USAMA SAJJAD MIRZA
01-134172-057

Covid-19 Prevention: Social Distancing and Face Mask Detection

Bachelor of Science in Computer Science

Supervisor: Zubaria Inayat

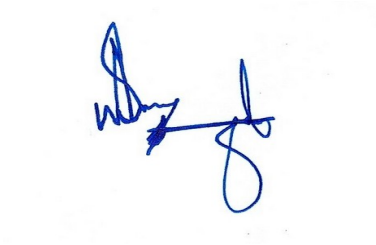
Department of Computer Science
Bahria University, Islamabad

May 2021

Bahria University
Islamabad Campus
(Department of Computer Science)

CERTIFICATE

We accept the work contained in the report titled "Covid-19 Prevention: Social Distancing and Face Mask Detection" as a confirmation to the required standard for the partial fulfillment for the degree of BS(CS)



Internal Examiner

Name: Dr. Saba Mahmood

Date: 8/7/21

External Examiner

Name:Dr. Waseem Shahzad_

Date:28/06/21

Project Coordinator

Name:_____

Date:_____

Head of Department

Name:_____

Date:_____

Abstract

This study examines a potential solution to the quickly spreading Coronavirus disease. The current societal landscape has gone through a drastic change due to Covid-19. The sudden appearance of the virus made the few things that have propelled humanity so far non-viable; communication and collaboration. People are no longer able to meet with each other or step foot outside which has resulted in the downward spiral of the economy. It is in the best interest of the people to try to mitigate the effects of the virus and return to life before the virus as soon as possible which can only be done so through prevention. This is why governments all over the world are looking for means to detect breaching of standard operating procedures on a large scale for densely populated areas, residential districts, large offices, and other enterprises to ensure safety. The purpose of this project is to design a system that detects whether the people in a public place maintain social distancing and whether every individual is wearing a face mask.

Acknowledgments

I would like to sincerely express my appreciation to several individuals and organizations for supporting me throughout my bachelor's study. First of all, I would like to thank my supervisor, Ms. Zubaria Inayat. Her guidance and practical advice have helped me immensely towards the completion of this project. Without her expertise and guidance, this project would not have been possible.

Secondly, I wish to thank Bahria university and all its staff for providing the necessary education and opportunity to even make this report possible.

Lastly, I would like to thank my family and friends for their constant unwavering encouragement and support throughout my studies.

Usama Sajjad Mirza
Islamabad, Pakistan

May, 2021

“Don’t bend; don’t water it down;
don’t try to make it logical; don’t edit your own soul according to the fashion.
Rather, follow your most intense obsessions mercilessly.”

Franz Kafka

Table of Contents

Table of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

The current societal landscape has gone through a drastic change due to Covid- 19 [1]. The quick transmission of the airborne Covid-19 has had effects similar to those of the depression era on both society and the economy. Huge corporations have gone bankrupt, stocks have plummeted, people have become unemployed [2]. The small percentage of organizations that have managed to avoid such fate has only done so through remote working [3]. Since more than 6 months have passed since the spread of the virus without a vaccine or cure, many countries have tried to slowly ease SOPs and ease lockdowns but this has backfired every single time resulting in further spread of the virus due to lack of individuals following preventive measures [4] [5]. Prevention is the only way that can allow individuals to be unaffected by the virus. This means maintaining 6 feet distance with every single individual and wearing face masks. If an individual ignores these preventive measures and contracts the virus the result is deadly [6].

We can ease the economy with the help of AI-driven automatic detection of individuals who are breaching SOPs. Implementing such a system would allow us to enforce SOPs while further lessening close human contact [7]. Deployment of this system in public spaces with lots of people would allow authorities to be less distracted to check for masks and social distancing on every individual and give more attention to apprehend individuals who breach SOPs. It'd allow for a more efficient approach to check or enforce SOPs in both open and closed spaces. It would also allow for

offices, universities, and commercial complexes to continue their operations safely [8]. The main objective of this project is to design a system that detects whether the people in a public place maintain social distancing and whether every individual is wearing a face mask.

1.2 Objective

The system we will be designing and implementing has the following objectives:

- To design a system that detects whether an individual is wearing a face mask
- To design a system that detects whether two or more individuals are maintaining a social distance of approximately six feet

The face mask detection and social distance detection modules will both use the MobileNetv2 object detection model since it is accurate and lightweight [9]. The face mask detection module uses a custom-trained model whereas the social distance detection module uses a pre-trained model.

1.3 Scope

The scope of the project is limited to using a laptop only. The cost to implement this project is high due to the high level of hardware required to implement it to a certain efficiency. The cost and efficiency of the algorithm used depend on the hardware at hand. Both cost and efficiency are directly proportional meaning if one requires more accurate detection the resources required and the cost will be equally high [10]. We will be leveraging the accuracy and resource efficiency of the MobileNetv2 architecture to develop a system that can easily be used on legacy systems, mobile systems, and embedded systems.

1.4 Existing Systems

Our market research resulted in the conclusion that there is a lack of an efficient and accurate prevention system for the coronavirus. There are multiple reasons for this such as high costs, lack of features, hardware, and contract locks. All these reasons have contributed to the low technology adoption by public safety departments. Our system aims to provide a solution to these drawbacks. For a more in-depth look at existing systems please refer to chapter 2.

1.5 Document Structure

The content of our document follows a certain schema to make sure it is easy to follow through, digestible and organized. The document structure is as follows:

- Introduction: This chapter gives a high-level view of this document. It includes an introduction to the problem we are trying to solve, the objectives we aim to accomplish, and

the scope in which we aim to accomplish it.

- Literature Review: This chapter aims to describe, summarize, objectively evaluate and clarify previous research related to our problem. It gives a theoretical base for the problem we are trying to solve and gives an overview of the current existing technologies similar to our system and briefly describes them.
- Requirement Specifications: In this chapter, we look over the existing system and its drawbacks and propose a solution that addresses said drawbacks. We also define the functional and non-functional requirements of our system and look at some of its use cases.
- Design: In this chapter, we give an overview of the design of the system. We look at its architecture, the design constraints, and the software/hardware requirements. We also go over the design methodology being used, a high-level diagram, a low-level diagram, and a sequence diagram of the system.
- System Implementation: In this chapter, we document all the implementation work of the system as well as describe the working of the system in detail. We look at the graphical user interface of our system as well as how each module works. In the end, we briefly explain the technical details of the system.
- System Testing and Evaluation: This chapter documents all the testing and evaluation work done on our system to ensure it is up to standards and is fulfilling all functional and non-functional requirements. Some of the tests we go over are unit testing, performance testing, black and white box testing, and more. We also look at some test cases for our system.
- Conclusion: This chapter aims to concisely explain the conclusive statement we reach. It lists some methods to improve our system and describes what we learned throughout the whole process of planning, designing, implementing, and testing our system.
- References: This chapter enlists all the sources used in this report. It allows for any interested individual to refer directly to the source work referred in the report and read it. This helps ensure that the information written in this document is accurate, complete, and credible.

Chapter 2

Literature Review

2.1 Background

Computer Vision is a sub-field of artificial intelligence in which we try to understand and replicate the tasks that the human visual system can do [11]. Early experiments in computer vision around the 1950s were fairly primitive, such as attempting to sort basic shapes into respective categories [12]. Today, Computer vision is a 50 billion USD industry (approx.), with applications in industries ranging from consumer electronics (smartphones & self-driving cars) and medicine (tumor detection and rare disease detection) to education (attendance and immersive teaching) and security (suspect identification and CCTV) [13] [14].

Computer Vision was revolutionized in the 1980s when Yan LeCun introduced the convolutional neural network (CNN) [15]. A CNN is an artificial neural network that has an input, an output, and multiple hidden layers, some of which are convolution – that is, they use mathematical models to pass data to successive layers [16]. Revolutionary at its time, It was the most primitive replication of the way humans process information. It laid the groundwork for the current advancements in deep learning which all build upon Yan's work [17]. Today there are multiple types of neural networks, implementations of which vary from very minimalist approaches to state-of-the-art scientifically published algorithms which take into account input size, speed, memory, efficiency, accuracy, etc., and can process large amounts of data quickly [18].

The input taken from computer vision algorithms is the image (frame) or video (consecutive frames) for which Classification, Detection, and Tracking are to be done. The output is the correctly detected instance/object [19]. In the following sections of this paper, we will explore the basics of neural networks, object detection, and current technologies relating to our work.

2.2 Neural Networks

Over the past few decades, the artificial intelligence systems with the best performance such as the speech recognizers used in smartphones or Google's contemporary automatic translator have come to fruition due to a technique called deep structured learning. Deep learning is the name given for an approach to synthetic intelligence called neural networks. The field has seen volatile fame over the past half of the century. Neural networks were originally proposed in 1944 by Warren McCullough and Walter Pitts. They were researchers at the University of Chicago who then moved to MIT in 1952 as founding participants of what is now known to be the first cognitive science department [20]. Neural networks are computing structures with interconnected nodes that work similarly to neurons in the human brain. Leveraging complex algorithms, they can recognize hidden styles, patterns, and correlations in random or labeled data, cluster, and classify them [21]. They attempt to emulate the way the human brain processes information. They interpret sensory facts via a perceptron that's defined later on in this document. They take random uncooked information as input and label or cluster it. They identify numerical patterns from this raw data and translate them into contained vectors, definite real-world measurable statistics whether the input is photos, audio, textual content, or time series, whichever must be translated [22]. A neural network consists of three important levels known as layers:

- Input Layer: In this layer, we feed all of the inputs to the neural network or model.
- Hidden Layers: These layers are the layers that are placed between the input and the output layers. A neural network could have more than one hidden layer.
- Output Layer: It is the Last layer of the Neural Network. It takes the records from hidden layers after information processing. Thus, the processed information is made available at the output layer [23].

Before we proceed further to understand how a neural network functions, it is necessary to have a basic understanding of deep learning and how it differs from artificial intelligence and machine learning. Deep learning can be summarized as the use of neural networks with single or multiple layers to solve complex problems. Deep learning as a whole comes under the umbrella of machine learning as its subset. These neural networks try and emulate the working of the human brain although the capability of the latter is far from the reach of even the most complex and advanced neural network. Even though a neural network with only one layer can make approximate predictions, we can introduce additional hidden layers to optimize and refine the accuracy of our model [24]. Machine learning comes under the umbrella of artificial intelligence as its subset. It is centered on enabling computer systems to research from facts and to enhance with experience – as opposed to being explicitly programmed to do so. In machine learning, algorithms are trained to identify and locate hidden patterns and correlations in huge datasets and to make the most accurate predictions, and come to suitable conclusions based on that deep analysis [25]. Applications that leverage machine learning improves with consistent use and become more accurate in their predictions and conclusions the more data they have access to [26]. At its core, artificial intelligence is a field that aims to employ computer science fundamentals on large datasets of random information to extract meaningful and tangible information from them. It additionally encompasses

sub-fields of machine learning and deep learning, these disciplines are comprised of AI algorithms and neural network models which seek to create systems that make predictions, classifications, or conclusions based on input information [27]. In conclusion, we now understand that artificial intelligence is the parent field of computer science which has machine learning as its sub-field or child which in turn has deep learning as its sub-field or child. For us to figure out how a neural network works, we need to dissect it down to its most fundamental unit, which is a perceptron [28]. A perceptron is a neural network with only one layer which is composed of four parts. These are the input values to be fed, the weights to be multiplied and bias, the net sum which is then multiplied result, and an activation function that holds some set of rules [29]. The perceptron first takes the inputs (i) which it obtains from the input layer, it then performs the arithmetic function of multiplication on the obtained input (i) with their respectively assigned weights (w). The improved values are then all summed up, the resultant value of which is the weighted sum. We then apply a suitable activation function on the weighted sum. The result received from the activation function is mapped values from the input to the respective output [30]. To explain it with an example: Let us assume we have one neuron, three inputs i1, i2, i3 multiplied by the respective weights w1, w2, w3 respectively, and that there is a function, inside our neuron, which will produce an output. This function may look like y = i1w1 + i2w2 + i3w3. This function is called the weighted sum. We can then pass this weighted sum into an activation function which will convert this input into a certain output based on a set of rules [31]. The following is a real world example of a perceptron:

Decision: Should I make biryani for dinner?
Possible inputs:

$i^1 = 1$	#	The ingredients necessary to make biryani are present.
$i^2 = 0$	#	I want to eat something other than biryani.
$i^3 = 1$	#	My friend really wants to eat biryani.

Possible Weights:

$w^1 = 3$	#	Since the ingredients are available, I'm willing to make biryani.
$w^2 = 4$	#	Biryani is my favorite food and I really want to eat it.
$w^3 = 2$	#	It is not my concern if my friend wants to eat biryani.

The weighted sum of the above values will determine the decision to make biryani or not for dinner. In this way, we can design our neural networks to solve a myriad of problems [32]**.** There are different kinds of deep neural networks designed to overcome different problems. Each has its advantages/disadvantages and use-cases. Common types of neural networks are feed-forward, radial basis functions, convolutional neural networks, multi-layer perceptron, recurrent neural networks, modular neural networks, etc [33].

2.3 Object Detection

Object detection is the fruit that has resulted due to years of progress in computer vision and photo processing. It is the technology that detects and defines objects like humans, man-made structures,

and automobiles in digital photos and videos [34]. It is extensively utilized in tasks leveraging computer vision technology which include but aren't limited to photo annotation, human facial detection, human facial recognition, activity recognition, video item co-segmentation [35] [36] [37]. Object detection is occasionally referred to as object recognition/identification. These terms are interchangeable and are equivalent in meaning and use. However, object detection is not the same as some other common technologies in computer vision including but not limited to classification (assigning one or more than one classes to a photo), key factor detection (identification of specific points in a photo), or semantic segmentation (separating the photo into specific areas through masks) [38]. We can use a variety of techniques to perform item detection depending on our requirements. Some of the popular object techniques are region-based convolutional networks also known as R-CNN's, you only look once also known as YOLO, and single-shot detector also known as SSD [39] [40] [41]. There are two approaches to object detection:

- The first approach is to create and train our own custom object detector. To train a custom object detector from zero, we first need to design an architecture for learning the various interest points for the items of interest. We also need to process a huge set of data that is labeled to train our custom detector. The outcome achieved by an object detector that has been trained on a custom dataset can be exceptional. It is also required to manually establish the weights and layers in the neural network which can take a large amount of time and training data [42].
- The second approach is to use a pre-trained object detector. The workflow depending on different individuals' requirements can be different. There may be either a lack of time or resources. Therefore it is sometimes much more convenient to make use of something known as transfer learning. In this, we start with a pre-trained neural network then fine-tune it according to the requirements of our program or system. Although this approach can yield quicker results since the item detectors have been trained on huge datasets of photos already but the achieved outcome will be very much average as compared to a custom trained object detector [43].

The pipeline of item recognition is most often simplified into the following processes known as photo classification, item localization, and item detection.

- Photo Classification: In this task, we anticipate the type or class of an item in a photo [44].
  - Input: The input for this task is a picture with one object, like a picture.
  - Outcome: The outcome of this task is a class label that is one or more integers that are mapped to class labels.
- Item Localization: In this task, we locate the presence of objects in a photo and indicate their location with a bounding box [45].
  - Input: The input for this task is a picture with one or more objects, like a picture.
  - Outcome: The outcome of this task is one or more bounding boxes that are defined by some extent, width, and height.
- Item Detection: In this task, we locate the presence of objects with a bounding box and types or classes of the located objects in a photo [46].
  - Input: The input for this task is a picture with one or more objects, like a picture.
  - Outcome: The outcome of this task is one or more bounding boxes that are defined by

some extent, width, and height, and a category label for every bounding box.

