detecting the user mask. The system will detect if the user is wearing a mask or not.

As we had hoped, we were able to clean the system by utilizing the OpenCV. Open Computer Vision was designed for approaches proficiency and a strong emphasis on real-time face-to-face systems.

So, it is a face mask detection using a web camera. After opening the webcam camera, it will start looking for the mask in the user's face.

that the task of recognizing training images

1.3 Objective

The goal of ML is to train a model using previously recorded, labeled data to estimate the value of some amount based on a new data item with unknown target categorization.

The projects' goals are given as follows:

□ Fully automatic system.
□ Real-Time mask detection.
□ User Friendly.
□ Can detect mask from anywhere.

1.4 Description

The following project is ML-based python development. Mask detection organization that is based on OpenCV.

A modern solution that can work with face mask detection. also, in trained images as well. Our system is built by using Python OpenCV, TensorFlow, Keras. The project is very user-friendly and easy to use for anyone.

1.5 Expected Outcome

Machine learning is at the heart of our project. It's just training our computer, and we'll obtain the desired results. After launching the system, it will take the real-time faces and detect masks from the user.

Our initiatives' expected outcomes are as given below:

- We can automatically take photos in real-time.
- The system has the ability of face mask detection.
- This webcam starts automatically.
- The python program is easy to use.
- The system's accuracy contains 100%.

1.6 Project Management and Finance

1.7 Report Layout

Introduction to Chapter 1

spoke around project motivation, objectives, and projected outcomes in this part.

Background Research (Chapter 2)

We spoke about the literature review of similar projects, and the scope of the project in this part. Our project's issue has given that.

Requirement Specification (Chapter 3)

In this part, we outline our project needs. That is python machine learning libraries like OpenCV, TensorFlow,Keras.

Design Specification (Chapter 4)

This section provides an overview of the project's whole programming design.

Implementation and Testing (Chapter 5)

This section covers the project's implementation and testing results.

Final Thoughts and Plans for the Future (Chapter 6)

This section provides a summary of the project's conclusion and future work.

CHAPTER 2

BACKGROUND STUDY

2.1 Terminologies

The system is an automatic face mask detection system with real-time face mask detection technology. The system automatically launches the webcam and starts detecting the mask in the user. The system is very easy to use. The project has had a huge impact on machine learning python-based library functions.

2.2 Literature review

Nagrath et. al. have described in their paper picture dataset, which was separated into groups of persons with masks and people without masks, were used in the suggested face mask detection model dubbed SSDMNV2. Having been completed successfully The deep neural OpenCV approach. The networks utilized in this model produced beneficial outcomes. Categorization The MobilenetV2 image classifier was used to properly classify pictures. This is one of the suggested approach's distinguishing features [1].

Bhambaniet. al. have presented in their study that they developed a, in which object detection is performed using YOLO v4. The 3 categories that Face that are both masked and uncovered are identified at the same time. as well as entire groups of individuals Using the coordinates provided by the identification of a classmate, the relative distance between two individuals is so assessed using optical concepts[2]

Militante et. al. have described in their paper that This research is provided an alert system using Convolutional Neural Networks and deep learning approaches. Networks. This procedure yields precise and timely results. detection of facemasks The results of the tests demonstrate a distinguished accuracy rate in spotting folks wearing a facemask and not wearing a facemask putting mask recognition model was successful. its endeavor with the VGG-16 CNN model attaining a for the result is a score of 96 percent was obtained. Furthermore, the research provides a covid 19 accessories virus by determining that an alert is triggered if the individual is not wearing a facemask [3].

Vinh et. al. has presented in their paper that this work is to detect the mask of the people. The introduced model here with the algorithm that gets the employee's image enhancement technique to improve the accuracy of the following system project. For the advancement of the YOLOv3 network algorithm, the system is working in real-time smoothly with 30 frames per second speed. This is the reason the system is reducing to get mask detection[4].

Kollreider et. al. has described in their work that they introduced a revolutionary real-time approach for face detection in this paper. However, the approach may be used as a universal image-object detector. The fact that it is simple to use to identify mouths lends experimental credence to this viewpoint. The characteristics were investigated by experimentation, and we showed information richness as evaluated by judicial qualities and robustness to the effects of extreme lighting changes. which increases their computational benefit. [5].

Suresh et. al. has described that in research, we presented a process that detects whether or not a person is and alerts if they are not. It suggested the system employs a Computer Vision system and Part of a general to assist the public in guaranteeing that by mask-wearing persons and to prevent the transmission of the COVID-19 virus. Our study also supports security and identifying mask or non-mask people in the crowd.

and will be able to take further action. The suggested system may be used in venues such as train stations, retail malls, workplaces, schools, and airports [6].

Fang et. al. has presented in their work that A system based on is given in this paper. The system can recognize the mask-wearing status of people appearing in a camera feed or video stream in general, and indicate the result on the video output. The face mask identification approach employs Har cascade classifiers. investigated in an attempt to decrease system detection. Finally, the detection system is shown to be capable of running at 45.80fps in 1080P, with a detection latency of 0.13s and a detection precision of 96.55 percent. The method can be updated in the future to include [7].

Yadav et. al. has described in their study that protecting individuals by automatically monitoring public places to prevent the transition of corona and security by reducing physical surveillance is required. Real-time camera feeds with a are required. Thus, under the current circumstances, this proposed method will run efficiently and aid in tracking when the lockout is lifted. Automated access to public locations is possible. We have gone through the tracking of social distance and the identification of social distancing in great detail [8].

Medieval et. al. has presented in their paper that When utilizing the Faster R-CNN algorithm to create shows who is unmasked and who is not keeping. file, the social distance is set to 3 m. the social distance was measured 50 times to ensure accuracy. The distance between 2 object motions. When capturing a complicated image, the Faster R-CNN algorithm is used to modify the contrast and brightness video stream with mask were easily recognized with 93.4 percent accuracy using

Faster CNN when diverse images and different color images happened. Both Mask Detection Applications [9].

In their study, Jiang et. al. has presented that the use of real-time mask detection is growing. To solve this issue, we built the PWMFD with high-quality masked face photos and created SE-YOLOv3, a fast and accurate mask detector with a channel attention mechanism that improved the backbone network's feature extraction capability. Furthermore, we applied GIoU and focal loss and the accompanying data augmentation to increase the model's accuracy and resilience. In the future, we will gather additional data and strike a balance between different types of data to enhance the PWMFD. Furthermore, we will take parameters and flops into account and deploy SE-YOLOv3 on lightweight devices, which can further improve performance [10].

Sethi et. al has described in their work that both the FMY3 and FMNMobile detection models achieve high accuracy, although the FMY3 is entirely dependent on the annotations of the mask in the photos. The primary reason for selecting another idea is that we first identify human faces using the cutting-edge Resnet SSD300 model, and then we utilize NASNetMobile as the backbone architecture to forecast face masks for classification tasks. When compared to the FMY3, FMNMobile delivers superior accuracy with a recall rate of 98 percent. Image 8 and Image 10 clearly show a little change in bounding box predictions, and in the case of FMNMobilenet, bounding boxes are predicted rather effectively. The FMY3 model, on the other hand, may be used alone to identify face masks [11].

Bhuiyan et. al. has presented in their work that we have devised a method for determining. It performs well in photos, and our detection findings, Inside the video, our model produces remarkable results with a frame rate of 19. This research aims to create a model of utilizing to create the entire architecture. Even though the dataset we acquired is not particularly diverse, the best accretes in testing with some random machine learning techniques [12]

2.3 Comparative Studies

The project is s massive result of Python library functions. We have concentrated more during the development of our project. We worked on every simple need and python approach of that site to work our real-time face mask detection of different users.

We investigated and discovered various comparable duties for us, however, all precision and security are lacking in the Covid situation. In this pandemic period, we have to ensure mask-wearing practice in every aspect. Our project is built for recognizing faces with a mask in them.

The majority of the enterprises are in face mask detection.

Our venture is capable of distinguishing between confrontation and perception of a mask or without a mask.

Also, it can measure the percentage of the mask and not the mask of the user.

2.4 Scope of the problem

We built the project to make this project creates for it to properly recognize the asks in the user's face. As we all know, PC read image data in the show that from the webcam.

As a result, a few faces unwinding clusters are almost the same result.

Exactness is less than 70%. Similarly, for this attempt, we must get it. But the project is attempting over 100% accuracy. The system is performing well in the mask detection observation.

The following functionalities are required for the project we building here,

- Pandas
- OpenCV
- NumPy
- TensorFlow
- Keras

The project is basedonpythonanditwillrunfrequentlyonanyPersonalComputerorLaptopwith a ram with a minimum of 4 GB and the required ram is 8GB.

As a result, Detection is less than 80%. Similarly, we must get this library in any case for this attempt.

After the libraries can be used to make the project much easier. Here is some short description about these,

□ **Pandas**: Pandas are powerful, powerful, versatile, and simple to use for machine studies.

□ **OpenCV**: OpenCV is a massive open-source computer vision, machine learning, and image processing library. OpenCV is compatible with a broad range of programming languages, including Python. When it is paired with other libraries, such as Numpy, a highly efficient library for numerical operations, the number of weapons in your arsenal grows.

Numpy; NumPy, which stands for Numerical Python, is a library that contains multidimensional array objects as well as a set of algorithms for manipulating such arrays. NumPy can conduct mathematical and logical operations on arrays. NumPy is a Python library.

TensorFlow: It features a robust, flexible ecosystem of tools, libraries, and community resources that enable academics to push the boundaries of ML and developers to quickly construct and deploy ML-powered applications.

□ **Keras**: Keras adheres to best practices for lowering a cognitive load: it provides consistent and straightforward APIs, reduces the number of user activities necessary for typical use cases, and delivers clear and actionable error signals.

2.5 Challenges

When a machine learning project is trained, it acts like a machine. As a result, it is critical to educate and MLto accept inputs, we must train in real-time.

After getting the images in real-time in the webcam there will be a rectangular shape in the faces of the user and then it will recognize the face mask from the user. If the user wears a mask the system will immediately provide a green rectangular box and shows the accuracy of that wearing face mask.

CHAPTER 3

REQUIREMENT SPECIFICATION

3.1 Process and Modeling

Ours is a python project with ML, and the project is an automatic face mask detection system based on python OpenCV face recognition. The system operates auto and includes real-time face mask identification to check whether the user is wearing a mask or not. This system's graphical user interface saves time. And the most interesting fact it will provide percentage value on the user mask. Simple to use. The following system is built basically in Python libraries.

3.2 Machine Learning

ML is called machine theory because of research that provides devices like computers with the capacity for learning without being explicitly programmed. They're frequently trained to deal with a variety of life concerns. People have become accustomed to performing Machine Learning tasks by now.

All of the computations and numerical and applied arithmetic equations are physically mysteriously composed. As a result, the process became time-consuming, dull, and wasteful. In any case, in these modern times, In comparison to past days, different Python libraries have gotten richly straightforward and efficient. is superior to AI libraries that have been used.

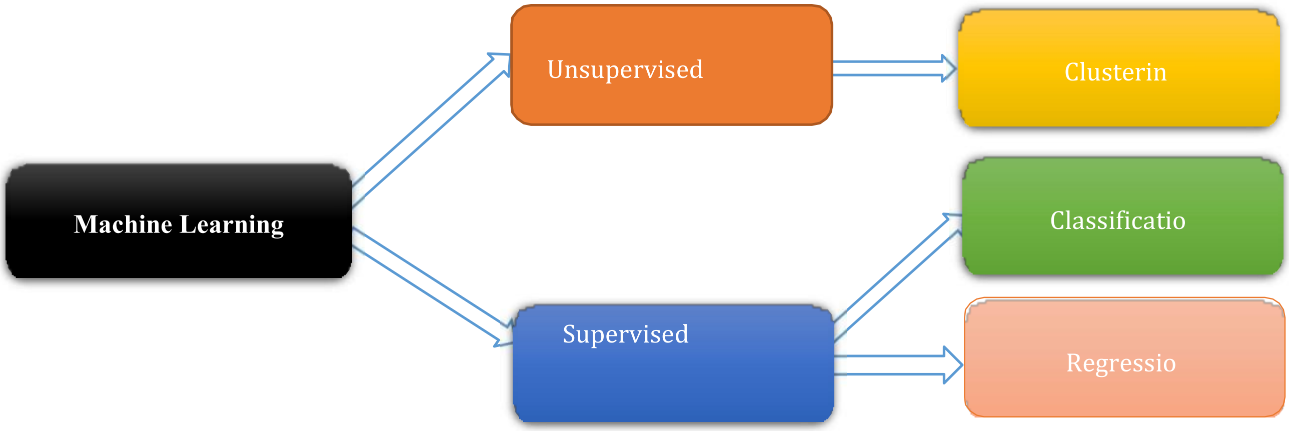


Figure 3.2.1: Flow of ML

3.3 ML Libraries & logical data model

ML is research that provides devices like laptops with the capacity for learning without being explicitly programmed. They are often accustomed to resolving many types of life challenges. In the past, individuals were accustomed to performing Machines. Commitment to writing all of the computations, numerical, and applied scientific recipes. This resulted in the technique being time-consuming, boring, and wasteful. Anyway, it's becoming popular these days.

In comparison to past days using many python libraries, this is essentially simple and inexpensive. structures, as well as modules the following Python libraries, are used in Machine Learning:

- OpenCV
- TensorFlow
- Pandas
- Numpy
- Matplotlib
- Keras

OpenCV

Then Open Computer Vision (OpenCV) is a computer vision with ML, and image processing, and it currently has a significant impact on the continuous process, which is incredibly important in today's frameworks. It may be used to systematize images and videos.

order things, faces, or, in any case, human handwriting When it formed forces with numerous organizations, to investigate We employ vector space to discern picture structure and its various congregations and execute an approximated process on these highlights. OpenCV recognizes. OpenCV will certainly open up the camera and start recognizing masks in the user ahead at the camera.

Pandas

software suite built for the Python computer. It provides information formats and approaches for influencing numerical tables and time scheduling in particular.

When we need to utilize pandas for data analysis, we usually use

1. The major reference in python or a NumPy used to extract panda data

2. Use pandas to a local file, which is often a CSV file. But we have used jpg or png image files to detect the image had a mask on it or not.

Similarly, be a book record holder, beat expectations, and so on. Pandas are used to save our involvement.

3.4 PythonlibraryIDE:

IDE is software that controls whole offices for software developers to progress their programming. An IDE often includes at the very least a source code editor, assembly by the user to use it frequently. This job requires

We used the PyCharm python compiler tool. The task also can be done in Collab as well.

Python Library:

Python libraries are a collection of capabilities and approaches that allow you to perform several tasks while without writing code. As an example, the Python Imaging Library (PIL) is one of the main libraries for image handling in Python. Pad is an excellent companion degree.

PIL fork was established. OpenCV python may be a python library aimed for constant pc vision.

as well as image processing Each library provides a large number of supporting modules that may be imported.

There are some most used machine learning libraries,

□ Matplotlib
□ Scipy
□ Scrapy
□ NumpySQLA
□ lchemy
□ Pandas
□ PIL
□ TensorFlow

Five of the libraries were used for our projects. TensorFlow, Numpy, and Pandas are the names of these characters.

TensorFlow:

TensorFlow is widely used in machine learning, notably the deep learning algorithm family. These will take a long time to complete, which is where GPUs come in because they give faster processing speed than CPUs. For this, the OpenCV GPU module is utilized. You will be the one who decides which classes and methods to employ, therefore some familiarity with GPU programming is required. You can simply use the GPU implementation in TensorFlow by specifying the number of GPUs you have or if you want to use both.

Overall, if you truly want to understand how computer vision works from the beginning to the end.

Open Computer Vision toolbox. It has built-in classes and methods for image and video processing and analysis. OpenCV also supports machine learning. The majority of its built-in machine learning algorithms are designed for its core use, computer vision

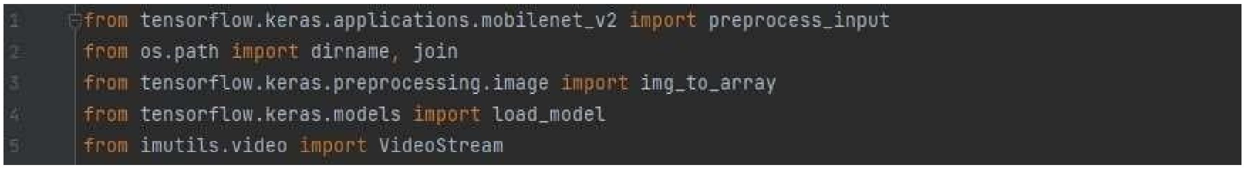
TensorFlow libraries in our project have shown below,

from TensorFlow. Eras. applications. mobilenet_v2 import preprocess input

from os. path import dirname, join

from TensorFlow.Keras.preprocessing.image import img_to_array

from TensorFlow. Keras. models import load model



```
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from os.path import dirname, join
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model
from imutils.video import VideoStream
```

3.4.1 TensorFlow

NumPy:

NumPy is a Python application for logical calculating. NumPy powers massive, relatable exhibitions and grids. When compared to Python records, NumPy clusters are faster.

NumPy is widely regarded as one of Python's very common Machine learning libraries.

TensorFlow, OpenCV, and other libraries make use of Numpy to do various tasks.

Tensors and OpenCV are used in this process. The greatest and most important component is the cluster interface of the number

Features of Numpy

- Array objects
- sophisticated operation.
- The Fourier transform is all useful.
- It makes coding genuine simple, and it is simple to gain a grip on the notions.
- Numpy is sophisticated and easy to use.
- Varieties of sophisticated scientific activities are fundamental.

Numpy libraries in our project have given below,

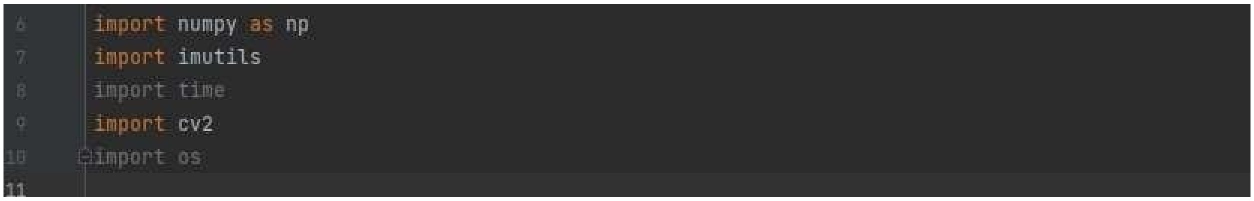
import NumPy as np

import imutils

import time

import cv2

import os



```
import numpy as np
import imutils
import time
import cv2
import os
```

3.4.2 NumPy

3.5 Data Flow Diagram the design requirement

It is called communicating with a data flow via a procedure or framework. The DFD also provides information on the yields and contributions of each ingredient as well as the method itself. A data flow sheet can be a graphical representation of the "flow" of information through the ANIF.

views on technique DFD is frequently used to a basic breakthrough in the creation of a blueprint. while not going into great detail on the structure DVD

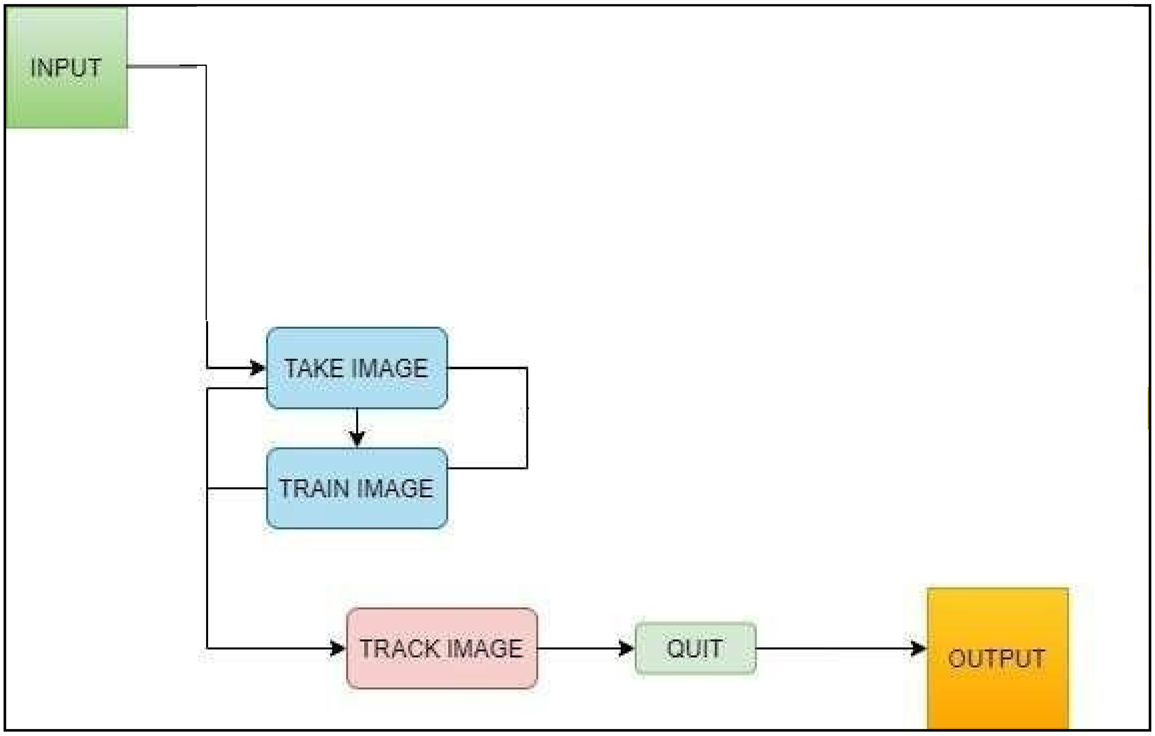


Figure 3.5.1: DFD of face mask detection project.

CHAPTER 4

DESIGN SPECIFICATION

4.1 Front – End Design

It is essentially an organization or project-style framework. It generates as a User Interface design. This is the primary draw of a project. This is the means of communication of the user and the outside of the design, also known as client experience design

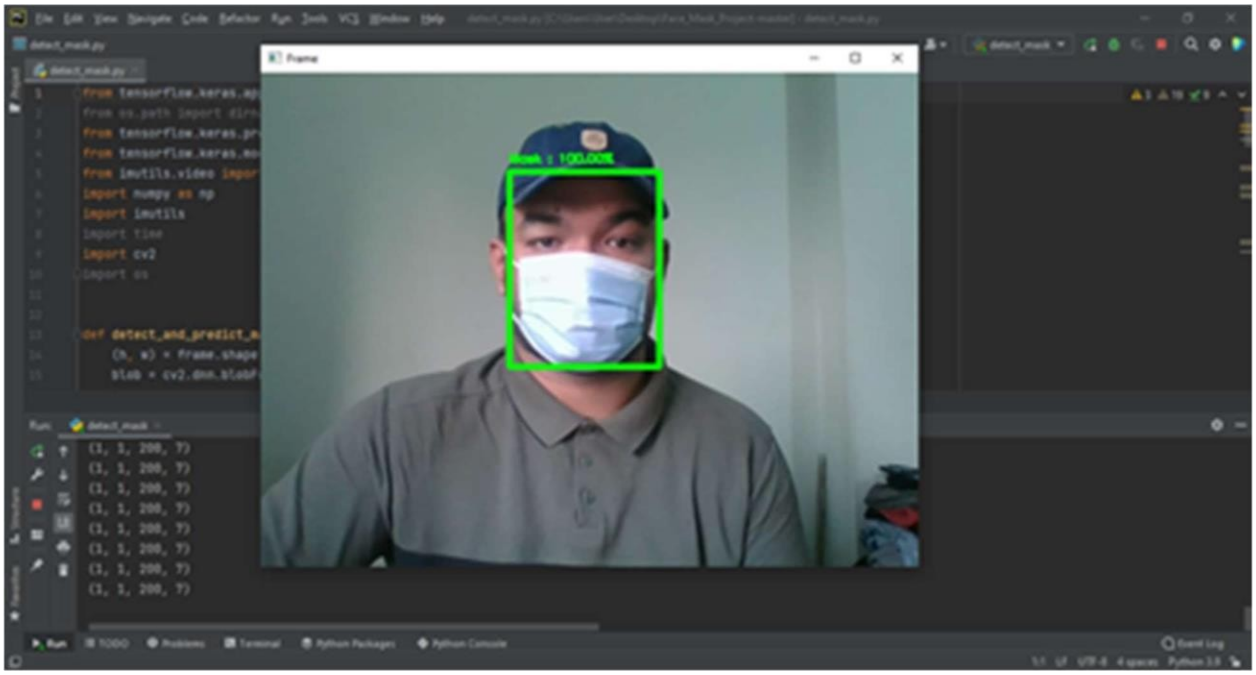
A developmental detour. We created a graphical user interface (GUI) depending on the outlook of the project.