Now that we have a better understanding of the basics of object detection, we look at the various deep learning approaches and how they have evolved over the past decade.

- RCNN: One common procedure is to leverage a locale-based neural network or locales with attributes of convolutional neural networks. It is an advanced procedure that employs deep learning models to perform item detection [47]. The architecture employs the selective search process which returns around two thousand locale proposals. The proposed locales are then sent to the convolutional neural network which processes CNN attributes. The produced attributes are then sent in a support vector machine model to categorize the items present in the locale proposal. We also need to apply a bounding box regression to localize the items available in the photo more accurately [48]. One disadvantage of this approach is that it takes a massive amount of time to train the network since one would have to categorize over two thousand proposals per photo. It is difficult and not feasible to deploy in real-time as the time taken for each test photo is around fifty seconds [49].
- Fast RCNN: Another common procedure is to leverage the fast RCNN process. This process builds upon the existing architecture of RCNN by leveraging the use of spatial pyramid pooling/SPPnet to increase the overall speed of the RCNN model. Spatial pooling is employed to compute the convolutional neural network depiction for the complete photo a single time. We then use this depiction to compute the convolutional neural network depiction for every portion computed by the selective search process of RCNN. The procedure is taught from point to point [50].
- Faster RCNN: This approach is an improvement of the two previously mentioned procedures. Due to the constriction in the fast RCNN structure of locale proposal formation alongside selective search, this improved approach introduces its own locale proposal network which is much quicker in comparison to selective search. It enhances the locale proposal formation model during training. These enhancements in the procedure assist in minimizing the total time taken to perform item detection in comparison to previous approaches by a few seconds. When trained with standard datasets such as COCO and others, the mean average precision yielded is approximately eighty percent, which is around ten percent more than fast RCNN. The introduction of the locale proposal network within the architecture added more towards the overall mean average precision as compared to selective search [51].
- YOLO: This is a moderately new and innovative approach as compared to previous methods. The acronym stands for you only look once. The process of item detection in this procedure is tackled using regression. After that is done, it offers the categorical likelihood of the traced photos. This algorithm leverages convolutional neural networks to trace items in real-time. Furthermore, it just needs to move forward within the neural network once to trace items. It has very high speed and accuracy as compared to previous procedures and its learning capabilities are also very diverse. Other technologies it leverages are residual blocks, bounding box regression, and IoU. [52].
- SSD: This approach is one of the most versatile methods and also the one used in this project. The acronym stands for single-shot detector. Its name is derived from the fact that

the method assimilates to chart categorization and regression issues straight from untouched photo pixels to bounding box key points and categorical likelihood, in unary overall moves [53]. It is among the most common item detection processes used because of its simplicity in execution, relatively acceptable accuracy, and low computational resources required. For this reason, it is also preferred to be used on machines with low resources or on embedded devices [54]. This method is entirely a convolutional neural network that can be broken down into three main segments. Foundational convolutions extracted from a prevailing photo categorization architecture which offers lower-level attribute charts, Supplementary convolutions introduced besides the foundational network which offers higher-level attribute charts, Projection convolutions which will recognize and find items within the attribute charts [55].

2.4 Current Technology

Although the industry remains at the bleeding edge of computer vision technology, there hasn't been an automated prevention solution available for the pandemic ever since last year. There are solutions with somewhat similar functionality as our application available however our system improves on some key industry-wide practices which have resulted in no widespread adoption until now [56]. Most of the available solutions are extremely expensive and ones that are relatively accessible for first-world countries have inconveniences such as forced ecosystem inclusion, proprietary software, and additional fees for updates or maintenance which make them inconvenient for institutions to deploy [57]. Most solutions do not even have social distance detection. After analyzing the features of available applications in the market, we decided to equip our system with better features while lowering costs, lowering barriers for adoption, and lowering maintenance. The core features are compared in table 9. The following are some of the solutions similar to ours:

2.4.1 HikVision

HikVision is an IoT Solutions provider which offers face mask detection cameras. The cost of their products is around 250 to 400 USD which is somewhat reasonable. They do not offer social distance detection. The company requires quarterly contracts for hiring maintenance teams which is very expensive on a large scale. The software and hardware are proprietary so if an issue arises, the user is stuck with the expensive product [58] [59] [60].

2.4.2 Rhombus

Rhombus is a Security Solutions Provider which offers cameras with the ability to detect Face masks and also social distancing. The cost of their products is around 300 to 1200 USD, which is very high. The company does not require any quarterly or yearly contracts for hiring maintenance teams but payment is only required when the maintenance team is called by the user. The software and hardware are also proprietary so if an issue arises, the user is stuck with the extremely expensive product [61] [62] [63].

2.4.3 Tyco

Tyco is an AI Solutions Provider which offers cameras and software solutions to detect Face masks. The cost of their products is around 250 to 1000 USD, which is very high. The company requires yearly contracts for hiring maintenance teams. They do not offer social distance detection. The hardware can only be bought in large batches of 100+ items. The software and hardware are also proprietary so if an issue arises, the user is stuck with the extremely expensive product [64] [65] [66].

2.4.4 FindFace

FindFace provides facial recognition solutions to governments and institutions. The company recommends users' cameras from partner companies that can be used together with their software. They do offer social distance detection. The cost of their services is very high. The company requires yearly contracts for use of the software. The software and hardware are also proprietary so if an issue arises, the user is stuck with the extremely expensive product [67] [68] [69].

2.4.5 Invixium

Invixium is a Public Safety Solutions provider which offers a multitude of hardware and software solutions. The company offers face mask detection access control solutions which cost around 300 to 1000 USD which is expensive. They do not offer social distance detection. The company requires quarterly contracts for hiring maintenance teams which is very expensive on a large scale. The software and hardware are proprietary so if an issue arises, the user is stuck with the expensive product [70] [71] [72].

Our proposed system provides features in two modules. The first module provides users with a face mask detection facility. The second module provides a social distance detection facility for users to detect crowded areas. Users can then contact authorities and notify breaches of SOPs.

Chapter 3

Requirement Specifications

3.1 Existing System

There is a multitude of companies that have worked throughout the pandemic to offer solutions for the spread of infectious diseases with the help of technology but none of them have been put to use anywhere by the governments of the world. The following are the reasons why:

1. Cost: The base price for all solutions is very expensive, ranging from 200 to 1500 USD. When the system is scaled these costs become inconceivably large.
2. Build: The hardware often takes up a lot of space since most cameras are held on a stand at approximately a height level.
3. Risk: Since all devices use expensive cameras with expensive computers such as iPad's or laptops the risk of theft in most areas is high.
4. Additional Fees: Companies charge recurrent fees for maintenance and access to software which can add up quickly.
5. Contracts: Since most companies operate at an enterprise level, conducting operations business to business, they enforce yearly or quarterly contracts to make it inconvenient for the opposing parties to leave in case of disagreement or dissatisfaction.

Lack of Features: Almost all solutions only offer face mask detection.

3.2 Proposed System

Our proposed system attempts to simplify and diminish the drawbacks of the existing system. This is intended to be accomplished by taking advantage of existing technologies and systems instead of trying to reinvent the wheel and in turn keeping costs low, keeping the physical footprint small, making the whole system cheap, making the system easy to be scaled and easy to maintain. The system will have relatively sufficient efficiency and is designed to be quickly pushed to embedded systems or legacy systems and be used in hundreds of thousands of places at once allowing governments to grab some sort of control of a turbulent unforeseen situation and breathe some sort of stability in society.

Our proposed system is divided into two modules. The first module aims to perform face mask detection. The second module aims to provide social distance detection. User interaction required is minimum since the security staff simply has to run the software and provide input either through the camera or through images/videos. The regions of interest will be extracted from the provided Input and will be classified using Deep Learning Neural Network Models and output the spatial location of the objects/instances. The input will be classified on "Yes" if Face Mask is worn or "No" if it isn't worn AND "Yes" if a social distance of approximately six feet is kept or "No" if the social distance isn't kept.

The following Figure 1 shows the approach used for detecting face mask and social distance:

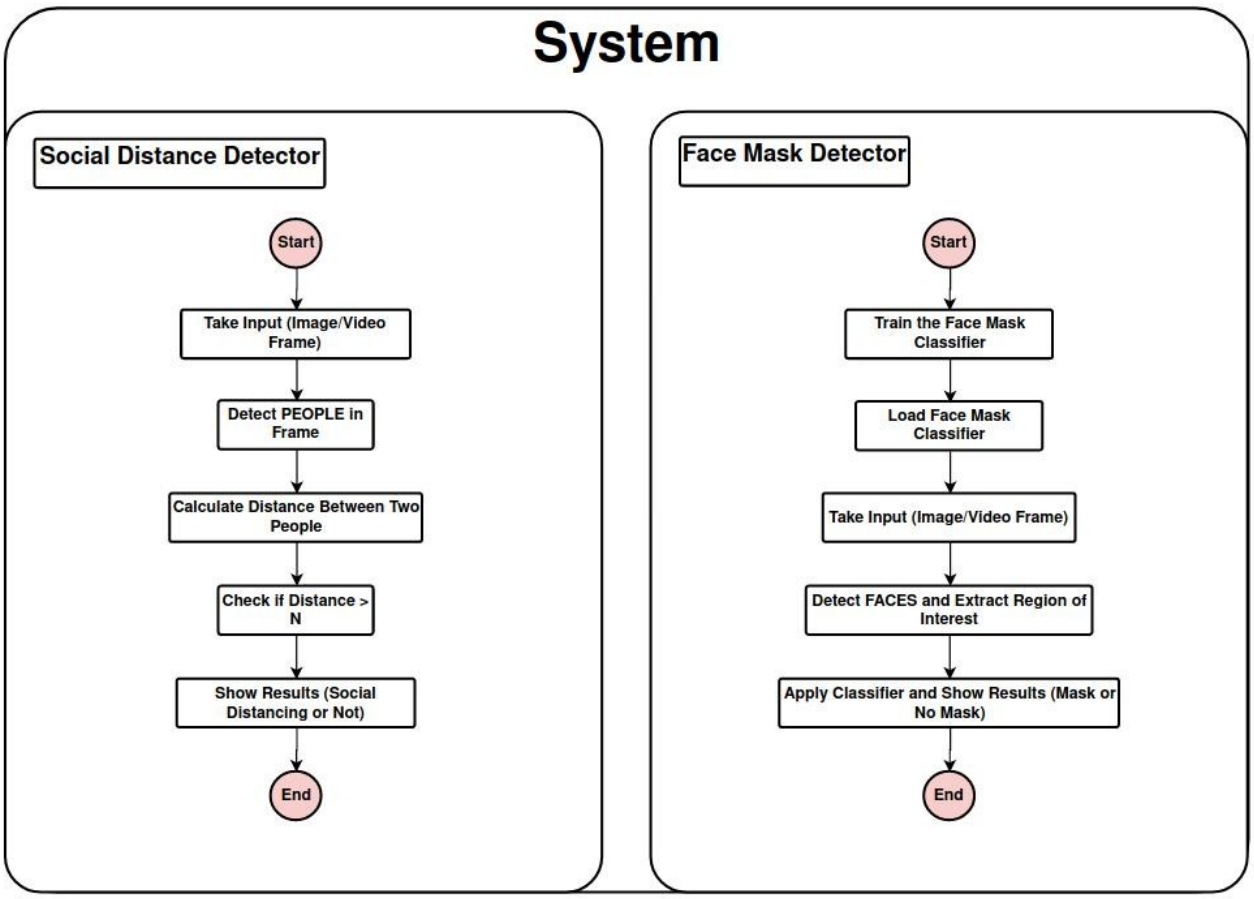


Figure 1: General Flow of Proposed System

As Figure 1 shows, we aim to develop an efficient, accurate, and reliable detection system that will reduce the work of authorities and governments by automating the detection processes. The proposed automated system is divided into two separate modules for simplicity and ease of maintenance or upgrade.

3.3 Functional Requirements

- The system must take image, video or real-time webcam stream as input.
- The system must be able to detect multiple faces and determine whether they are wearing a face mask or not.
- The system must be able to detect people and determine whether approximate social distance is being maintained or not.
- The system must give image, video or real-time webcam stream as with detected face masks or social distance between people as output.
- The system must accept file upload as input for image and video.
- The system must display invariance to light under normal daytime conditions.
- The system must have an accessible graphical user interface to interact with modules.

3.4 Non-Functional Requirements

- The system shall achieve 99% up-time and 50 hours mean time before failure.
- The system shall leave approximately 20% of available resources available unused.
- The system shall finish restart cycle in under 120 seconds.
- The system shall be easy to navigate, understandable and accessible to adult members aged 18 to 60 with minimum to no training.
- The system shall not require maintenance shutdowns and critical failures must be resolved by restarting the system.
- The system shall allow a developer with at least one year of experience supporting similar software applications to design, implement, modify and test new or old product features.

3.5 Limitations

Although we may specify certain constricting specifications in our problem domain, It may not be possible to strictly adhere to these conditions when implementing the said system in the real world. The Social Distance module will measure the approximate distance between two people and might have some discrepancy under certain conditions.

3.6 Use Cases

The proposed system takes advantage of deep learning and AI to target the public health and safety industry. The system can be slightly altered to fit more than one solution application area. For example, The system could be altered to work with a door. If a person is wearing a face mask the door will be unlocked. If a person is not wearing a Face mask the door will remain locked. This can be used at executive meetings, conferences, gatherings, etc. to only allow people who are following standard operating procedures inside. It could also be used at restaurants and café kitchens even after the epidemic ends to maintain hygiene. The following figures show some relevant use case diagrams:

3.6.1 Use Case 1: SD & FM Detection System Use Case Diagram

The use case diagram shown in Figure 2 shows the overall features of our SD & FM Detection System.

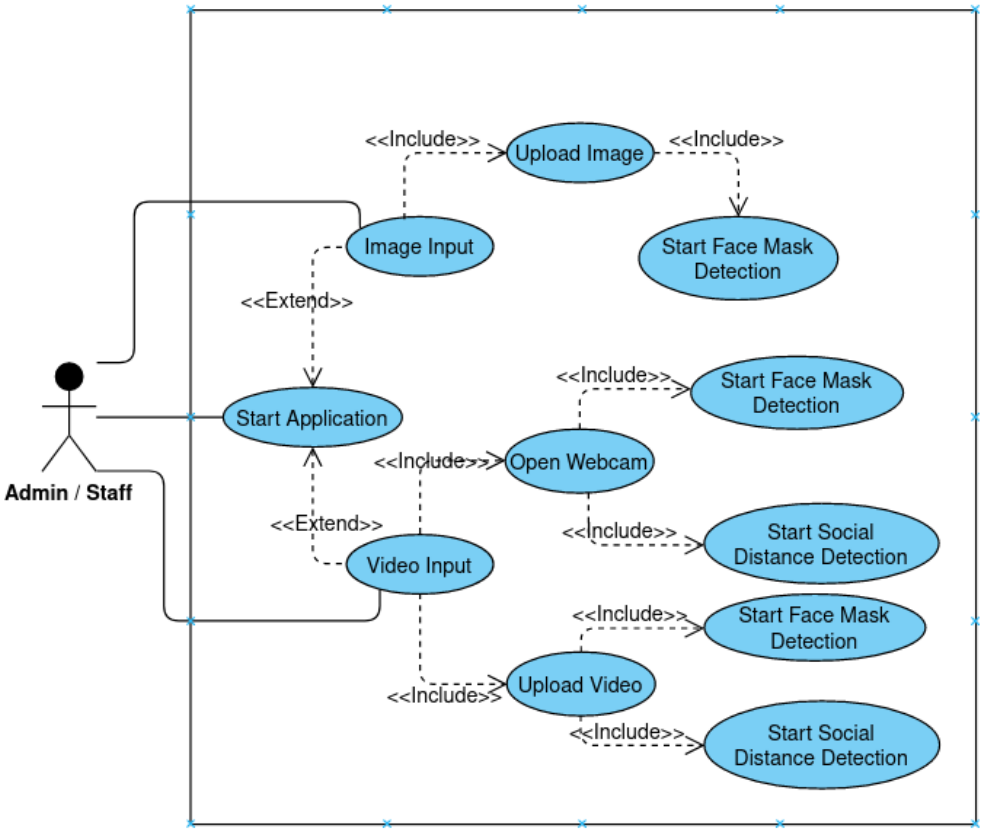


Figure 2: Use Case Diagram of SD & FM Detection System

The following Table 1 shows the various conditions involved, actors, and their roles in interacting with the SD & FM Detection System.