This project is simple.

In the following project, we will generate a webcam display that will respond intently after running the system and starts detecting mask on the user's face. If the use wears any kind of mask on his face it will respond.

Features with Face Mask:

The GUI of the project with webcam with turns "green" and shows the percentage of the mask. There will be a rectangular space in the display around the face of the user. There will be an accuracy percentage of the mask value in the display.


File Edit View Navigate Code Refactor Run Tools VCS Window Help
detect_mask.py
Frame
Mask : 100.00%
Run: detect_mask
Run TODO Problems Terminal Python Packages Python Console

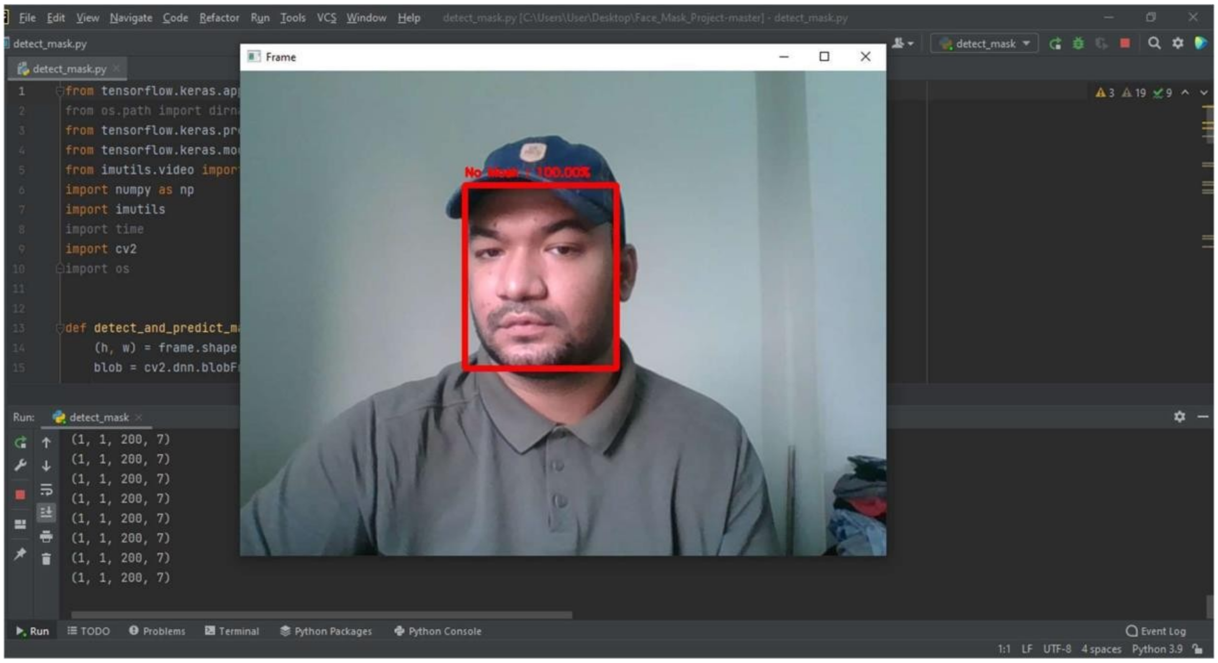
```
from tensorflow.keras.ap
from os.path import dirn
from tensorflow.keras.pr
from tensorflow.keras.mo
from imutils.video impor
import numpy as np
import imutils
import time
import cv2
import os


def detect_and_predict_m
    (h, w) = frame.shape
    blob = cv2.dnn.blobF
```

```
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
```

4.1.1 Front-end Design

Features without Face Mask:

The GUI of the project with webcam with turns “red” and shows the percentage of the mask. There will be a rectangular space in the display around the face of the user same as the mask criteria. There will be an accuracy percentage of the no mask value in the display.


File Edit View Navigate Code Refactor Run Tools VCS Window Help
detect_mask.py [C:\Users\User\Desktop\Face_Mask_Project-master] - detect_mask.py
detect_mask.py
detect_mask
Frame
No Mask: 100.00%
Run: detect_mask
Run TODO Problems Terminal Python Packages Python Console
Event Log
1:1 LF UTF-8 4 spaces Python 3.9

```
from tensorflow.keras.ap
from os.path import dirn
from tensorflow.keras.pr
from tensorflow.keras.mo
from imutils.video impor
import numpy as np
import imutils
import time
import cv2
import os


def detect_and_predict_m
    (h, w) = frame.shape
    blob = cv2.dnn.blobF
```

```
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
```

4.1.2 Frontend design

4.2 Back End Design

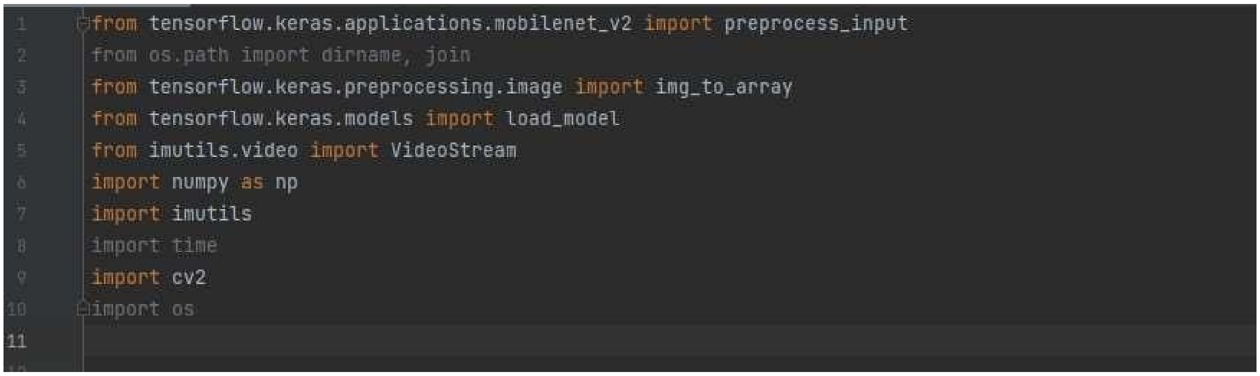
When the back part is completed appropriately, a project is completed. Back-end design is a critical component of every project. The back end is the most important element.

In the following project, we employed a sort of ML packages in the back-end design.

The major python libraries have been inserted in this automated face detection system. We have also implemented the available libraries of Numpy, OpenCV, TensorFlow, Keras, and OS.

The back-end coding has been done with Python and completely done with PyCharm, The project has all the necessary imports of Numpy, TensorFlow, Keras, OpenCV, Imutils, OS, cv2, and many more.

The first part of the back-end code contains the needed functionalities that help the system run simultaneously.



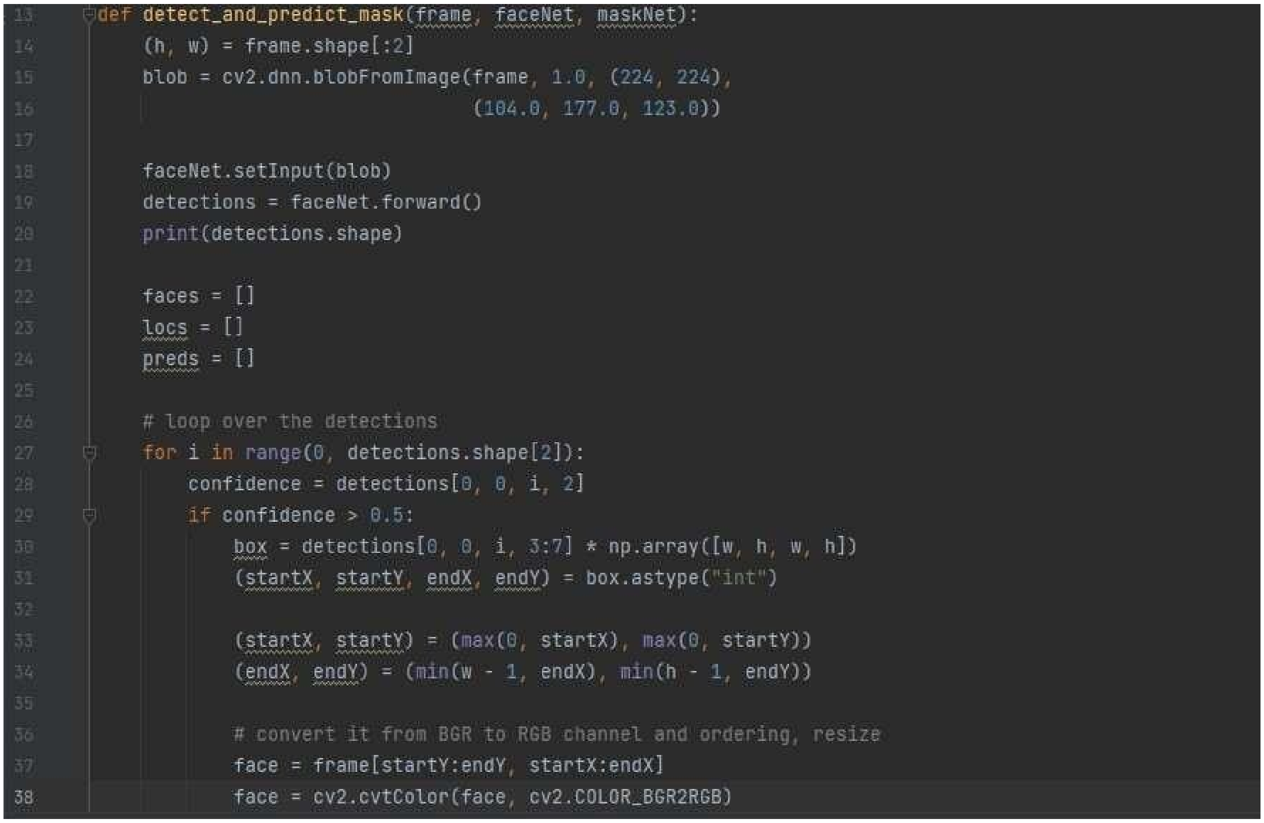
```
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from os.path import dirname, join
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model
from imutils.video import VideoStream
import numpy as np
import imutils
import time
import cv2
import os
```

Figure 4.2.1: Python libraries in Back End Design.

We have created a def part that is called “detect_and_predict_mask”. Which is includes frames, facets, and markets.

After the def part, we have to build a loop function including an if-else statement for the detection.

In the following part, we will convert the BGR to the RGB channel and order the project to work properly.



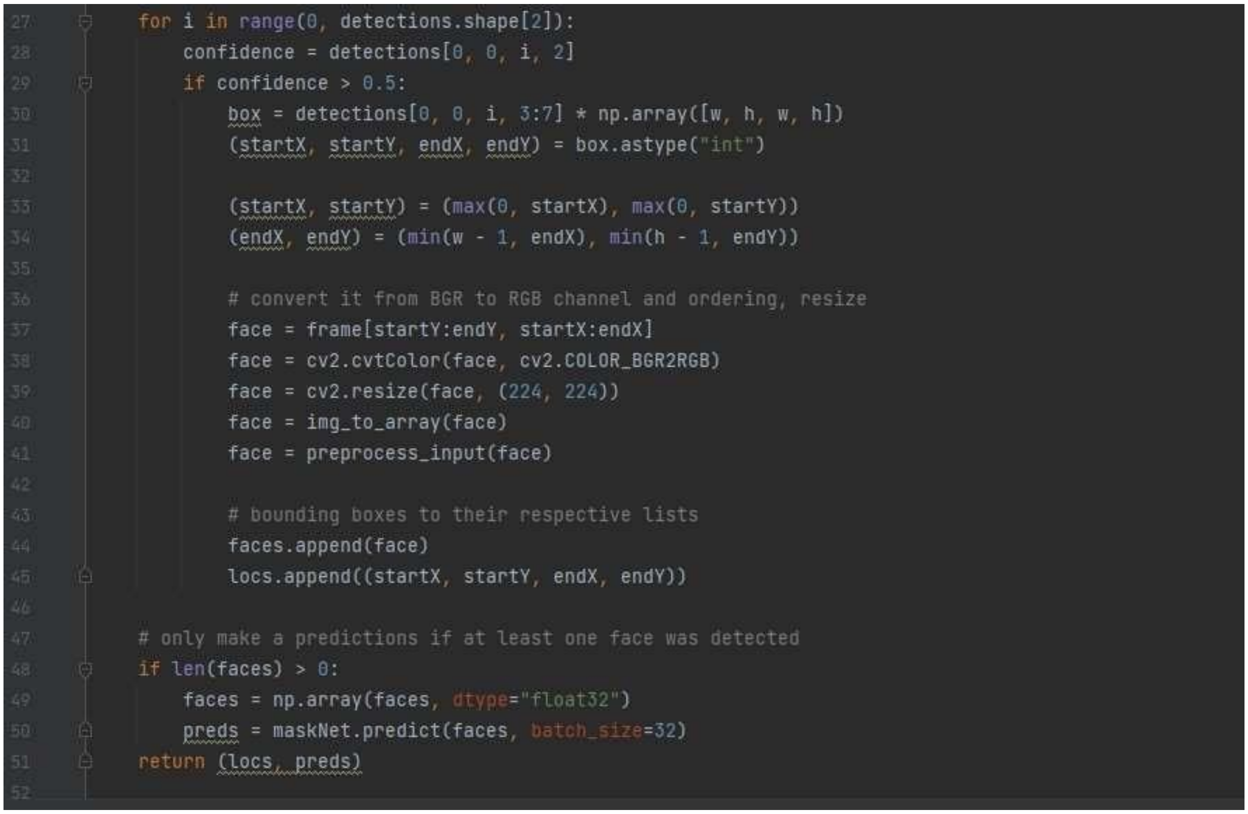
```
def detect_and_predict_mask(frame, faceNet, maskNet):
    (h, w) = frame.shape[:2]
    blob = cv2.dnn.blobFromImage(frame, 1.0, (224, 224),
                                 (104.0, 177.0, 123.0))

    faceNet.setInput(blob)
    detections = faceNet.forward()
    print(detections.shape)

    faces = []
    locs = []
    preds = []

    # loop over the detections
    for i in range(0, detections.shape[2]):
        confidence = detections[0, 0, i, 2]
        if confidence > 0.5:
            box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
            (startX, startY, endX, endY) = box.astype("int")

            (startX, startY) = (max(0, startX), max(0, startY))
            (endX, endY) = (min(w - 1, endX), min(h - 1, endY))

            # convert it from BGR to RGB channel and ordering, resize
            face = frame[startY:endY, startX:endX]
            face = cv2.cvtColor(face, cv2.COLOR_BGR2RGB)
```

Figure 4.2.2: Def part of the Back End Design.

In the loop part, we will be able to fin a rage with detection. Then with an if-else statement with a range of >0.5. Then convert the system from BGR to RGB channel and order.

Initializing the face values including cv2 (224, 244)

.



```
    for i in range(0, detections.shape[2]):
        confidence = detections[0, 0, i, 2]
        if confidence > 0.5:
            box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
            (startX, startY, endX, endY) = box.astype("int")

            (startX, startY) = (max(0, startX), max(0, startY))
            (endX, endY) = (min(w - 1, endX), min(h - 1, endY))

            # convert it from BGR to RGB channel and ordering, resize
            face = frame[startY:endY, startX:endX]
            face = cv2.cvtColor(face, cv2.COLOR_BGR2RGB)
            face = cv2.resize(face, (224, 224))
            face = img_to_array(face)
            face = preprocess_input(face)

            # bounding boxes to their respective lists
            faces.append(face)
            locs.append((startX, startY, endX, endY))

    # only make a predictions if at least one face was detected
    if len(faces) > 0:
        faces = np.array(faces, dtype="float32")
        preds = maskNet.predict(faces, batch_size=32)
    return (locs, preds)
```

Figure 4.2.3: BRG to RGB part of the Back End Design.

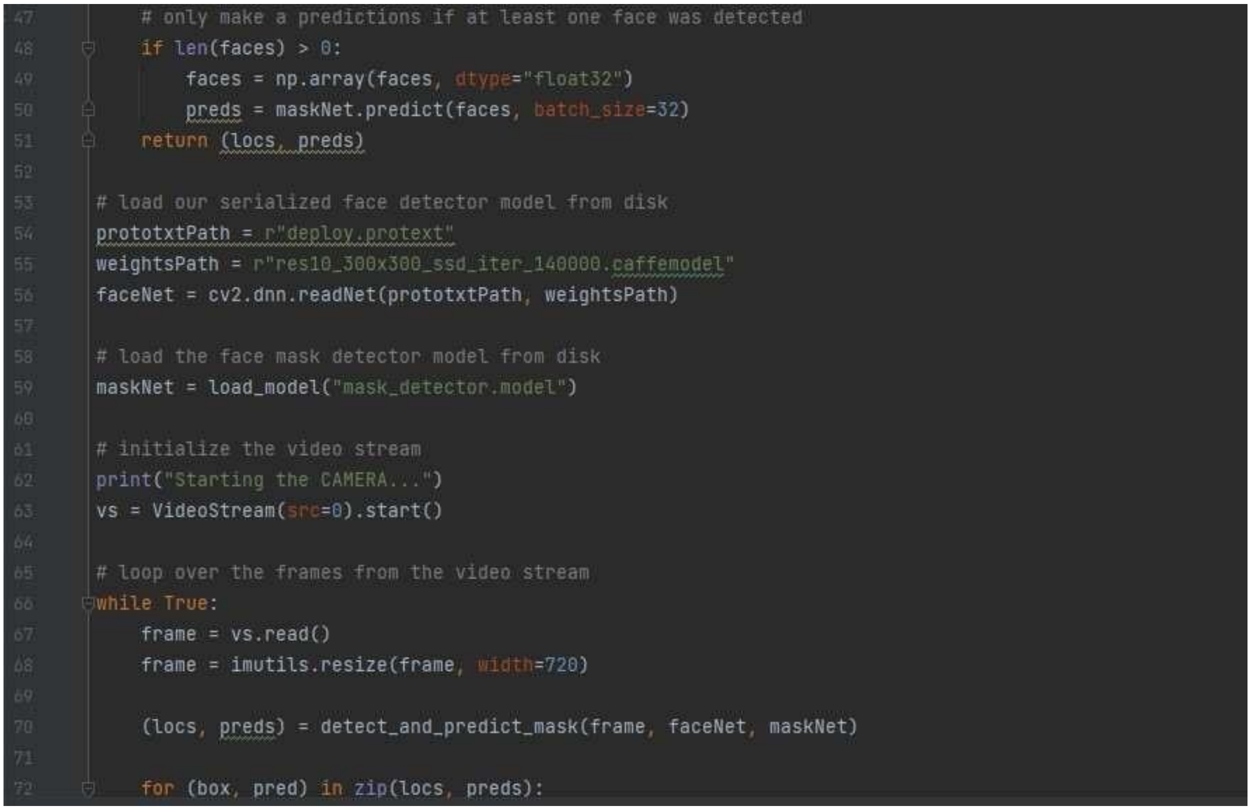
We have used another if-else statement for mask values from the face after recognition.

It will only predict if at least one of the faces of the user is detected with a mask or no

mask. Then it will return the value instead. After that, it will disk.

After that, it will load the face mask detector model from the disk that we have done earlier.

Then the system initializes the video stream with a webcam to detect the mask of the user. and provide an accuracy percentage if not it will show no mask and show no mask accuracy percentage as well.



```
    # only make a predictions if at least one face was detected
    if len(faces) > 0:
        faces = np.array(faces, dtype="float32")
        preds = maskNet.predict(faces, batch_size=32)
    return (locs, preds)

# load our serialized face detector model from disk
prototxtPath = r"deploy.protext"
weightsPath = r"res10_300x300_ssd_iter_140000.caffemodel"
faceNet = cv2.dnn.readNet(prototxtPath, weightsPath)

# load the face mask detector model from disk
maskNet = load_model("mask_detector.model")

# initialize the video stream
print("Starting the CAMERA...")
vs = VideoStream(src=0).start()

# loop over the frames from the video stream
while True:
    frame = vs.read()
    frame = imutils.resize(frame, width=720)

    (locs, preds) = detect_and_predict_mask(frame, faceNet, maskNet)

    for (box, pred) in zip(locs, preds):
```

Figure 4.2.4: Detector model and video stream Back End Design.

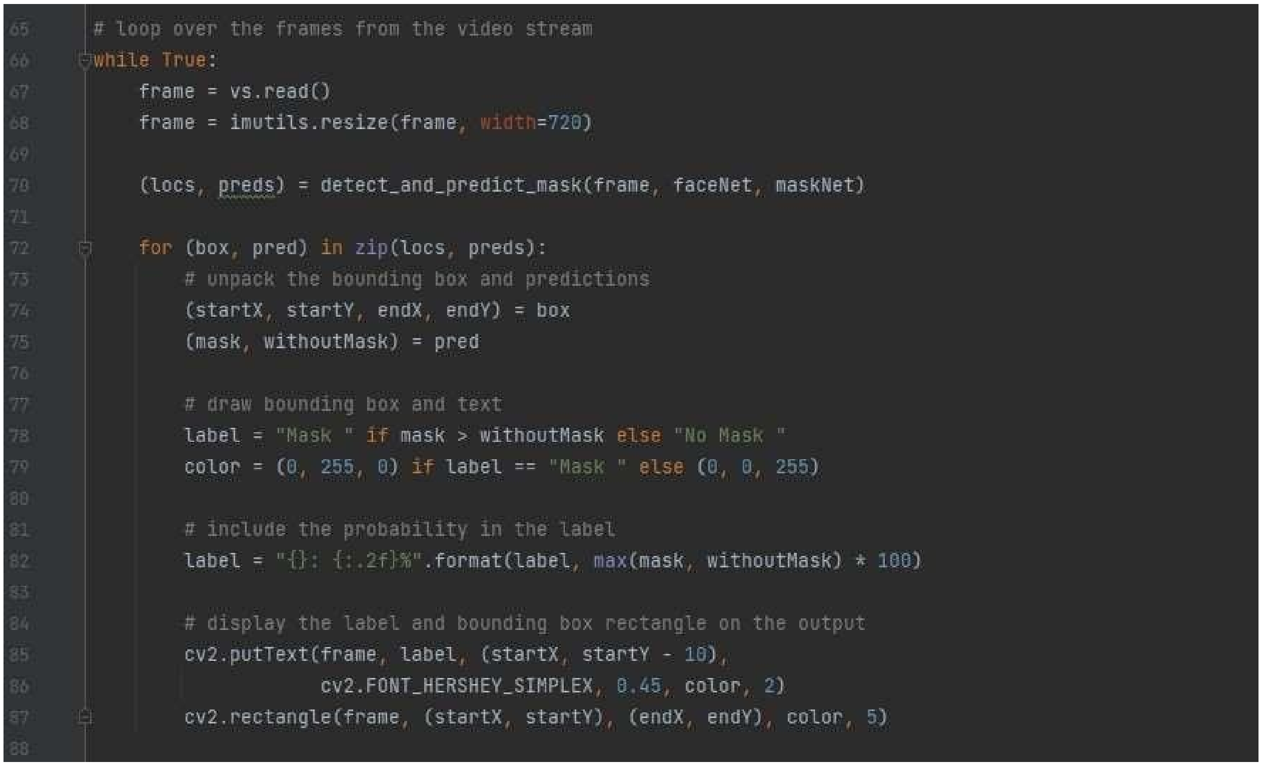
After taking the webcam video stream with a width of 720 pixels it will be a 720-pixel display of the webcam to visualize the mask detection system to configure the user’s mask from the outside.