Table 1: SD & FM Detection System Features

Title	SD & FM Detection System
Actors	Admin/Staff
Roles	Admin/Staff interacts with the features of the SD & FM Detection System.
Description	This use case consists of actor, multiple direct activities (Start, Image Input, Video Input) and multiple use cases.
Pre-Condition	SD & FM Detection System dependencies installed.
Post-Condition	SD & FM Detection System allow Admin/Staff to use its features.

3.6.2 Use Case 2: SD & FM Detection System Use Case Diagram

The use case diagram shown in Figure 3 shows the Face Mask Detection feature of our SD & FM Detection System.

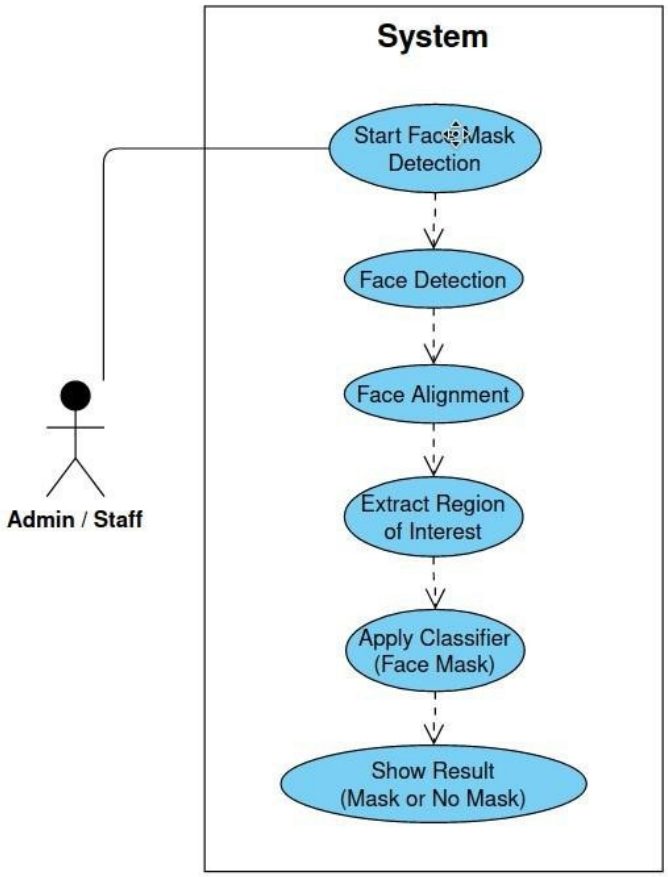


Figure 3: Use Case Diagram of SD & FM Detection System

The following Table 2 shows the various conditions involved, actors, and their roles in interacting with the SD & FM Detection System.

Table 2: Face Mask Detection Feature

Title	Face Mask Detection Feature
Actors	Admin/Staff
Roles	Admin/Staff interacts with the Face Mask Detection feature of the SD & FM Detection System.
Description	This use case consists of actor, one direct activity (Start) and multiple use cases.
Pre-Condition	SD & FM Detection System dependencies installed.
Post-Condition	SD & FM Detection System allow Admin/Staff to use its features.

3.6.3 Use Case 3: SD & FM Detection System Use Case Diagram

The use case diagram shown in Figure 4 shows the Social Distance Detection feature of our SD & FM Detection System.

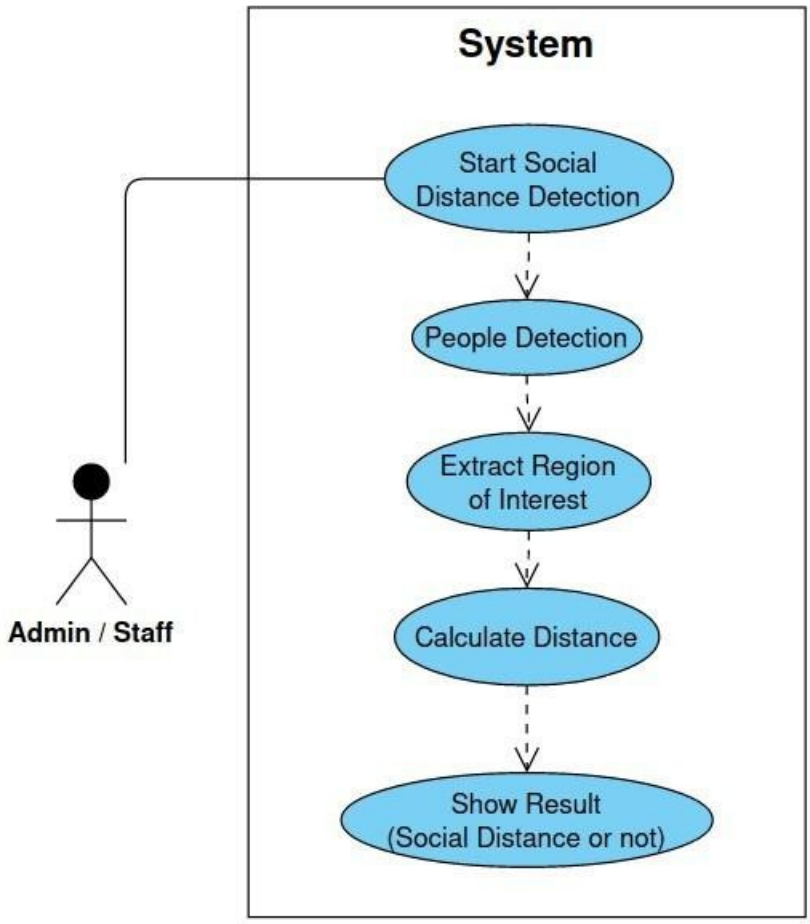


Figure 4: Use Case Diagram of SD & FM Detection System

The following Table 3 shows the various conditions involved, actors, and their roles in interacting with the SD & FM Detection System.

Table 3: Social Distance Detection Feature

Title	Social Distance Detection Feature
Actors	Admin/Staff
Roles	Admin/Staff interacts with the Social Distance Detection feature of the SD & FM Detection System.
Description	This use case consists of actor, one direct activity (Start) and multiple use cases.
Pre-Condition	SD & FM Detection System dependencies installed.
Post-Condition	SD & FM Detection System allow Admin/Staff to use its features.

Chapter 4

Design

4.1 System Architecture

The architecture of the System is primitive by design to enable compatibility on legacy systems. The System is divided into two parts, the Face Mask Detector and the Social Distance Detector, for simplicity and ease of update/maintenance. In SD & FM Detection System users will be able to select image or video as input and choose whether to perform Social distance detection or face mask detection. Our deep learning model will be trained on a "With Mask/Without Mask" Open Source Dataset. The system will have a Presentation layer (A Graphical User Interface with which the user can interact with the software), an Application Layer (Where all our core logic is defined) but no Data Layer since it is illegal to take an individuals image or video without their consent. Our system will therefore not store any data and thus will not require a database. The data layer in Figure 5 is arbitrary and only to get a quick high-level view of the system. Data is pulled from the physical storage onboard the device on which the software is run. By discarding the database our system can be ported to mobiles, embedded systems, and legacy machines by dockerizing. Figure 5 shows the detailed architecture of the SD & FM Detection System.

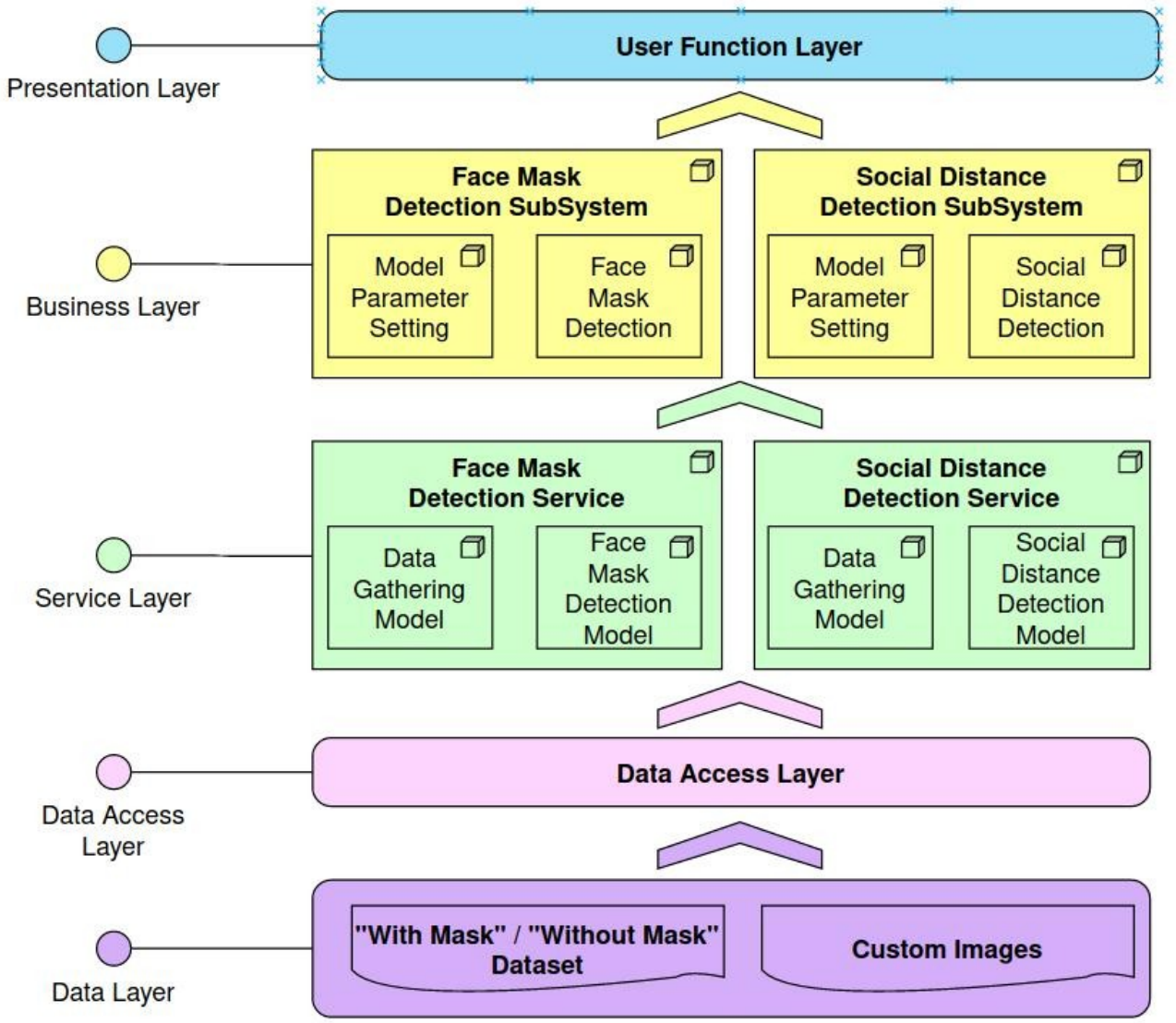


Figure 5: System Architecture of SD & FM Detection System

4.2 Design Constraints

4.2.1 Software and Hardware Requirements

- Computer with Operating System (Windows/Linux)
- JetBrains PyCharm Professional 2021
- Webcam/Camera
- Reasonable Storage (approx. 100 MB+)
- Reasonable Processing (1/2 GB Ram)

4.2.2 Data Requirements

- With Face Mask and Without Face Mask Dataset

4.2.3 Development Environment Requirements

- JetBrains PyCharm Professional 2021
- Qt Designer
- Qt Creator

4.2.4 Programming Language

- Python
- Shell Script

4.3 Design Methodology

An overview of the proposed system is presented in Figure 5. The design methodology for the system is the waterfall method shown in the following Figure 6.

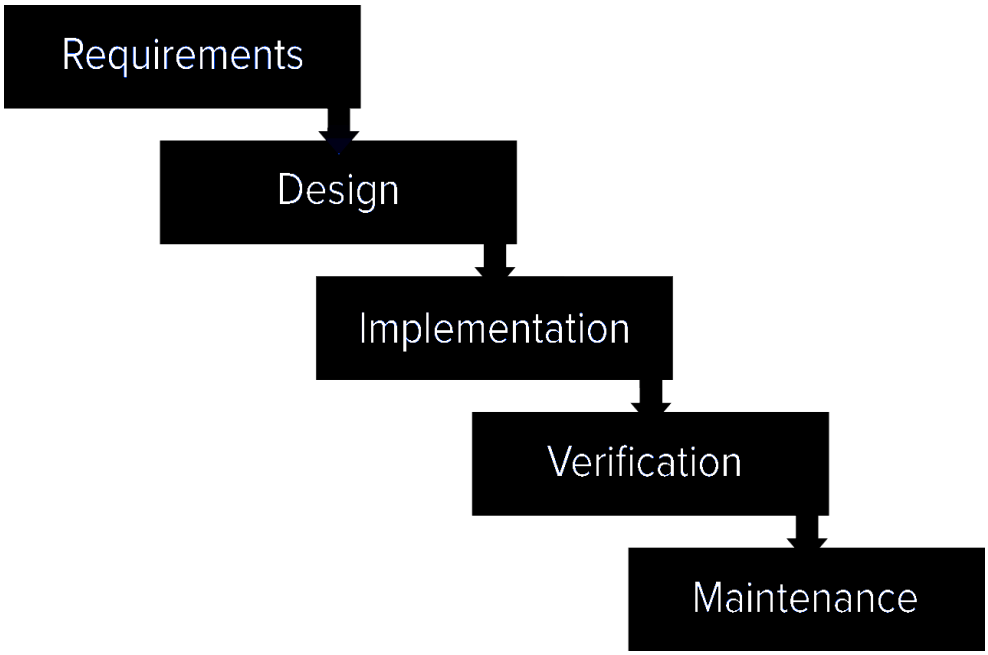


Figure 6: Selected Design Methodology (Waterfall)

4.4 High-Level Design

Figure 7 shows a high-level logical view of the system.