Then unpack the bounding box and predictions of the system

After that drawing the bounding box and text of the system

Including the probability in the label as well.

Soon it will show the values.



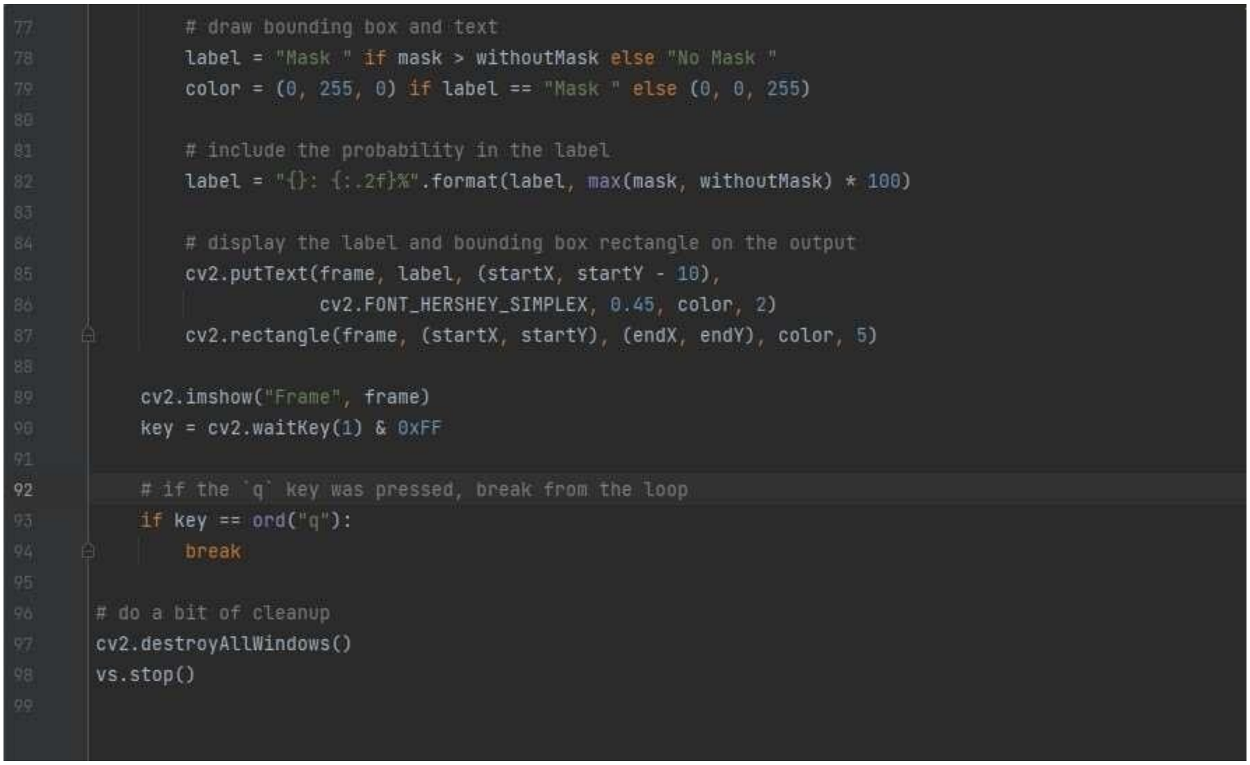
```python
# loop over the frames from the video stream
while True:
    frame = vs.read()
    frame = imutils.resize(frame, width=720)

    (locs, preds) = detect_and_predict_mask(frame, faceNet, maskNet)

    for (box, pred) in zip(locs, preds):
        # unpack the bounding box and predictions
        (startX, startY, endX, endY) = box
        (mask, withoutMask) = pred

        # draw bounding box and text
        label = "Mask " if mask > withoutMask else "No Mask "
        color = (0, 255, 0) if label == "Mask " else (0, 0, 255)

        # include the probability in the label
        label = "{}: {:.2f}%".format(label, max(mask, withoutMask) * 100)

        # display the label and bounding box rectangle on the output
        cv2.putText(frame, label, (startX, startY - 10),
              cv2.FONT_HERSHEY_SIMPLEX, 0.45, color, 2)
        cv2.rectangle(frame, (startX, startY), (endX, endY), color, 5)
```

Figure 4.2.5: Output values Back End Design.

Finally, the project is ready to run. After launching the project, the webcam of 720 pixels will display soon and start detecting masks in the user interface.

If it gets a mask on the user, it will show a percentage value if not it will show a no mask percentage value as well.

If the user presses the “q” key the loop of the system will break instantly and the system will end at that moment.



```
        # draw bounding box and text
        label = "Mask " if mask > withoutMask else "No Mask "
        color = (0, 255, 0) if label == "Mask " else (0, 0, 255)

        # include the probability in the label
        label = "{}: {:.2f}%".format(label, max(mask, withoutMask) * 100)

        # display the label and bounding box rectangle on the output
        cv2.putText(frame, label, (startX, startY - 10),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.45, color, 2)
        cv2.rectangle(frame, (startX, startY), (endX, endY), color, 5)

    cv2.imshow("Frame", frame)
    key = cv2.waitKey(1) & 0xFF

    # if the `q` key was pressed, break from the loop
    if key == ord("q"):
        break

# do a bit of cleanup
cv2.destroyAllWindows()
vs.stop()
```

Figure 4.2.6: Last display Back End Design.

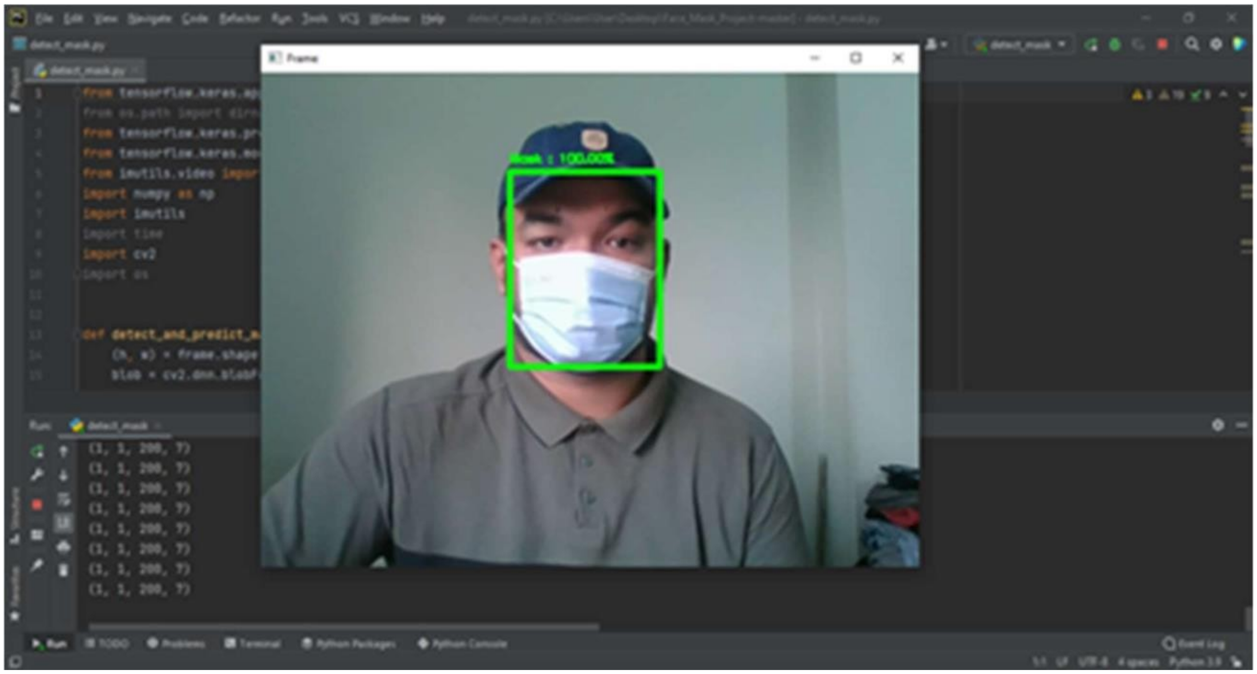
4.3 Infarction design and user experience

We will describe our project's experiment. The term "experimental outcome" refers to the process of determining whether or not a system's performance is adequate. The following is the experimental outcome of our project:

When we run the project, the first thing will come up with a webcam display to detect face masks from the user who is operating the project.

The webcam with turns “green” and shows the percentage of the mask. There will be a rectangular space in the display around the face of the user. There will be an accuracy percentage of the mask value in the display.

The webcam with turns “red” and shows the percentage of the mask. There will be a rectangular space in the display around the face of the user same as the mask criteria. There will be an accuracy percentage of the no mask value in the display.



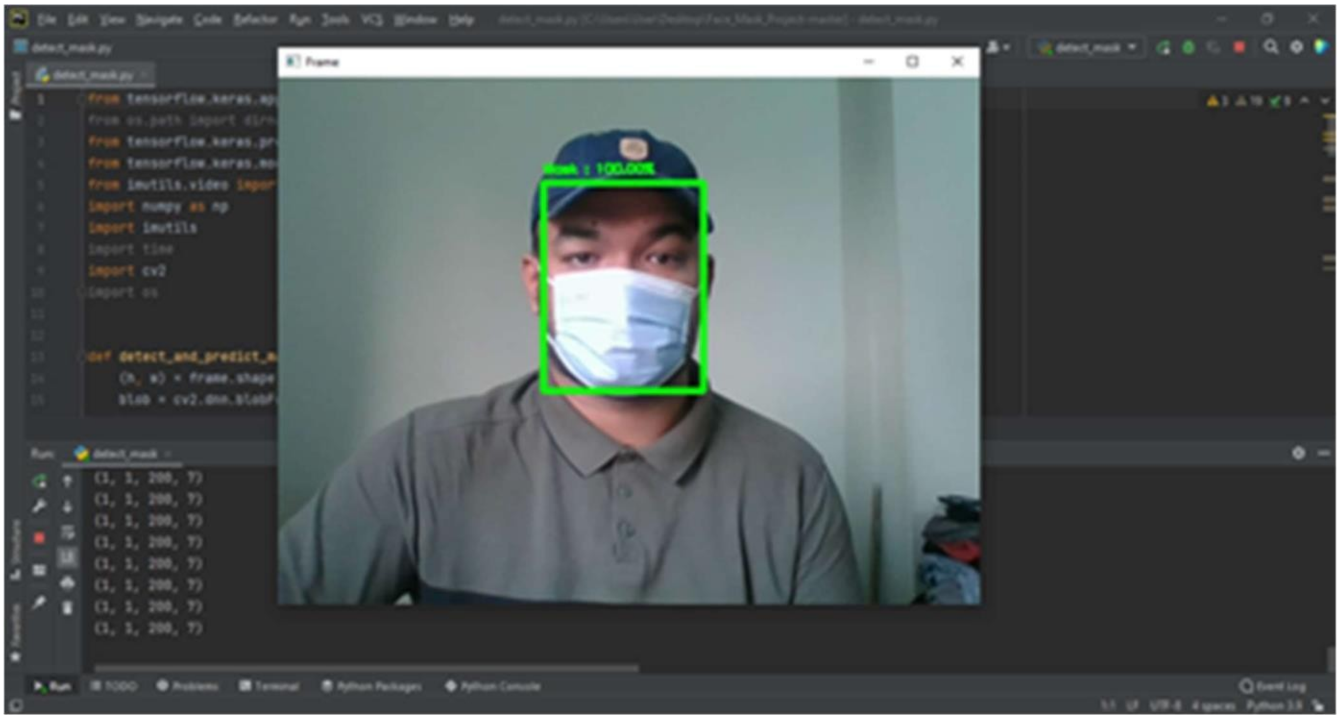
```
from tensorflow.keras.ap
from os.path import dirn
from tensorflow.keras.pr
from tensorflow.keras.mo
from imutils.video impor
import numpy as np
import imutils
import time
import cv2
import os


def detect_and_predict_m
    (h, w) = frame.shape
    blob = cv2.dnn.blobF
```

```
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
```

File Edit View Navigate Code Refactor Run Tools VCS Window Help
detect_mask.py
Frame
Mask : 100.00%
Run: detect_mask
Run TODO Problems Terminal Python Packages Python Console

Figure 4.3.1: Experimental result

After that will show the train image result:



```
from tensorflow.keras.ap
from os.path import dirn
from tensorflow.keras.pr
from tensorflow.keras.mo
from imutils.video impor
import numpy as np
import imutils
import time
import cv2
import os


def detect_and_predict_m
    (h, w) = frame.shape
    blob = cv2.dnn.blobF
```

```
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
```

Figure 4.3.2: Experimental result 2

4.4 Implementation requirement

The major python libraries have been inserted in this automated face detection system. We have also implemented the available libraries of Numpy, OpenCV, TensorFlow, Keras, and OS.

The back-end coding has been done with Python and completely done with PyCharm, The project has all the necessary imports of Numpy, TensorFlow, Keras, OpenCV, Imutils, OS, cv2, and many more.

The first part of the back-end code contains the needed functionalities that help the system run simultaneously. Our project takes a real-time face and stores it in the database.

CHAPTER 5

IMPLEMENTING AND TESTING

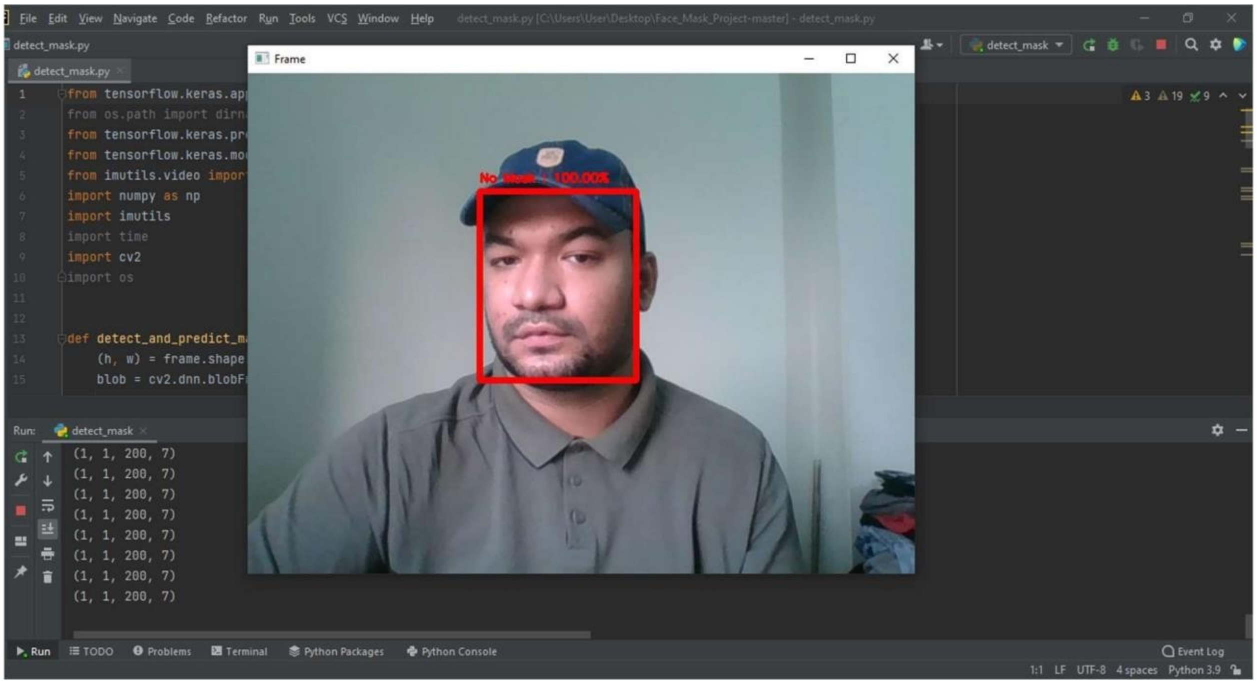
5.1 Take Real-Time Image with webcam

When a machine learning project is trained, it acts like a machine. As a result, it is critical to educate and MLto accept inputs, we must train in real-time.

After getting the images in real-time in the webcam there will be a rectangular shape in the faces of the user and then it will recognize the face mask from the user. If the user wears a mask the system will immediately provide a green rectangular box and shows the accuracy of that wearing face mask.

If the user is not wearing any mask at all. Then it will show a red rectangular box and shows an accuracy of not wearing any face mask at all.

The expected accuracy will be possible 100% by the way of wearing or not wearing a face mask.


File Edit View Navigate Code Refactor Run Tools VCS Window Help
detect_mask.py [C:\Users\User\Desktop\Face_Mask_Project-master] - detect_mask.py
detect_mask.py
detect_mask
Frame
No Mask : 100.00%
Run:
Run
TODO
Problems
Terminal
Python Packages
Python Console
Event Log
1:1 LF UTF-8 4 spaces Python 3.9

```
from tensorflow.keras.ap
from os.path import dirn
from tensorflow.keras.pr
from tensorflow.keras.mo
from imutils.video impor
import numpy as np
import imutils
import time
import cv2
import os


def detect_and_predict_m
    (h, w) = frame.shape
    blob = cv2.dnn.blobF
```

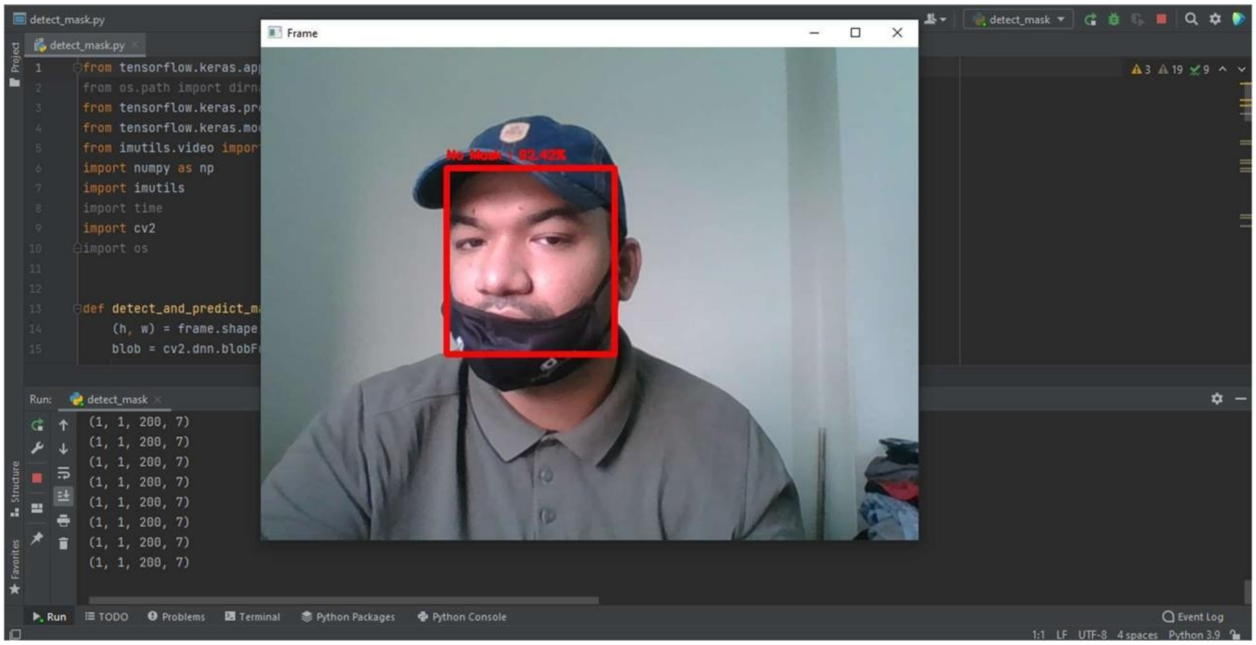
```
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
```

Figure 5.1.1: Take Image

5.2 Implementation of Front-end design

Machine learning algorithms may be used to train images. One of them, the LBPH face recognizer, is used in our project. It's quite simple to use. It is the simplest method for picture training since it is so basic and easy to grasp. This algorithm must meet certain conditions to run.

In our project, we are using real-time images from the user that has been working perfectly here.



```
from tensorflow.keras.ap
from os.path import dirn
from tensorflow.keras.pr
from tensorflow.keras.mo
from imutils.video impor
import numpy as np
import imutils
import time
import cv2
import os


def detect_and_predict_m
    (h, w) = frame.shape
    blob = cv2.dnn.blobF
```

```
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
```

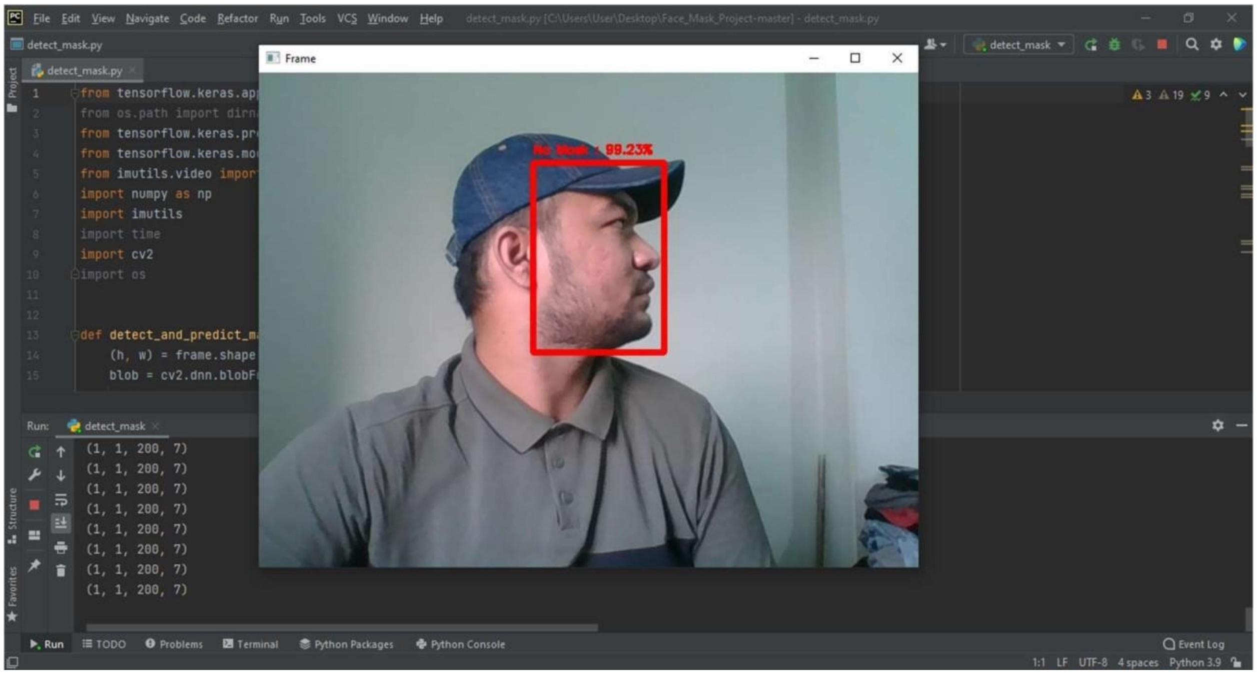
Figure 5.3.1: Train Image.

5.3 Testing Implementation

We have used OpenCV to detect face masks. It has an algorithm that can detect the face mask immediately.

In our project when taken images from the user from the webcam have trained the image in OpenCV.

After showing the image of the user from the webcam our project will be able to generate the accuracy of 100% detection rate of mask or no mask people.