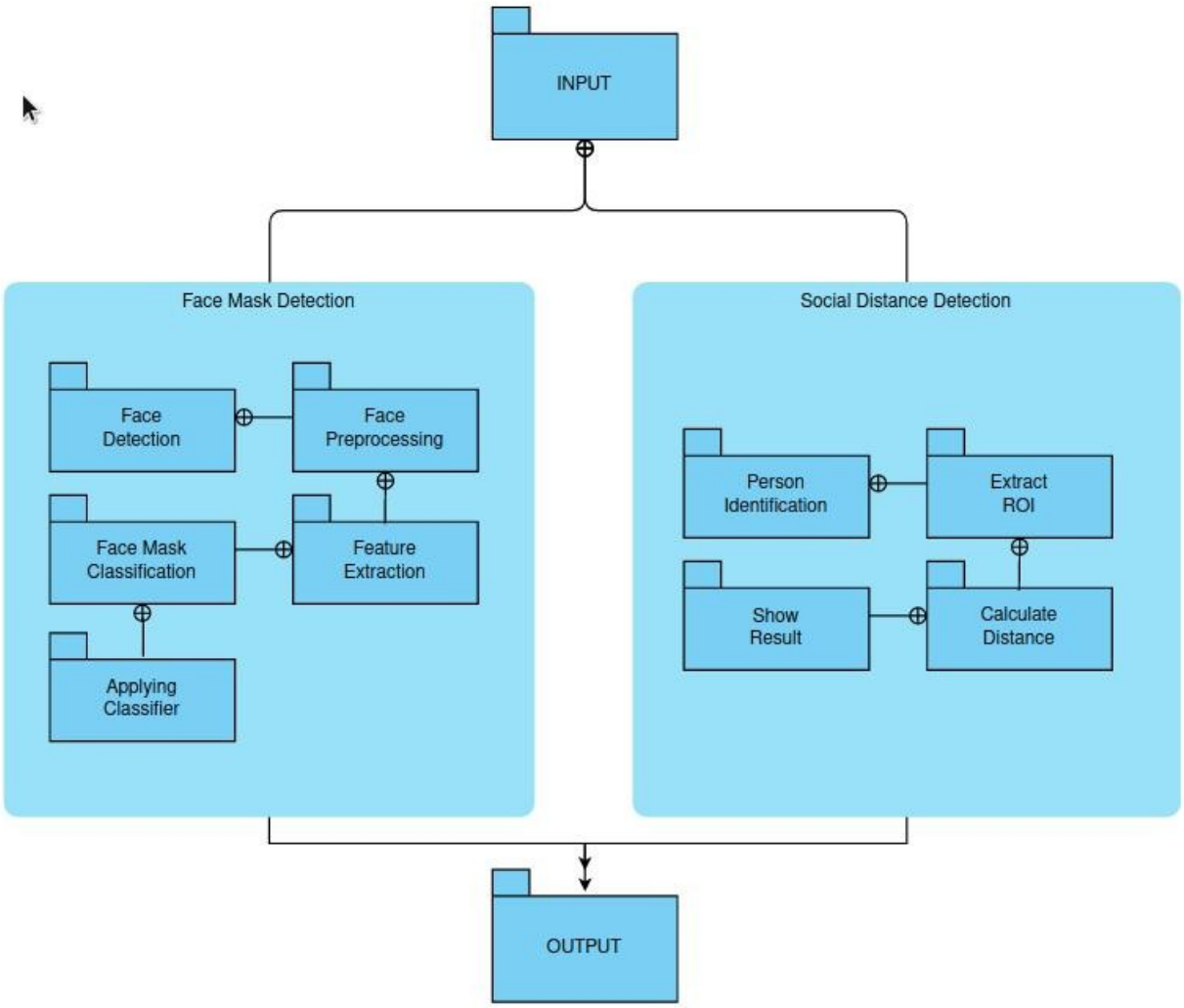


Figure 7: Package Diagram for SD & FM Detection System

4.5 Sequence Diagram

Figure 8 shows a sequence view of the system.

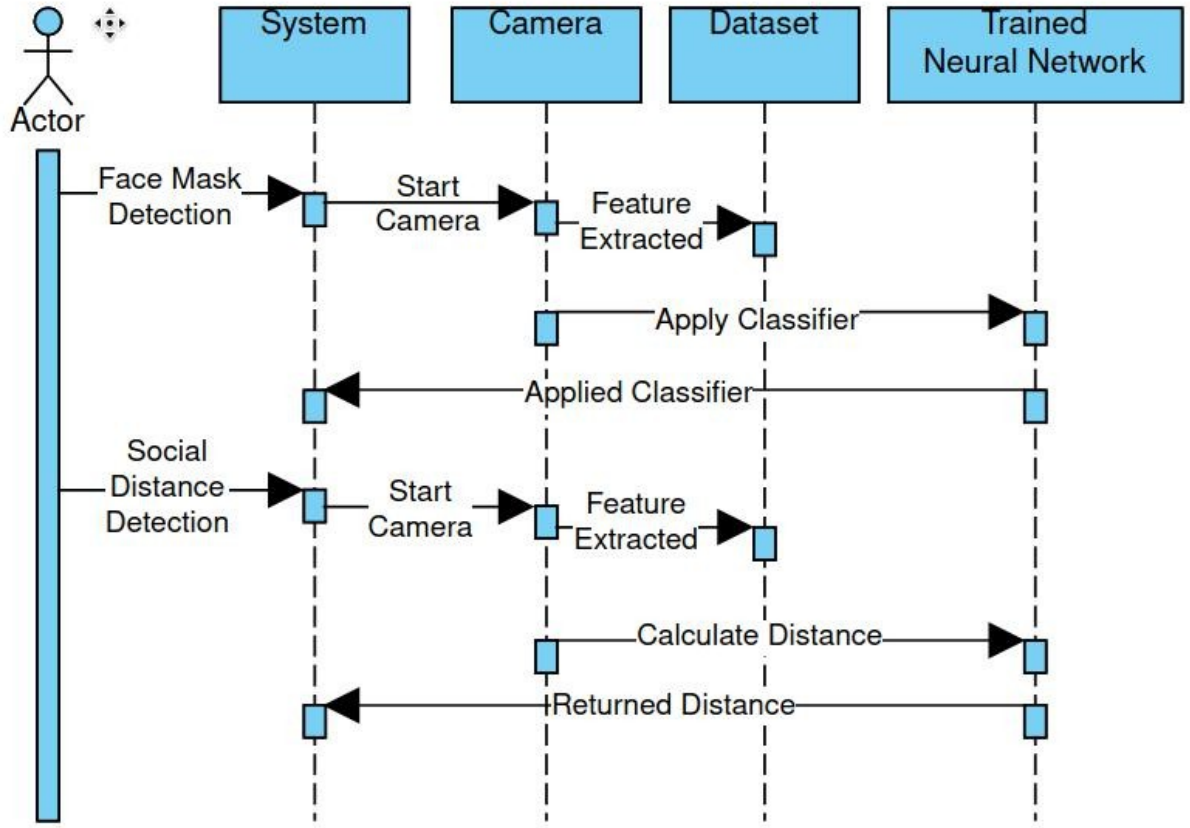


Figure 8: Basic Sequence Diagram for SD & FM Detection System

4.6 Low-Level Design

Since we are using some artificial intelligence frameworks such as TensorFlow which have more than hundreds of classes, it is not feasible to draw them. Refer to Figure 1 for Low-Level Design.

Chapter 5

System Implementation

5.1 System Summary

The SD&FM Detection System is designed for detecting whether or not an individual is wearing a face mask and whether or not a group of individuals is maintaining a social distance of approximately six feet. The system has a graphical user interface for interacting with the system's various modules. The user selects the type of detection they wish to perform, the system performs the detections, and gives back the output. The user has to perform a minimal amount of work for the SD&FM Detection System to work.

5.2 Application Interface

The GUI of the SD&FM Detection System is very simple and user-friendly. The SD&FM Detection System will be widely used by people such as security personnel and information technology technicians to ensure adherence to standard operating procedures. The system interface consists of a simple selector through which the user can interact with the various modules of the system. The user simply needs to choose the type of detection they wish to run and provide input either via image, video, or real-time webcam stream.

5.2.1 User Access Level

Users can access the system by simply double-clicking on the executable shell script in the main folder. Since the GUI is written in shell, it can run on all major platforms such as Windows, Mac, and Linux. The user does not have to make any sort of account or log in to access the system. The only requirement is to have a copy of the SD&FM Detection System's folder on their system's hard drive. The user can then easily navigate to it, and run the executable shell script. Figure 9 shows the folder layout of the SD&FM Detection System. The system is divided into two main modules for simplicity, the FaceMaskDetector module, and the SocialDistanceDetector module. Each folder contains relevant .py files, machine learning models, and test files. The system is run by double-clicking on the app.sh shell script.

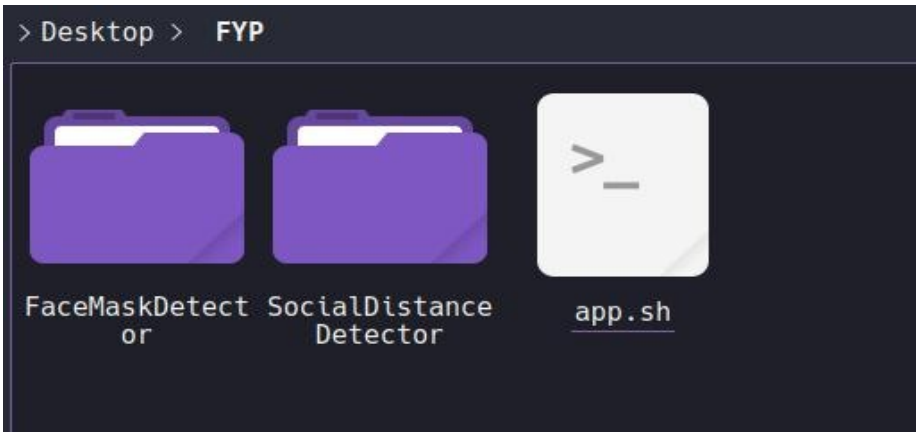


Figure 9: SD&FM Detection System Folder Layout

5.2.2 Understanding the main menu

The main menu of the application will have five options for the user to choose from as can be seen in Figure 10. The options listed are the types of detections that the system offers. These are:

- Face Mask Detector On Image.
- Face Mask Detector On Video.
- Face Mask Detector On Webcam.
- Social Distance Detector On Video.
- Social Distance Detector On Webcam.

The user has to choose one by selecting it then clicking okay.

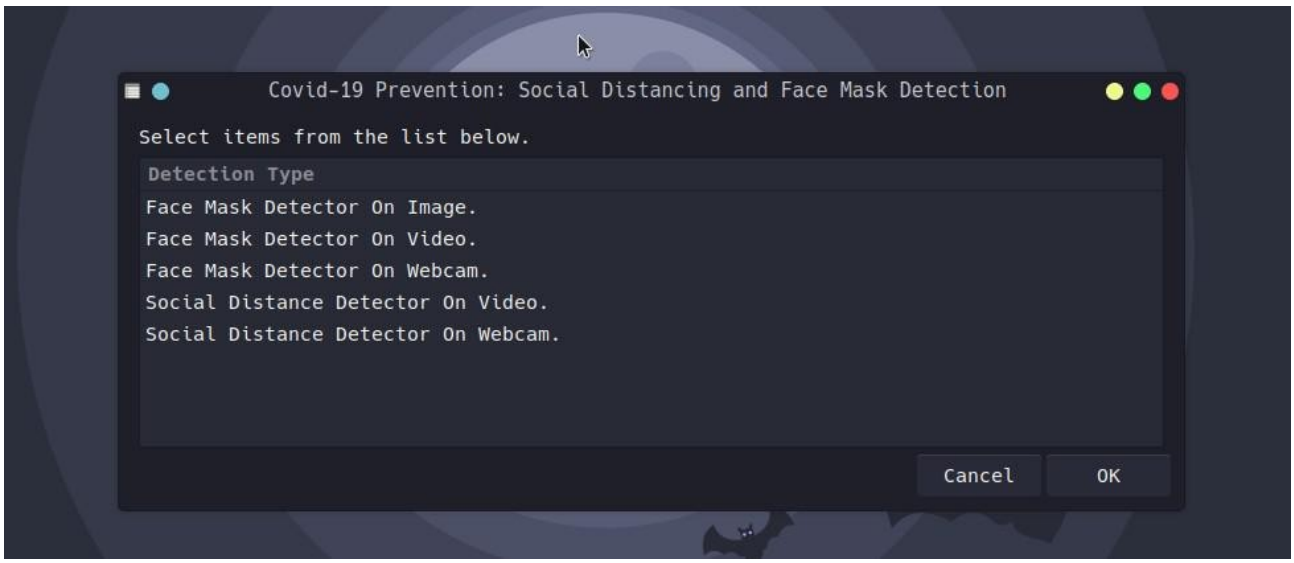


Figure 10: SD&FM Detection System Main Menu

5.3 Face Mask Detector On Image

After selecting the option and clicking okay, a file selection dialog box is shown through which the user can upload a file to detect face masks. The file selection dialog box is shown in Figure 11.

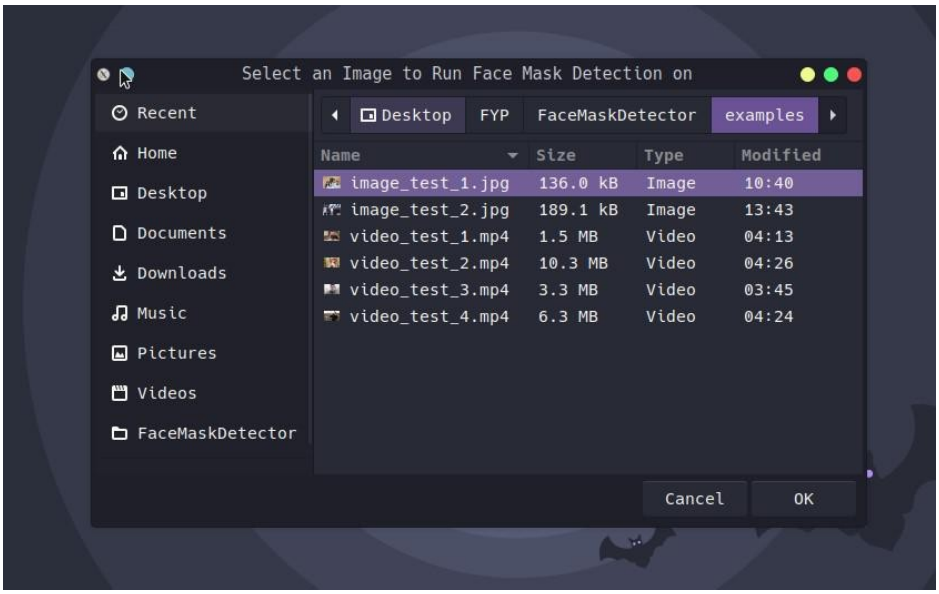


Figure 11: After a selection is made the user is shown a dialog box to upload an image/video.

After the user makes a selection and clicks okay, the image/video they chose is sent to our python detection program as input. After a few seconds, the model outputs the result with detected facemasks on faces. Example outputs are shown in Figures 12 and 13.