File Edit View Navigate Code Refactor Run Tools VCS Window Help
detect_mask.py [C:\Users\User\Desktop\Face_Mask_Project-master] - detect_mask.py
detect_mask.py
Frame
No Mask : 99.23%
detect_mask
Run: detect_mask
Run
TODO
Problems
Terminal
Python Packages
Python Console
Event Log
1:1 LF UTF-8 4 spaces Python 3.9

```
from tensorflow.keras.ap
from os.path import dirn
from tensorflow.keras.pr
from tensorflow.keras.mo
from imutils.video impor
import numpy as np
import imutils
import time
import cv2
import os


def detect_and_predict_m
    (h, w) = frame.shape
    blob = cv2.dnn.blobF
```

```
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
```

Figure 5.4.1: FaceRecognize.

5.4 Test Results and Reports

The project findings in the right manner allow an add-on to assess testing outcomes quickly.Ourprojectrequiresimagesforalargenumberoftrainingthatpictureit can comprehend the images. Take a photo, a train picture in webcam and then after finishing the task of redetecting mask it will automatically shut down by pressing the “q”button.

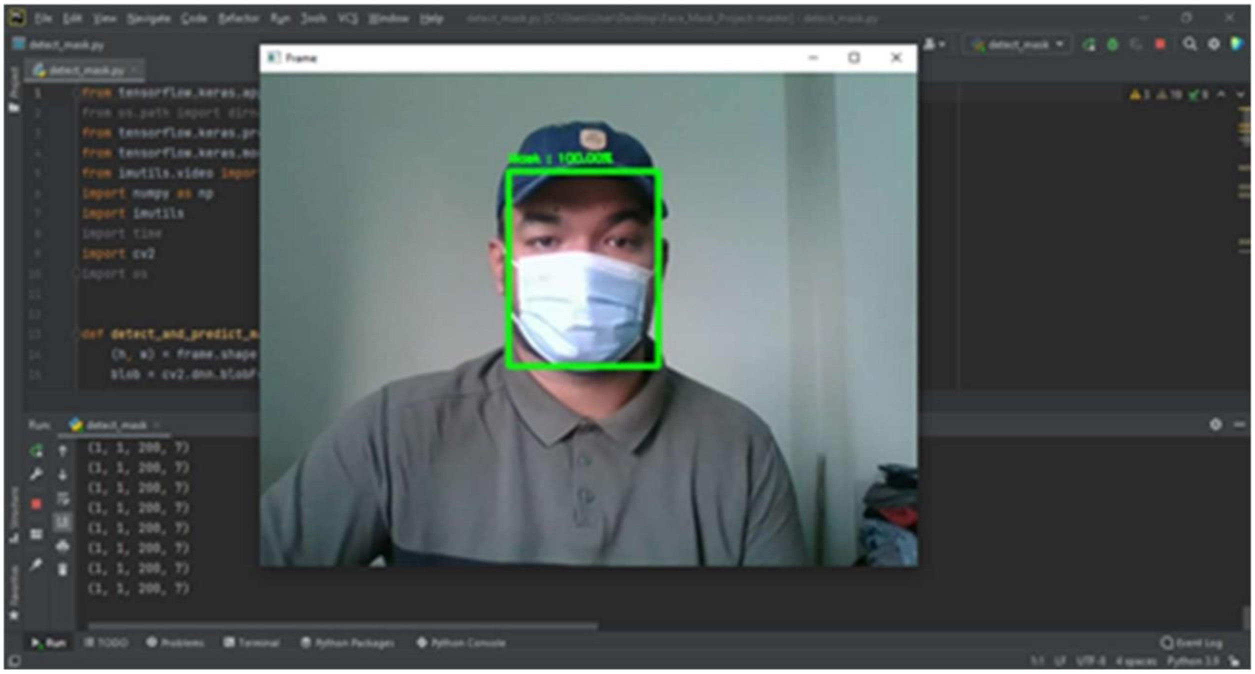
have been used to getting our images, and track catch will recognize the user image by webcam.

After getting the images in real-time in the webcam there will be a rectangular shape in the faces of the user and then it will recognize the face mask from the user. If the user wears a mask the system will immediately provide a green rectangular box and shows the accuracy of that wearing face mask.

If the user is not wearing any mask at all. Then it will show a red rectangular box and shows an accuracy of not wearing any face mask at all.

The expected accuracy will be possible 100% by the way of mask and unmask.

The LBPH computation aids in the preparation of images, and it was implemented in the OpenCV package



```
from tensorflow.keras.ap
from os.path import dir
from tensorflow.keras.pr
from tensorflow.keras.mo
from imutils.video impor
import numpy as np
import imutils
import time
import cv2
import os


def detect_and_predict_m
    (h, w) = frame.shape
    blob = cv2.dnn.blobF
```

```
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
(1, 1, 200, 7)
```

Figure 5.5.1: Test Result 1

Test Implementation

Tests	Inputs	Expected Outcomes	Obtained Outcomes	Passes/ Fails	Date of taste
1. Launch	Laptops/Desktops Python Compiler/PyCharm	Watch up the webcam	Successfully detected mask	Pass	23.10.2021
2. Library	Take Image	Successfully detected mask	Successfully detected mask	Pass	23.10.2021
3. Show Image in webcam	Detect Face Mask in the display	Successfully detected	Successfully detected mask	Pass	23.10.2021

Table 5.5.6: Test Implementation.

5.5 Implementation requirements

ML system is focused on machine learning; thus, we will require Python to complete it. it's an ML application that we constructed with a python field. Several prerequisites must be met for this project to be implemented. They are as follows:

We have used an environment for python that is PyCharm. It is a very efficient Python compiler.

Python implementation in PyCharm

- □ Import and download machine learning libraries.
- □ Python libraries should be downloaded and imported.
- □ webcam with a great resolution with 720 pixels.
- □ Laptop/Desktop computer with 4GB RAM and an idle CPU.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Discussion and Conclusion

This initiative aids in properly assessing involvement and is a long-term endeavor.

AI is being used in every industry, making our lives simpler and easier. Simply configure a web camera to capture images and train those images, which will be perceived as a result. This type of venture of the beta job This initiative will result in greater accuracy in facial mask recognition.

We have a Wearing mask, one of the major rules we have to maintain in this pandemic situation. We have created an ML project that will sense immediately if an individual. The project can be a crying need for this pandemic situation. The project has been built with python with the help of library functions like TensorFlow, OpenCV, Keras, and many more.

The project could be used in many aspects in this Covid 19 pandemic situation to detect masks from the user.

Finally, we endeavored to make this project excellent and professional for any kind of wor

6.2 Future Plan

Every time, all of the technology will be improved for the global benefit. This is in an age where technology is continually evolving, and machine learning is no exception.

Our system is built on the OpenCV library, TensorFlow and we want to upgrade it in the future. to compel us

Make the project more efficient, add new functionalities ,and improve its accuracy. There are some additions.

The Machine learning algorithms are improving day by day.Our project has been done with python and the help of library functions like TensorFlow, OpenCV, Keras, and many more.

The project could be used in many aspects in this Covid 19 pandemic situation to detect masks from the user.

We want to undertake the following in the future to improve the project with some more functionality:

- Detect various objects like face glasses or face makeup.
- Multiple eutectics multiple objects as well.
- Create a big dataset and import it with an expected huge out

REFERENCES

[1] Nagrath, P., Jain, R., Madan, A., Arora, R., Kataria, P., & Hemanth, J. (2021). SSDMNV2: A real-time DNN-based face mask detection system using a single shot multibox detector and MobileNetV2. Sustainable cities and society, 66,102692.

[2] Bhambani, K., Jain, T., &Sultanpure, K. A. (2020, October). Real-time Face Mask and Social Distancing Violation Detection System using YOLO. In *2020 IEEE Bangalore Humanitarian Technology Conference (B-HTC)* (pp. 1-6).IEEE.

[3] Militante, S. V., & Dionisio, N. V. (2020, August). Real-time facemask recognition with alarm system using deep learning. In 2020 11th IEEE Control and System Graduate Research Colloquium (ICSGRC) (pp. 106-110). IEEE.

[4] Vinh, T. Q., & Anh, N. T. N. (2020, November). Real-Time Face Mask Detector Using YOLOv3 Algorithm and Haar Cascade Classifier. In 2020 International Conference on Advanced Computing and Applications (ACOMP) (pp. 146-149).IEEE.

[5] Kollreider, K., Fronthaler, H., Faraj, M. I., &Bigun, J. (2007). Real-time face detection and motion analysis with application in “liveness” assessment. IEEE Transactions on Information Forensics and Security, 2(3),548-558.

[6] Suresh, K., Palangappa, M. B., &Bhuvan, S. (2021, January). Face Mask Detection by using Optimistic Convolutional Neural Network. In 2021 6th International Conference on Inventive Computation Technologies (ICICT) (pp. 1084-1089).IEEE.

[7] Fang, T., Huang, X., &Saniie, J. (2021, May). Design Flow for Real-Time Face Mask Detection Using PYNQ System-on-Chip Platform. In 2021 IEEE International Conference on Electro Information Technology (EIT) (pp. 1-5).IEEE.

[8] Yadav, S. (2020). Deep learning-based safe social distancing and face mask detection in public areas for covid-19 safety guidelines adherence. International Journal for Research in Applied Science and Engineering Technology, 8(7),1368-1375.

[9] Meivel, S., Devi, K. I., Maheswari, S. U., &Menaka, J. V. (2021). Real-time data analysis of face mask detection and social distance measurement using Matlab. Materials Today:Proceedings.

[10] Jiang, X., Gao, T., Zhu, Z., & Zhao, Y. (2021). Real-Time Face Mask Detection Method Based on YOLOv3. Electronics, 10(7),837.

[11] Sethi, S., Kathuria, M., & Kaushik, T. (2021). A real-time integrated face mask detector to curtail the spread of coronavirus. CMES-Computer Modeling in Engineering and Sciences, 389-409.

[12] Bhuiyan, M. R., Khushbu, S. A., & Islam, M. S. (2020, July). A deep learning-based assistive system to classify COVID-19 face mask for human safety with YOLOv3. In 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT) (pp. 1-5).IEEE.

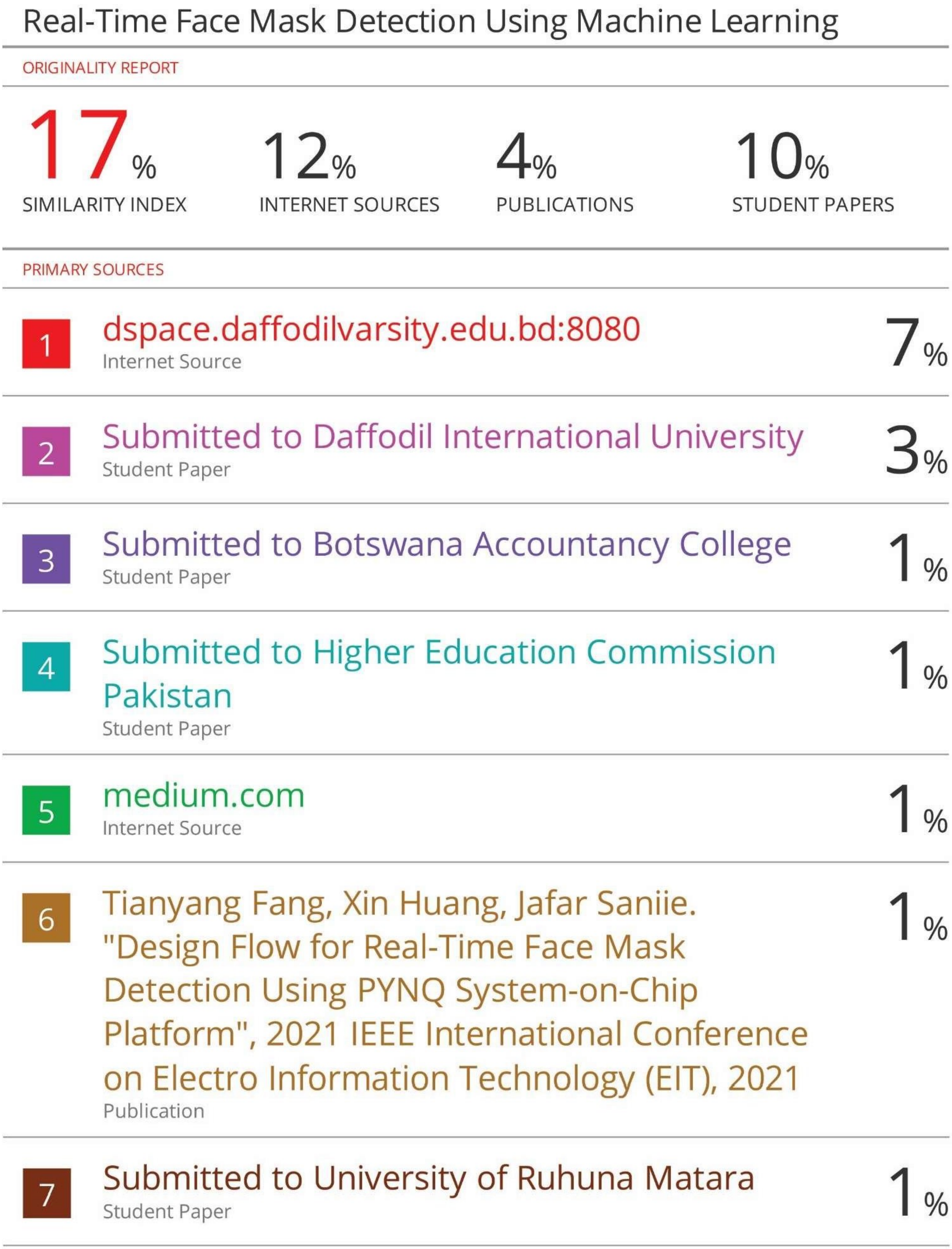
APPENDIX

Project Reflection

We had the opportunity to study data mining and machine learning in prior semesters. Because the courses are vital, we just introduce them routinely. However, even though it is a foundational subject, it has influenced us and piqued our interest in ML.

The discover uses it has. So, following, we present our concept to our supervisor, who then advises that we construct a mechanism for automatically taking attendance. we start to work, and complications in the daily life.

PLAGIARISM



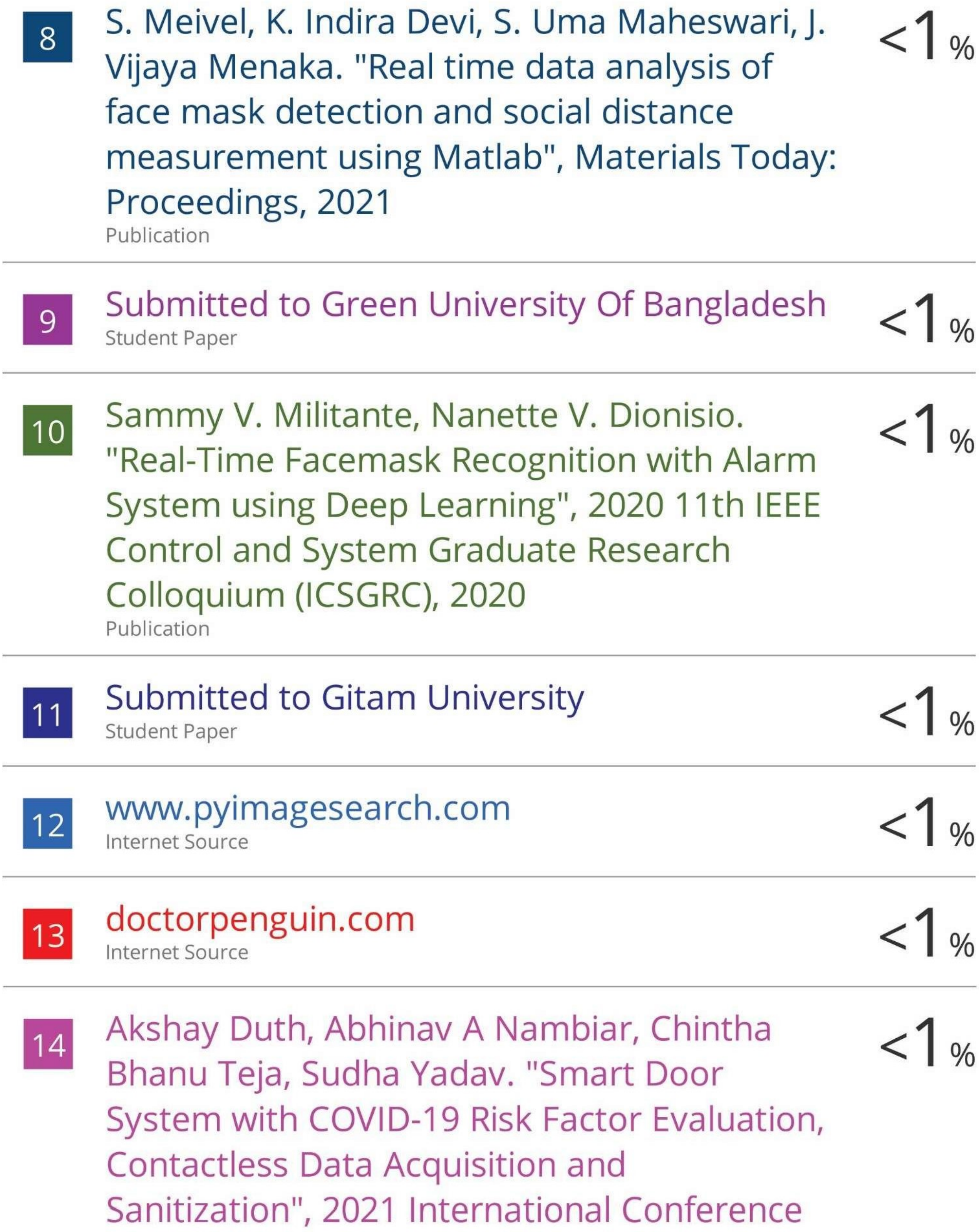
Real-Time Face Mask Detection Using Machine Learning

ORIGINALITY REPORT

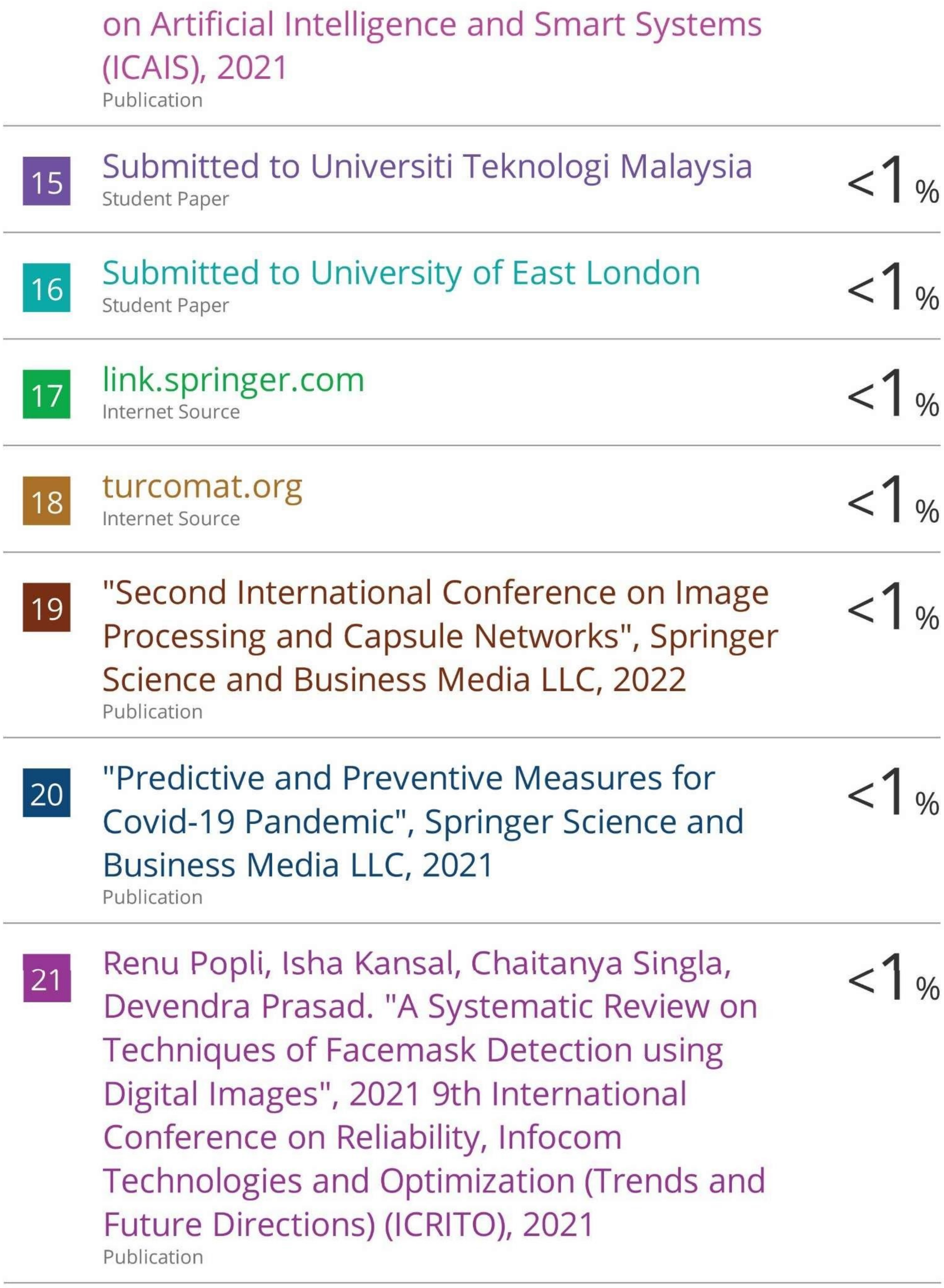
17%	12%	4%	10%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	dspace.daffodilvarsity.edu.bd:8080 Internet Source	7%
2	Submitted to Daffodil International University Student Paper	3%
3	Submitted to Botswana Accountancy College Student Paper	1%
4	Submitted to Higher Education Commission Pakistan Student Paper	1%
5	medium.com Internet Source	1%
6	Tianyang Fang, Xin Huang, Jafar Saniie. "Design Flow for Real-Time Face Mask Detection Using PYNQ System-on-Chip Platform", 2021 IEEE International Conference on Electro Information Technology (EIT), 2021 Publication	1%
7	Submitted to University of Ruhuna Matara Student Paper	1%



8	S. Meivel, K. Indira Devi, S. Uma Maheswari, J. Vijaya Menaka. "Real time data analysis of face mask detection and social distance measurement using Matlab", Materials Today: Proceedings, 2021 Publication	<1 %
9	Submitted to Green University Of Bangladesh Student Paper	<1 %
10	Sammy V. Militante, Nanette V. Dionisio. "Real-Time Facemask Recognition with Alarm System using Deep Learning", 2020 11th IEEE Control and System Graduate Research Colloquium (ICSGRC), 2020 Publication	<1 %
11	Submitted to Gitam University Student Paper	<1 %
12	www.pyimagesearch.com Internet Source	<1 %
13	doctorpenguin.com Internet Source	<1 %
14	Akshay Duth, Abhinav A Nambiar, Chintha Bhanu Teja, Sudha Yadav. "Smart Door System with COVID-19 Risk Factor Evaluation, Contactless Data Acquisition and Sanitization", 2021 International Conference	<1 %



	on Artificial Intelligence and Smart Systems (ICAIS), 2021 Publication	
15	Submitted to Universiti Teknologi Malaysia Student Paper	<1 %
16	Submitted to University of East London Student Paper	<1 %
17	link.springer.com Internet Source	<1 %
18	turcomat.org Internet Source	<1 %
19	"Second International Conference on Image Processing and Capsule Networks", Springer Science and Business Media LLC, 2022 Publication	<1 %
20	"Predictive and Preventive Measures for Covid-19 Pandemic", Springer Science and Business Media LLC, 2021 Publication	<1 %
21	Renu Popli, Isha Kansal, Chaitanya Singla, Devendra Prasad. "A Systematic Review on Techniques of Facemask Detection using Digital Images", 2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), 2021 Publication	<1 %