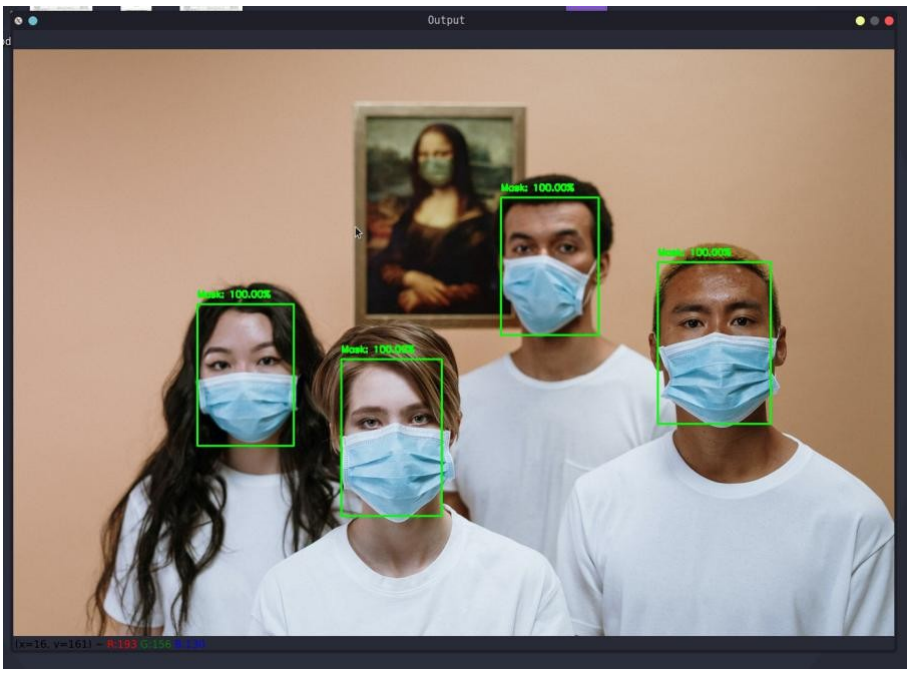


Figure 12: Face Mask Detection On Image Test Output 1

Figure 13: Face Mask Detection On Image Test Output 2

After the system shows the output to the user, a new information box is shown which contains all relevant information for the execution of the program. This information is passed from the Terminal to the dialog box so the user can easily tell if the program worked correctly or if they received an error. In case of error, they can copy the text and send it to the developer with ease. The information box resulting from running the detector in Figure 14 is shown below.

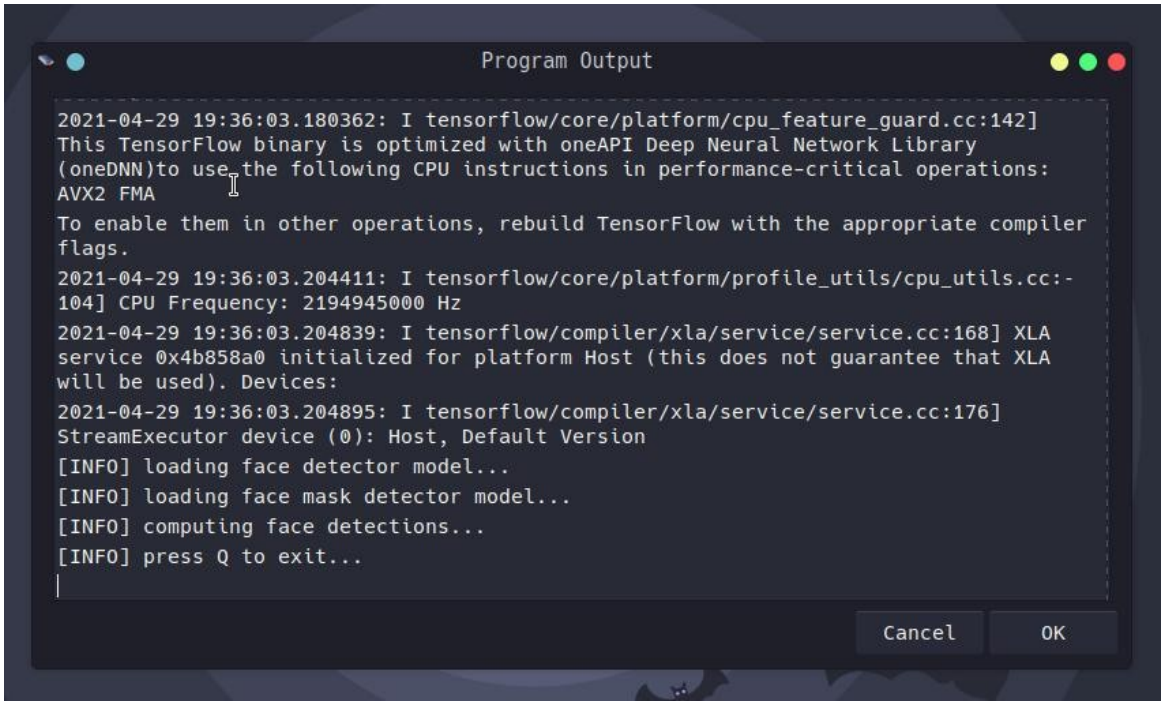


Figure 14: Program output after running face mask detection on image test output 2

After the system shows the program output and the information dialog to the user, the program terminates. The user is then free to rerun the program to perform another detection type.

5.4 Face Mask Detector On Video

The SD&FM Detection System also gives the user the choice to run detection on a saved video. The user needs to simply select the 'Face Mask Detector On Video.' option in the menu. Provide an input video file. And that's it. The rest of the heavy lifting is done by the system. Figure 15 shows an example output from a face mask detected in a video.

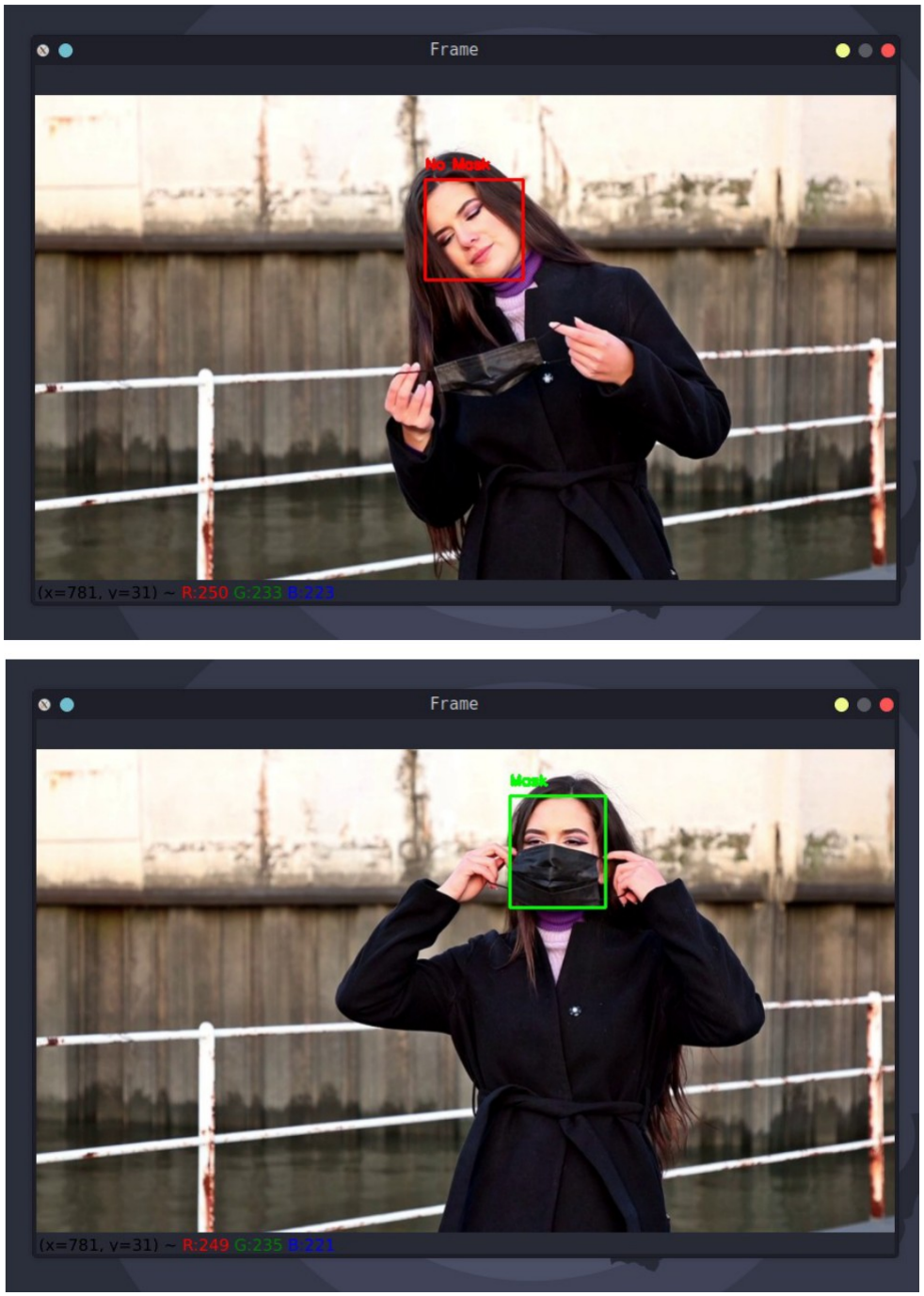


Figure 15: Shows Frame number 50 and 100 of our video file.

Detection on video is done in real-time where each frame of the video with detected bounding boxes is shown to the user frame by frame. After the video ends or the user presses 'Q' to force exit the program, the detection stops and the program output is shown. After which the program terminates.

5.5 Face Mask Detector On Webcam

The user can also run face mask detection on real-time webcam streams. This application greatly enhances the feature set of our system since it can have many solution application areas such as being used as an entry point to closed spaces such as offices or spaces which require a higher standard of hygiene such as kitchens. The user needs to simply select the 'Face Mask Detector on Webcam' option and click okay. The system will turn on the primary camera of the device and start performing detections on each of its frames as shown in Figure 16.

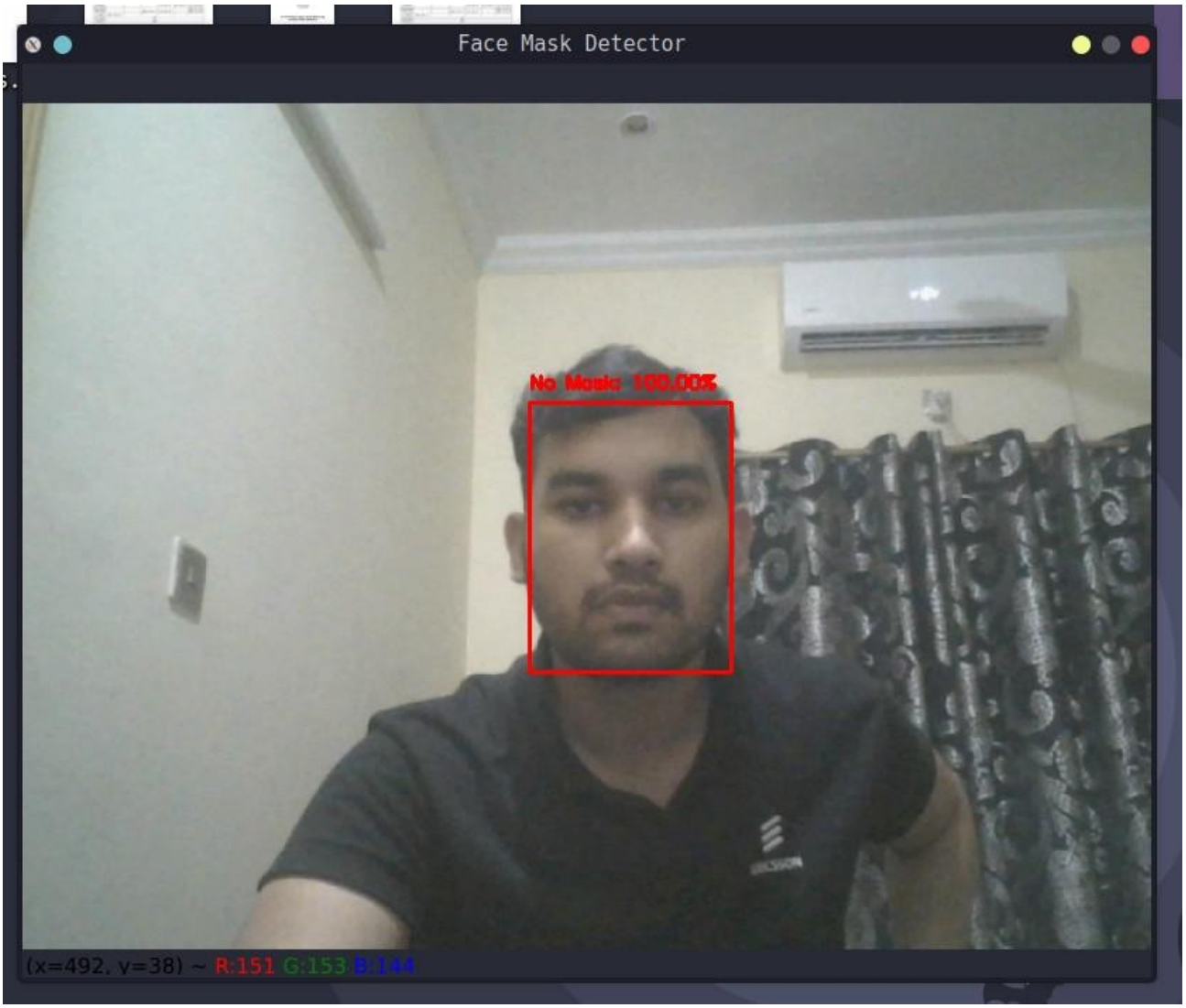


Figure 16: Face Mask Detector on Webcam.

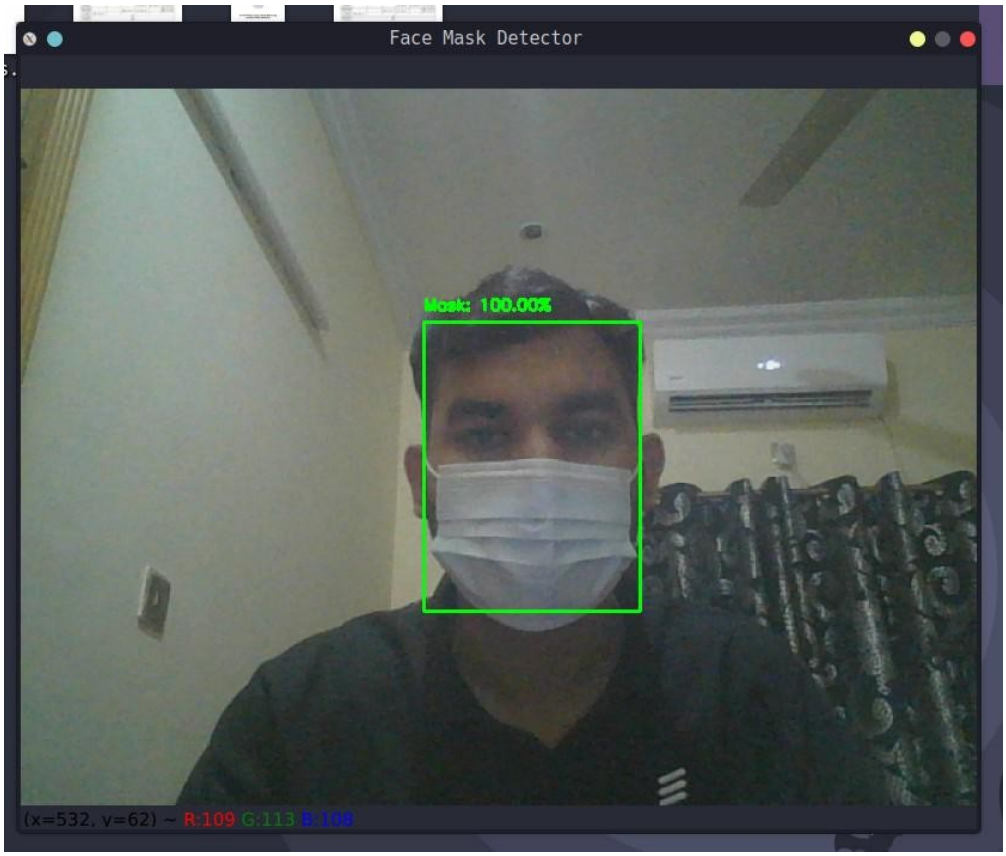


Figure 17: Face Mask is accurately classified.

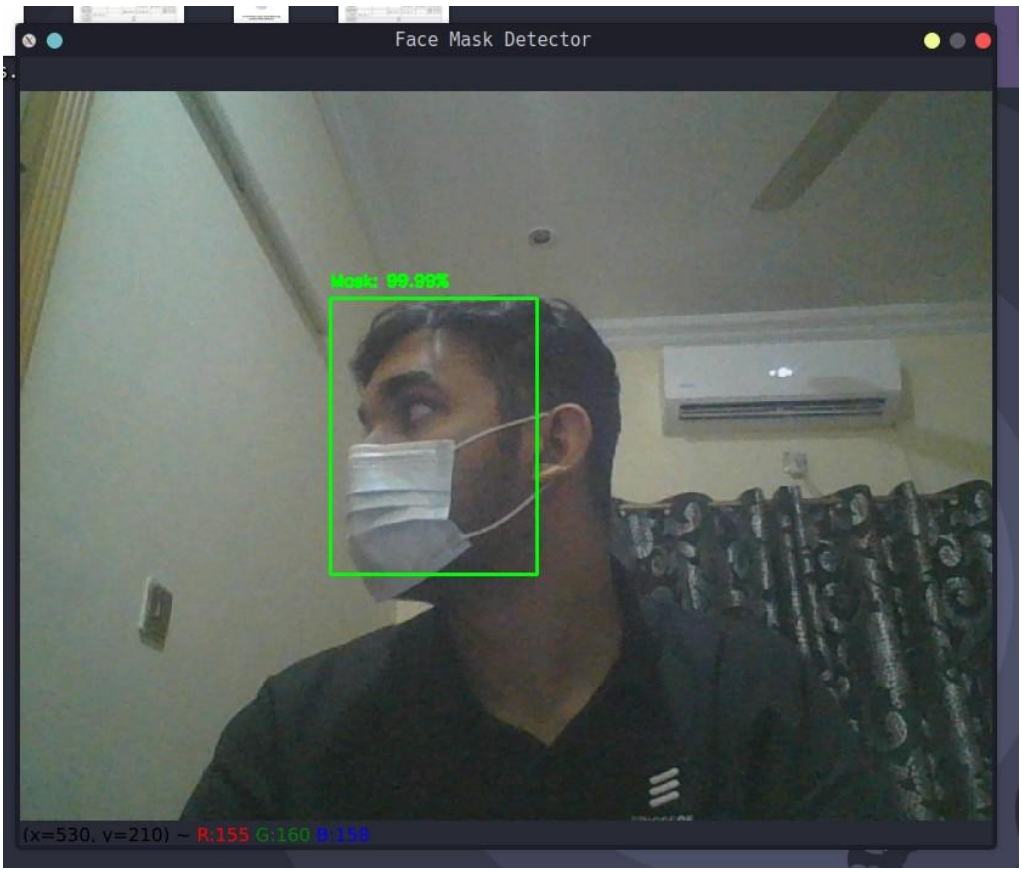


Figure 18: Side Views are accurately classified.

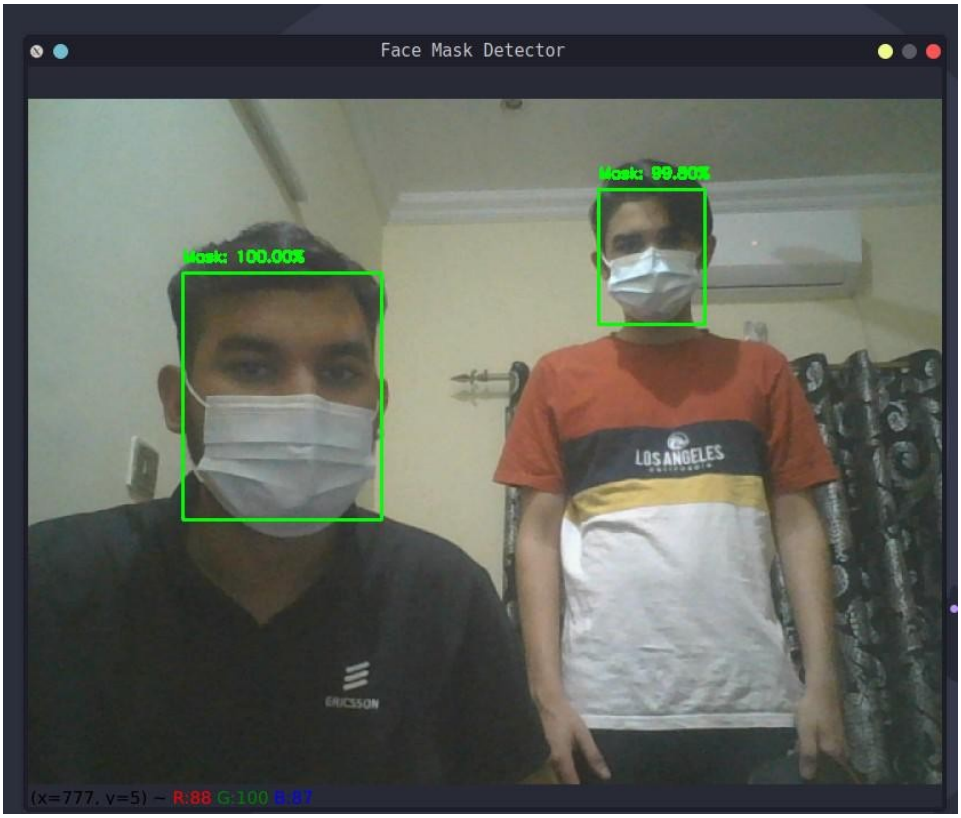


Figure 19: Face Mask on multiple people.

Figure 20: Side Views of multiple people.

Since a person's face won't always be facing straight at the camera in real-life scenarios, the SD&FM Detection System can accurately classify faces when they are looking in other directions, as can be seen in the previous figures. After the webcam detection ends, the program output is shown where the user can see if there was any error, after which the program terminates.

5.6 Social Distance Detector on Video

The SD&FM Detection system allows the user to check whether people are maintaining social distance in saved video files. The system uses triangle similarity to identify the distance or depth of a person from the camera and determine whether they are maintaining approximately 6 feet distance as can be seen in Figure 21.

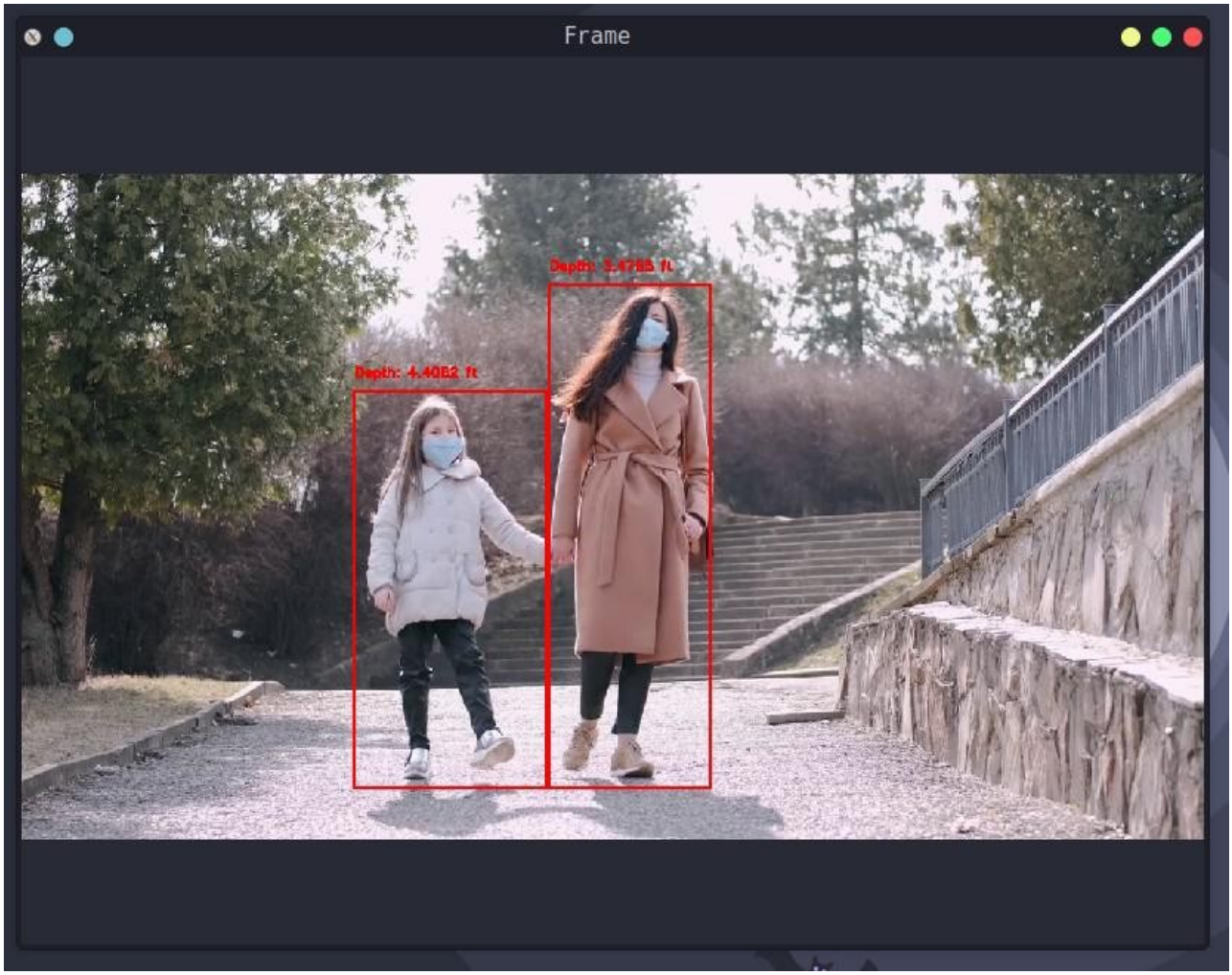


Figure 21: Social Distance Detection on Video Test Output 1

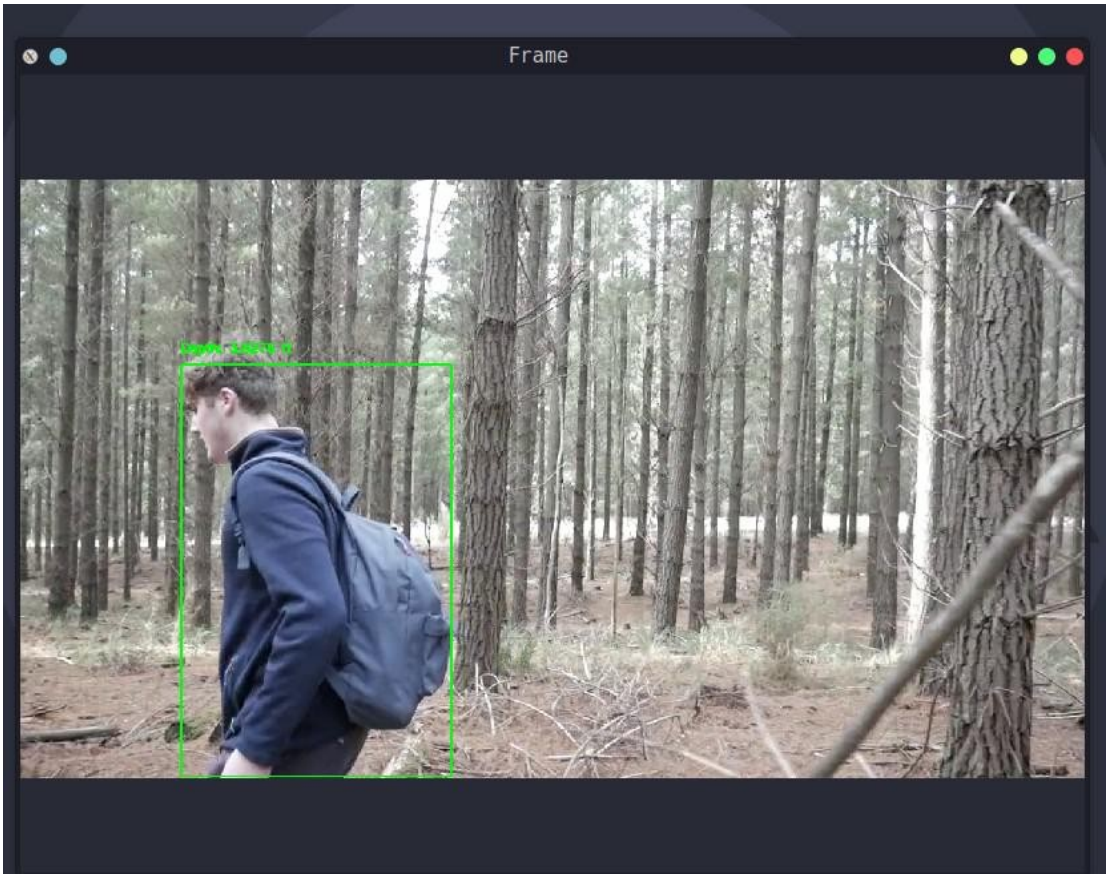


Figure 22: Social Distance Detection on Video Test Output

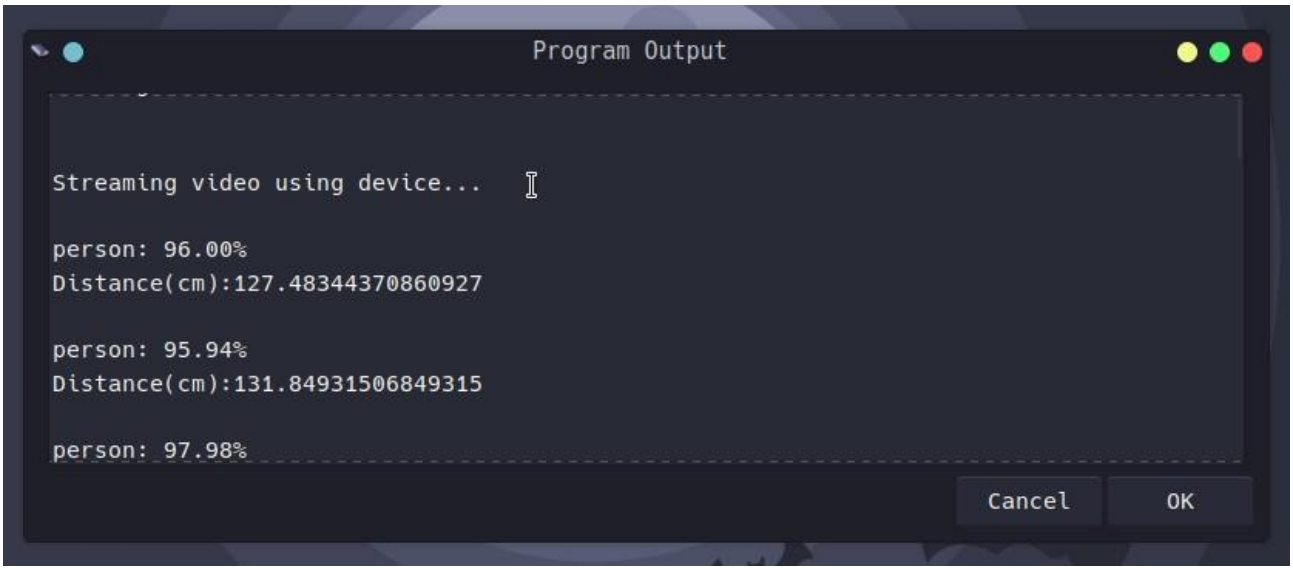


Figure 23: Program Output for Test Output 2

The Program output for the social distance detector shows the probability score of detecting people in each frame and their approximate distance from the camera.

5.7 Social Distance Detector on Webcam

The SD&FM Detection system allows the user to check whether people are maintaining social distance in a live webcam real-time stream. The system uses triangle similarity to identify the distance or depth of a person from the camera and determine whether they are maintaining approximately 6 feet distance. The 'Social Distance Detector on Video' module can be seen in Figure 24.

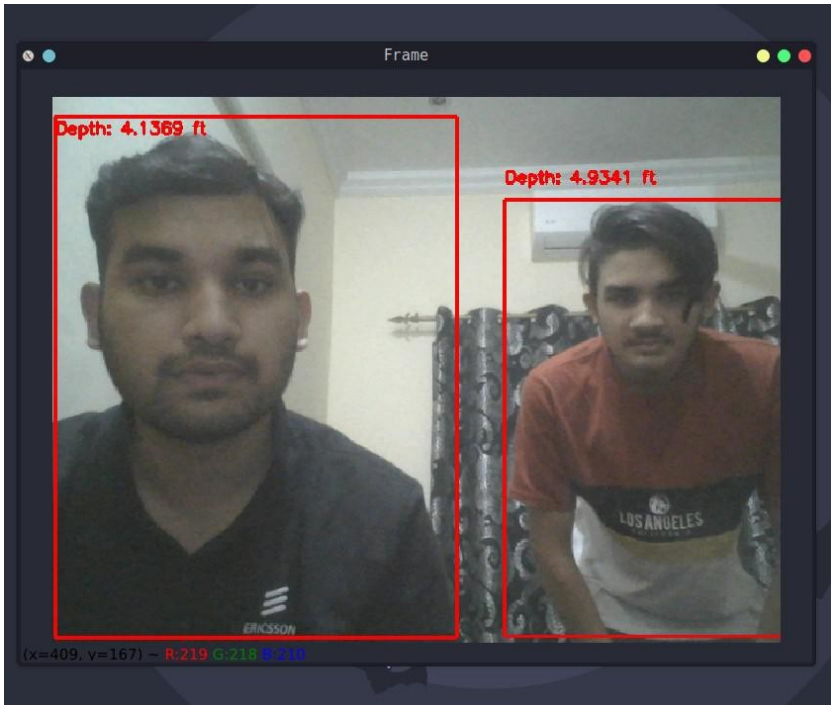


Figure 24: Social Distance Detection on Webcam (Too close)

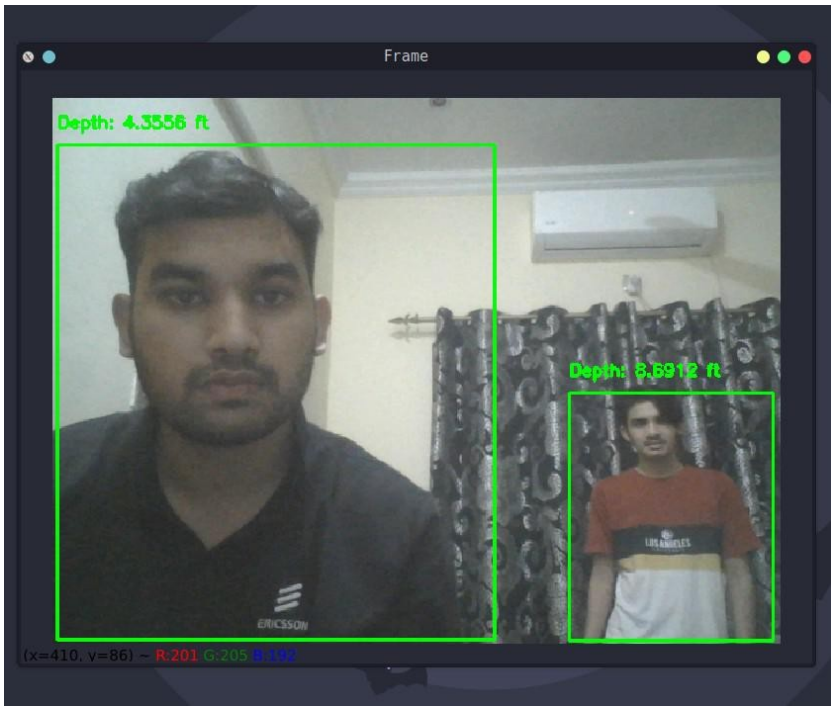


Figure 25: Social Distance Detection on Webcam (Maintaining Distance)

After detection ends the user is shown the program output for the social distance detector which shows the probability score of detecting people in each frame and their approximate distance from the camera. In case of an issue or error in the detection, the user can easily copy the information and send it to the developer. Afterward, the program terminates.

5.8 Technical Explanation

In this section we look at how exactly the system works, the models, techniques, and tools used for each module of the SD&FM Detection System. The entire system was coded in Python, except for the GUI which is an executable Shell script. The python libraries used were Tensorflow, Keras, OpenCV, Numpy, Scikit-Learn, imutils, Matplotlib, Argparse, Sys, and OS. Both modules use the MobileNetv2 architecture. The face mask dataset upon which our model was trained was taken from Kaggle and had approximately 3000 images of people with and without face masks.

5.8.1 Face Mask Detector Module

The Face Mask Detector Module has two phases. In the first phase, we train our face mask detector model. We load our face mask dataset, train our model, then store the face mask classifier. In the second phase, we apply our face mask detector model that is we load it from the primary storage, detect faces in images or video streams, extract the region of interest, apply our face mask classifier to each extracted region of interest, and then show the results.

5.8.2 Social Distance Detector Module

The social distance detector has only one phase since we do not need to train a custom model. We can use class labels to filter out the class we want to detect which in our case is the ‘people’ class. After that we compute the pairwise distances between centroids, calculate the distance and show the results. The module also uses camera calibration to measure the approximate depth of a person in a frame. This is done using Triangle similarity to ensure that people standing further back in a frame are not mistakenly identified to not be maintaining social distance.

5.8.3 Other Features

The system is very lightweight and can be easily ported to embedded devices. The total size of the SD&FM Detection system is 160 MB only. The reason for using MobileNetV2 architecture was its resource-efficient nature since our system is designed for quick mass deployment on embedded devices, with good accuracy, efficiency, and low cost.

5.9 Design Methodology

The methodology used to implement the above-mentioned system is to do so by using a computer language that excels in machine learning, artificial intelligence, deep learning, real-time people detection, and tracking. Hence, the language chosen for this project is python. Using python, and a multitude of deep learning and computer vision libraries, we created a system that would detect if the distance between two or more people is more or less than 6 feet and detect whether or not each individual is wearing face masks. We designed the graphical user interface of the system in QT Designer and implemented it in shell script using zenity. The software development life cycle followed was the waterfall model. The SD & FM Detection system can be slightly modified by interested parties to fit a multitude of solution application areas such as opening a kitchen door only if a worker is wearing a face mask. The cost to implement this program in a real-time video stream and achieve a good frame rate is very high. We picked an architecture that was both relatively accurate, fast, and resource-efficient enough to be used in legacy or embedded systems. The following is an overview of the cost of different types of implementations of this system:

- Our system
  - Frame Rate: 15 FPS real-time detection
  - Platform: Laptop
  - Cross-platform: Yes
  - Cost: None
- Alternative System 1
  - Frame Rate: 2 FPS
  - Platform: Raspberry Pi
  - Cross-platform: Yes
  - Cost: 28,000+ PKR
- Alternative System 2
  - Frame Rate: 7 FPS
  - Platform: Raspberry Pi + NCS
  - Cross-Platform: No
  - Cost: 50,000+ PKR

Chapter 6

System Testing and Evaluation

6.1 Software Testing Techniques

Software testing techniques are standard models used to ensure whether the system at hand is performing the functionality it was initially designed for, that is whether the system is fulfilling the objective, functional and non-functional requirements defined in the early stages of the project. The testing of our SD&FM Detection System was rigorous. Each module was checked thoroughly during and after development. Some of the various testing techniques we used are given below:

- Functional Testing
- Unit Testing
- System Testing
- White-box and Black-box Testing
- Non-Functional Testing
- Performance Testing
- Acceptance Testing

6.2 Functional Testing

Our SD&FM Detection System fulfills all the functional requirements mentioned in chapter 3. Our system

- can detect faces in static images
- can detect people in static images
- displays invariance to light under normal daytime conditions
- possesses near real-time performance
- can detect multiple faces and people simultaneously
- can take input from a webcam video stream
- can take both images and video as input
- can tell whether an individual is wearing a face mask or not
- can tell whether an individual is maintaining approximately six feet distance with other people

6.2.1 Unit Testing

We performed unit testing on every module of the system such as Face mask detection on the image, video, webcam, and Social distance detection on video and webcam. Since each function has its separate module, isolating the functions was not too difficult. The SD&FM Detection System's modules passed all unit tests.

6.2.2 System Testing

System testing is done by combining all the modules of the system and checking if they work together. We combined the individual Face Mask Detection files into one module and the Social Distance Detection files into another module. We then wrote a Shell script to combine the whole system. Chapter 5 shows the efficient working of our entire system. The SD&FM Detection System passed all system tests.

6.2.3 Black Box Testing

We checked the output results of the SD&FM Detection System Modules concerning the functional specifications by giving random input. Our system was given random images and videos as test data. The system classified the face masks and social distance detection accurately. The test data was taken from Pexels.

6.2.4 White Box Testing

We internally checked the performance of our system from the user's (IT technician and security personnel) view to ensuring everything is working fine and smooth. The system passed white-box tests.

6.3 Non-Functional Testing

Our SD&FM Detection System fulfills all the non-functional requirements mentioned in chapter 3. Our system

- is relatively efficient, accurate, and reliable
- is sufficiently fast
- requires minimal to no maintenance
- does not demand enhanced security
- is available vastly and has low hardware requirement is resource-efficient.

6.3.1 Performance Testing

The overall performance of the SD&FM Detection System is very good. Thanks to the resource efficiency of MobileNetV2 architecture, the system can perform detections on images and videos within 10 seconds and also perform real-time detection through the webcam stream. After performing further performance tests in PyCharm we have concluded that our overall system is responsive and sufficiently fast.

6.3.2 Acceptance Testing

We checked whether the main requirements of the system are fulfilled. We checked all the functional requirements which we mentioned earlier in the objective section are achieved or not. The system passed all acceptance tests.

6.4 Test Case

We designed some test cases to test inputs and the executable functionality of the SD&FM Detection System.

6.4.1 Test Case 1: Face Mask Detection on Image

In this test case, the user starts the application and runs the ‘Face mask detector on image’ module. Uploads the image file and expects the output to be the same image with detected Face masks.

Table 4: Test Case FMD on Image Module

Test Case	01
Description	To check the working of the face mask detection on the image module.
Initial State	The application must be started by double-clicking 'app.sh' file.
Function to be tested	Module execution successful or not.
Test Execution	• Select option through the menu
	• Upload image file
Expected Result	The user should be shown the output image with detected face masks.
Actual Result	Output image with detected face masks shown.
Status	Pass.

6.4.2 Test Case 2: Face Mask Detection on Video

In this test case, the user starts the application and runs the ‘Face mask detector on video’ module. Uploads the video file and expects the output to be the same video with detected Face masks.

Table 5: Test Case FMD on Video Module

Test Case	02
Description	To check the working of the face mask detection on the video module.
Initial State	The application must be started by double-clicking 'app.sh' file.
Function to be tested	Module execution successful or not.
Test Execution	• Select option through the menu
	• Upload video file
Expected Result	The user should be shown the output video with detected face masks.
Actual Result	Output video with detected face masks shown.
Status	Pass.

6.4.3 Test Case 3: Face Mask Detection on Webcam

In this test case, the user starts the application and runs the 'Face mask detector on webcam' module, and expects the webcam to start and show detected Face masks in real-time.

Table 6: Test Case FMD on Webcam Module

Test Case	03
Description	To check the working of the face mask detection on the webcam module.
Initial State	The application must be started by double-clicking 'app.sh' file.
Function to be tested	Module execution successful or not.
Test Execution	• Select option through the menu
Expected Result	The user should be shown the webcam stream with real-time detected face masks.
Actual Result	Real-time webcam stream with detected face masks shown.
Status	Pass.

6.4.4 Test Case 4: Social Distance Detection on Video

In this test case, the user starts the application and runs the ‘Social Distance detector on video’ module. Uploads the video file and expects the output to be the same video with detected social distance.

Table 7: Test Case SDD on Video Module

Test Case	04
Description	To check the working of the social distance detection on the video module.
Initial State	The application must be started by double-clicking 'app.sh' file.
Function to be tested	Module execution successful or not.
Test Execution	• Select option through the menu
	• Upload video file
Expected Result	The user should be shown the output video with the detected social distance.
Actual Result	Output video with the detected social distance shown.
Status	Pass.

6.4.5 Test Case 5: Social Distance Detection on Webcam

In this test case, the user starts the application and runs the ‘Social Distance detector on webcam’ module and expects the webcam to start and show detected social distance in real-time.

Table 8: Test Case SDD on Webcam Module

Test Case	05
Description	To check the working of the social distance detection on the webcam module.
Initial State	The application must be started by double-clicking 'app.sh' file.
Function to be tested	Module execution successful or not.
Test Execution	• Select option through the menu
Expected Result	The user should be shown the webcam stream with real-time detected social distance.
Actual Result	Real-time webcam stream with the detected social distance shown.
Status	Pass.

Chapter 7

Conclusion

7.1 Conclusion

A year ago we got inspired by the idea of this project by observing the then situation of covid in our country where there was no system in place to ensure standard operating procedures on a large scale quickly and cost-effectively. A year later, and there is still no system in place to ensure people are following SOPs in both closed and open spaces.

The Social Distance and Face Mask Detection System is intended to facilitate the civilians and security personnel to be able to detect and enforce standard operating procedures. The system can help authorities reprimand individuals who don't maintain 6 feet distance or wear face masks and put others at risk of getting the virus. The GUI provided is simple and easy to understand and use. The system gives the user multiple mediums for detection such as images, video, or webcam.

While designing and implementing this system we aimed to overcome the various drawbacks of our existing competitor systems such as high cost, high barrier to adoption, additional fees, and lack of multiple types of detection. The SD&FM Detection system is designed to be lightweight, relatively accurate, and cheap to deploy – removing barriers to surveillance adoption. The system does not require any sort of contracts, sign up or fees and does not lock the user into a certain platform. The user can freely modify the system with the help of any third party. Table 9 shows the Feature comparison of our system with existing competitor systems mentioned in Chapter 2.

Table 9: Feature Comparison Table

SR. No.	Solution	Face Mask Detection	Social Distance Detection	Cost	Barrier to Adoption	Additional Fees
1	Ours	Yes	Yes	Low	Low	Low
2	HikVision	Yes	No	Med	High	High
3	Rhombus	Yes	Yes	High	High	Med
4	Tyco	Yes	No	High	High	High
5	FindFace	Yes	Yes	High	Med	High
6	Invixium	Yes	No	High	Med	Low

7.2 Recommendations

Although our system does fairly well against existing competitor systems, there is still room for improvement. Regarding this system, we would like to recommend that the following work can be done to enhance the SD&FM Detection System such as:

- To train the face mask detection model with a better dataset to enhance its accuracy with different or obscure faces.
- To improve the social distance detection module by adding bird eye view calibration which would result in more accurate distance measurement.
- To add an overlay dashboard that provides more information to the user such as the total number of people at risk.

7.3 Learning Outcomes

From this project, we came to know how to make a computer vision application under the supervision of our supervisor for the final year project and following the deadlines side by side. It also gives us the benefits to understand how real-world projects are carried out in IT firms. The project has made us learn so many new as well as important things. The purpose of the final semester training is fulfilled with this project. And above all, the things which I have learned will be useful for all the upcoming projects and will give me an edge in the market.

Chapter 8

References

[1] Velavan, Thirumalaisamy P., and Christian G. Meyer. "The COVID-19 epidemic." *Tropical medicine & international health* 25.3 (2020): 278.

[2] Wikipedia contributors. “Economic Impact of the COVID-19 Pandemic.” *Wikipedia*, 1 May 2021, en.wikipedia.org/wiki/Economic_impact_of_the_COVID-19_pandemic.

[3] Segal, Edward. “How To Ensure Your Company Survives The Covid-19 Pandemic.” *Forbes*, 7 Nov. 2020, www.forbes.com/sites/edwardsegal/2020/11/06/how-business-executives-can-ensure-that-their-companies-will-survive-a-prolonged-coronavirus-crisis.

[4] Brothers, Will. “A Timeline of COVID-19 Vaccine Development.” *BioSpace*, 3 Dec. 2020, www.biospace.com/article/a-timeline-of-covid-19-vaccine-development.

[5] Luskin, Donald. “The Failed Experiment of Covid Lockdowns.” *WSJ*, 2 Sept. 2020, www.wsj.com/articles/the-failed-experiment-of-covid-lockdowns-11599000890.

[6] “COVID-19: How to Protect Yourself & Others.” *Centers for Disease Control and Prevention*, 8 Mar. 2021, www.cdc.gov/coronavirus/2019-ncov/prevent-getting-sick/prevention.html.

[7] Dourlain, Marisa. “Real-Time Face Mask Detection Addresses a Key COVID-19 Concern.” *CDW Solutions Blog*, 30 Oct. 2020, blog.cdw.com/software/real-time-face-mask-detection-addresses-a-key-covid-19-concern.

[8] “Using Artificial Intelligence to Help Combat COVID-19.” *OECD*, 2020, www.oecd.org/coronavirus/policy-responses/using-artificial-intelligence-to-help-combat-covid-19-ae4c5c21.

[9] Tsang, Sik-Ho. “Review: MobileNetV2 — Light Weight Model (Image Classification).” *Medium*, 1 Aug. 2019, towardsdatascience.com/review-mobilenetv2-light-weight-model-image-classification-8febb490e61c.

[10] Incze, Raul. “The Cost of Machine Learning Projects - Cognifeed.” *Medium*, 14 Sept. 2019, medium.com/cognifeed/the-cost-of-machine-learning-projects-7ca3aea03a5c.

[11] Huang, Thomas. "Computer vision: Evolution and promise." (1996).
[12] “Computer Vision: Timeline.” *SAS*, 2021, www.sas.com/en_us/insights/analytics/computer-vision.html.
[13] Future, Market Research. “Computer Vision Market Worth USD 48 Billion by 2023.” *GlobeNewswire News Room*, 9 Apr. 2019, www.globenewswire.com/news-release/2019/04/09/1799533/0/en/Computer-Vision-Market-Worth-USD-48-Billion-by-2023-The-Emergence-of-Computers-is-Intensifying-Global-Computer-Vision-Market-to-Prosper-with-Major-Developments-in-Virtual-Reality.html.
[14] Peregud, Irina. “Computer Vision Applications Examples Across Different Industries.” *InData Labs*, 3 May 2021, indatalabs.com/blog/applications-computer-vision-across-industries.
[15] Dickson, Ben. “What Is Computer Vision?” *PCMAG*, 9 Feb. 2020, www.pcmag.com/news/what-is-computer-vision.
[16] Techopedia. “Convolutional Neural Network (CNN).” *Techopedia.Com*, 5 Sept. 2018, www.techopedia.com/definition/32731/convolutional-neural-network-cnn.
[17] “Yann LeCun’s Biography.” *Yann Le Cun*, 2015, yann.lecun.com/ex/bio.html.
[18] Great Learning Team. “Types of Neural Networks.” *GreatLearning Blog*, 26 Apr. 2021, www.mygreatlearning.com/blog/types-of-neural-networks.
[19] Spirina, Katrine. “How Does Computer Vision Work.” *InData Labs*, 3 May 2021, indatalabs.com/blog/how-does-computer-vision-work.
[20] “Explained: Neural Networks.” *MIT News | Massachusetts Institute of Technology*, 14 Apr. 2017, news.mit.edu/2017/explained-neural-networks-deep-learning-0414.
[21] “Neural Networks - What Are They and Why Do They Matter?” *SAS*, 2021, www.sas.com/en_us/insights/analytics/neural-networks.html.
[22] “A Beginner’s Guide to Neural Networks and Deep Learning.” *Pathmind*, 2020, wiki.pathmind.com/neural-network.
[23] Admin. “What Are Different Layers in Neural Networks?” *I2tutorials*, 18 Oct. 2019, www.i2tutorials.com/what-are-different-layers-in-neural-networks.
[24] Education, Ibm Cloud. “Deep Learning.” *IBM*, 30 Apr. 2021, www.ibm.com/cloud/learn/deep-learning.
[25] “What Is Machine Learning?” *SAP Insights*, 5 Mar. 2021, insights.sap.com/what-is-machine-learning.
[26] IBM. “Machine Learning.” *IBM*, 25 Mar. 2021, www.ibm.com/cloud/learn/machine-learning.
[27] Education, Ibm Cloud. “Artificial Intelligence (AI).” *IBM*, 4 May 2021, www.ibm.com/cloud/learn/what-is-artificial-intelligence.
[28] Zupan, Jure. "Introduction to artificial neural network (ANN) methods: what they are and how to use them." *Acta Chimica Slovenica* 41 (1994): 327-327.
[29] DeepAI. “Perceptron.” *DeepAI*, 25 June 2020, deepai.org/machine-learning-glossary-and-terms/perceptron.
[30] Lateef, Zulaikha. “What Is A Neural Network? Introduction To Artificial Neural Networks.” *Edureka*, 28 Aug. 2019, www.edureka.co/blog/what-is-a-neural-network/#What%20is%20Deep%20Learning.
[31] Bhardwaj, Anjali. “What Is a Perceptron? – Basics of Neural Networks - Towards Data Science.” *Medium*, 12 Oct. 2020, towardsdatascience.com/what-is-a-perceptron-basics-of-neural-networks-c4cfea20c590.

[32] Countz, Thomas. “Perceptrons in Neural Networks - Thomas Countz.” *Medium*, 29 Mar. 2018, medium.com/@thomascountz/perceptrons-in-neural-networks-dc41f3e4c1b9.
[33] Mehta, Anukrati. “A Comprehensive Guide To Types Of Neural Networks.” *Digital Vidya*, 16 Oct. 2020, www.digitalvidya.com/blog/types-of-neural-networks.
[34] Chen, Jessie. “Object Detection: What Is It and How Is It Useful?” *PanaCast*, 24 July 2018, www.panacast.com/object-detection-what-is-it-and-how-is-it-useful.
[35] *Guan, Ling, ed. Multimedia image and video processing. CRC press, 2017.*
[36] Wu, Jianxin, et al. "A scalable approach to activity recognition based on object use." *2007 IEEE 11th international conference on computer vision*. IEEE, 2007.
[37] “Object Detection.” *Wikipedia*, 30 Apr. 2021, en.wikipedia.org/wiki/Object_detection#cite_note-GuanHe2012-2.
[38] Solawetz, Jacob. “The Ultimate Guide to Object Detection.” *Roboflow Blog*, 30 Apr. 2021, blog.roboflow.com/object-detection.
[39] “Region Based Convolutional Neural Networks.” *Wikipedia*, 7 Feb. 2021, en.wikipedia.org/wiki/Region_Based_Convolutional_Neural_Networks.
[40] Świeżewski, Jędrzej. “YOLO Algorithm and YOLO Object Detection: An Introduction - Appsilon | End To End Data Science Solutions.” *Appsilon | EndÂ ToÂ End Data Science Solutions*, 26 Mar. 2021, appsilon.com/object-detection-yolo-algorithm.
[41] Dash, Abhipraya Kumar. “Single Shot Detection (SSD) Algorithm.” *OpenGenus IQ: Learn Computer Science*, 21 Jan. 2019, iq.opengenus.org/single-shot-detection-ssd-algorithm.
[42] “Training Custom Object Detector — TensorFlow 2 Object Detection API Tutorial Documentation.” *TensorFlow*, 2021, tensorflow-object-detection-api-tutorial.readthedocs.io/en/latest/training.html.
[43] Yadam, Gopikrishna. “Object Detection Using TensorFlow and COCO Pre-Trained Models.” *Medium*, 21 May 2019, medium.com/object-detection-using-tensorflow-and-coco-pre/object-detection-using-tensorflow-and-coco-pre-trained-models-5d8386019a8.
[44] Jain, Taru. “Basics of Machine Learning Image Classification Techniques.” *OpenGenus IQ: Learn Computer Science*, 29 Aug. 2019, iq.opengenus.org/basics-of-machine-learning-image-classification-techniques.
[45] Brar, Harman. “Object Localization - Harman Brar.” *Medium*, 6 Dec. 2018, medium.com/@harman4422/object-localization-bd314d7e648f.
[46] Lim, Hengtee. “Everything You Need to Know About Object Detection Systems.” *Lionbridge AI*, 30 Nov. 2020, lionbridge.ai/articles/everything-you-need-to-know-about-object-detection-systems.
[47] Elfouly, Sharif. “R-CNN (Object Detection) - Sharif Elfouly.” *Medium*, 19 Nov. 2020, medium.com/@selfouly/r-cnn-3a9beddfd55a.
[48] GeeksforGeeks. “R-CNN | Region Based CNNs.” *GeeksforGeeks*, 1 Mar. 2020, www.geeksforgeeks.org/r-cnn-region-based-cnns.
[49] Balasubramanian, Ramji. “Region — Based Convolutional Neural Network (RCNN) - Analytics Vidhya.” *Medium*, 28 Jan. 2021, medium.com/analytics-vidhya/region-based-convolutional-neural-network-rcnn-b68ada0db871.
[50] Vadapalli, Pavan. “Ultimate Guide to Object Detection Using Deep Learning [2021].” *UpGrad Blog*, 11 Jan. 2021, www.upgrad.com/blog/ultimate-guide-to-object-detection-using-deep-learning.
[51] GeeksforGeeks. “Faster R-CNN | ML.” *GeeksforGeeks*, 1 Mar. 2020, www.geeksforgeeks.org/faster-r-cnn-ml.
[52] “Introduction to YOLO Algorithm for Object Detection.” *Section*, 2021, www.section.io/engineering-education/introduction-to-yolo-algorithm-for-object-detection.

[53] Znreza. “Object Detection with Single Shot Multibox Detector – AI DIARY OF ZNREZA.” *AIDiary*, 11 May 2019, ai-diary-by-znreza.com/object-detection-with-single-shot-multibox-detector.
[54] Liu, Wei, et al. "Ssd: Single shot multibox detector." *European conference on computer vision*. Springer, Cham, 2016.
[55] Awesome, Python. “SSD: Single Shot MultiBox Detector | a PyTorch Tutorial to Object Detection.” *Python Awesome*, 13 Jan. 2020, pythonawesome.com/ssd-single-shot-multibox-detector-a-pytorch-tutorial-to-object-detection.
[56] Nature Editorial. “Why Many Countries Failed at COVID Contact-Tracing — but Some Got It Right.” *Nature*, 2020, www.nature.com/articles/d41586-020-03518-4?error=cookies_not_supported&code=636ef067-4ce6-4201-a51a-c38fa62d2f53.
[57] Sharma, Mukesh. “How Drones Are Being Used to Combat COVID-19.” *Geospatial World*, 21 Apr. 2020, www.geospatialworld.net/blogs/how-drones-are-being-used-to-combat-covid-19.
[58] “Hikvision - Video Security System and IoT Solutions.” *Hiknow*, 2021, www.hikvision.com/en.
[59] Halcyonmss. “Hikvision Face Mask, Fever & Temperature Screening Systems.” *Halcyon CCTV Experts - San Diego*, 14 May 2020, halcyoncctvexperts.com/hikvision-face-mask-fever-temperature-screening-systems.
[60] “These Chinese firms were blacklisted for Uighur oppression. Now they want to sell COVID-19 surveillance tools to the West.” *Business Insider Nederland*, 6 June 2020, www.businessinsider.nl/blacklisted-chinese-firms-uighur-oppression-covid-19-surveillance-tech-2020-6?international=true&r=US.
[61] “Rhombus - Next-Generation Enterprise Security Camera System.” *Rhombus Camera*, 2021, www.rhombussystems.com.
[62] Rhombus Systems. “Rhombus Systems Releases Smart Security Cameras with Mask Detection to Help Reduce the Spread of COVID-19.” *PRNewWire*, 7 Jan. 2021, www.prnewswire.com/news-releases/rhombus-systems-releases-smart-security-cameras-with-mask-detection-to-help-reduce-the-spread-of-covid-19-301202460.html.
[63] “Safer Reopening with Smart Security Cameras: Introducing Rhombus Mask Detection.” *Rhombus Systems*, 4 Jan. 2021, www.rhombussystems.com/blog/safer-reopening-with-smart-security-cameras-introducing-rhombus-mask-detection.
[64] “Exacq Technologies.” *Tyco*, 2021, tyco-tsp.com/exacq.
[65] “Tyco AI Software | Exacq from Tyco Security Products.” *Face Mask Detection*, 2021, exacq.com/products/tyco-ai.
[66] Products, Tyco Security. “Face Mask Detection Applications Offer a Proactive Approach to Facility Safety | Exacq Blog.” *Tyco AI*, 14 Sept. 2020, blog.tycosp.com/exacq/2020/09/14/face-mask-detection-applications-offer-a-proactive-approach-to-facility-safety.
[67] NTechLab. “Facial Recognition Software | FindFace.” *FindFace Pro*, 6 Nov. 2020, findface.pro/en.
[68] NTechLab. “Biometric Solution against COVID-19.” *FindFace Pro*, 30 Oct. 2020, findface.pro/en/solution/biometric-solution-against-covid-19.
[69] Brewster, Thomas. “Remember FindFace? The Russian Facial Recognition Company Just Turned On A Massive, Multimillion-Dollar Moscow Surveillance System.” *Forbes*, 29 Jan. 2020, www.forbes.com/sites/thomasbrewster/2020/01/29/findface-rolls-out-huge-facial-recognition-surveillance-in-moscow-russia.

[70] Invixium. “Global Leader in Biometric Access Control Solutions.” *Invixium*, 6 May 2021, www.invixium.com.
[71] Invixium. “Face Mask Detection Solution | Invixium Mask Detection.” *Invixium*, 5 May 2021, www.invixium.com/face-recognition-mask-detection-system.
[72] Invixium. “Intelligent Biometric Access Control with Facial Recognition | IXM Titan.” *Invixium*, 3 May 2021, www.invixium.com/ixm-titan-face-recognition.