AUTHORIZED ENTRY USING FACE MASK DETECTION

A Project Report

Submitted in partial fulfilment of the
requirements for the award of the Degree of

BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

By

Khushali Chandrashekhar Bhurke

Seat Number: 18302B00540

Under the esteemed guidance of

Mrs. Prachi Mahajan

Assistant Professor, Department of Information Technology

VSIT | Vidyalankar School of Information Technology

DEPARTMENT OF INFORMATION TECHNOLOGY

VIDYALANKAR SCHOOL OF INFORMATION TECHNOLOGY

(Affiliated to University of Mumbai)

MUMBAI, 400 037

MAHARASHTRA

2020 - 2021

VIDYALANKAR SCHOOL OF INFORMATION TECHNOLOGY

(Affiliated to University of Mumbai)

MUMBAI-MAHARASHTRA-400037

DEPARTMENT OF INFORMATION TECHNOLOGY

VSIT | Vidyalankar School of Information Technology

CERTIFICATE

This is to certify that the project entitled, **"AUTHORIZED ENTRY USING FACE MASK DETECTION "**, is bonafied work of **Khushali Chandrashekhar Bhurke** bearing Seat No: 18302B0054 submitted in partial fulfilment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY from University of Mumbai.

Internal Guide	**Coordinator**
Internal Examiner	**External Examiner**

Date:	**College Seal**	**Principal**

ABSTRACT

The corona virus COVID-19 pandemic is causing a global health crisis so the effective protection methods is wearing a face mask in public areas according to the World Health Organization (WHO). The COVID-19 pandemic forced governments across the world to impose lockdowns to prevent virus transmissions.

Reports indicate that wearing face masks while at work clearly reduces the risk of transmission. An efficient and economic approach of using AI to create a safe environment in a manufacturing setup. A hybrid model using deep and classical machine learning for face mask detection will be presented. A face mask detection dataset consists of with mask and without mask images , we are going to use OpenCV to do real-time face detection from a live stream via our webcam.

We will use the dataset to build a COVID-19 face mask detector with computer vision using Python, OpenCV, and Tensor Flow and Keras. Our goal is to identify whether the person on image/video stream is wearing a face mask or not with the help of computer vision and deep learning.

Key Words: Deep Learning, Computer Vision, OpenCV, Tensorflow, Keras.

ACKNOWLEDGEMENT

We would like to express our special thanks and gratitude to ur project guide **Mrs. Prachi Mahajan** for guiding us to do the project work on time and giving us all support and guidance, which made complete our project duly. We are extremely thankful to her for providing such nice support and guidance.

We are also thankful for and fortunate enough to get constant encouragement, support and guidance from the teachers of information Technology who helped us in successfully completing our project work.

DECLARATION

I hereby declare that the project entitled, “**Authorized Entry Using Face Mask Detection**” done at Vidyalankar School of Information Technology, has not been in any case duplicated to submit to any other universities for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfillment of the requirements for the award of degree of **BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)** to be submitted as final semester project as part of our curriculum.

Name and Signature of the Student

Table of Contents

List of Tables

List of Figures

Chapter 1 : Introduction

1.1 BACKGROUND

ALMOST everyone wears a mask during the COVID-19 coronavirus epidemic. Face recognition techniques, the most important means of identification, have nearly failed, which has brought huge dilemmas to authentication applications that rely on face recognition, such as community entry and exit, face access control, face attendance, face gates at train stations, face authentication based mobile payment, face recognition based social security investigation, etc.

In particular, in the public security check like railway stations, the gates based on traditional face recognition systems cannot effectively recognize the masked faces, but removing masks for passing authentication will increase the risk of virus infection. Because the COVID-19 virus can be spread through contact, the unlocking systems based on passwords or fingerprints are unsafe. It is much safer through face recognition without touching, but the existing face recognition solutions are no longer reliable when wearing a mask.

To solve above mentioned difficulties, it is necessary to improve the existing face recognition approaches that heavily rely on all facial feature points, so that identity verification can still be performed reliably in the case of incompletely exposed faces The state-of-the-art face recognizers are all designed based on deep learning, which depend on massive training dataset.

Thus, developing face recognition algorithms for masked faces requires a large number of masked face samples. At present, there is no publicly available masked face dataset, and so this work proposes to construct masked face datasets by different means.

1.2 Objective

To identify the person on image/video stream wearing face mask with the help of computer vision and deep learning algorithm by using the OpenCV and TensorFlow.

1.3 purpose , features , Attributes , Applications

1.3.1 Purpose of the model

considering the covid-19 outbreak, i think this is best project that i can work as IT student. Today everyone is aware of taking precaution and safety measures regarding covid-19, so face mask detection will play a huge role to avoid corona virus.

This project helps us to spread the awareness among people using face mask properly. It detects the face mask on your face whether the person is hiding his/her face by mask or not. it also check the face mask is properly cover your face both nose and mouth.

1.3.2 Features of the Face Mask Detection System

The system is easy to implement in any existing organizational system.

Custom alerts can be sent to the person with or without a face mask or the one whose face is unrecognizable in the admin system.

No need to install any hardware as the system can be connected with your existing surveillance system only.

The system can be used easily with any camera or hardware like surveillance cameras.

The system restricts access for those not wearing the masks and notifies the authorities.

You can customize the face mask detection system based on your business requirements.

You can check the analytics based on the system generated reports.

Easy to access and control the movements from any device through face mask detection applications.

Partially occluded faces either with mask or hair or hand, can be easily detected.

1.3.3 Attributes Matter the Most!

Detecting masked faces is challenging because the system can not detect incomplete or inaccurate facial features. But thanks to advanced technology and continuous research and studies performed by so many tech-leaders. Moreover, artificial intelligence (AI) and machine learning (ML) communities are developing various models withstanding the urgent need for detecting masked faces. Detecting masked faces requires making a system that can identify the available datasets of a masked face, i.e., facial features. The advanced detection technology includes various attributes as mentioned hereunder:

Location of Faces, annotated by a shape i.e., square

Face Orientation, includes front, left, left-front, right, right-front.

Location of Eyes, need to mark eye centers.

Location of Masks, annotated by a shape i.e., rectangles.

Type of Masks, i.e., human-made masks with or without logo, face covered by hand, etc.

Occlusion Degree, defining a face into four regions – eyes, nose, chin, and mouth

1.3.4 Applications

face mask detection application

Airports

The Face Mask Detection System can be used at airports to detect travelers without masks. Face data of travelers can be captured in the system at the entrance. If a traveler is found to be without a face mask, their picture is sent to the airport authorities so that they could take quick action. If the person's face is already stored, like the face of an Airport worker, it can send the alert to the worker's phone directly.

Hospitals

Using Face Mask Detection System, Hospitals can monitor if their staff is wearing masks during their shift or not. If any health worker is found without a mask, they will receive a notification with a reminder to wear a mask. Also, if quarantine people who are required to wear a mask, the system can keep an eye and detect if the mask is present or not and send notification automatically or report to the authorities.

Offices

The Face Mask Detection System can be used at office premises to detect if employees are maintaining safety standards at work. It monitors employees without masks and sends them a reminder to wear a mask. The reports can be downloaded or sent an email at the end of the day to capture people who are not complying with the regulations or the requirements.

CHAPTER 2 : SURVEY OF TECNOLOGY

2.1 Four Outstanding Features of Raspberry Pi That Need To Know

The Raspberry Pi 3 B+ is capable of outperforming the other models in the market because of the many features that it possesses.

Raspberry Pi Model B+ features

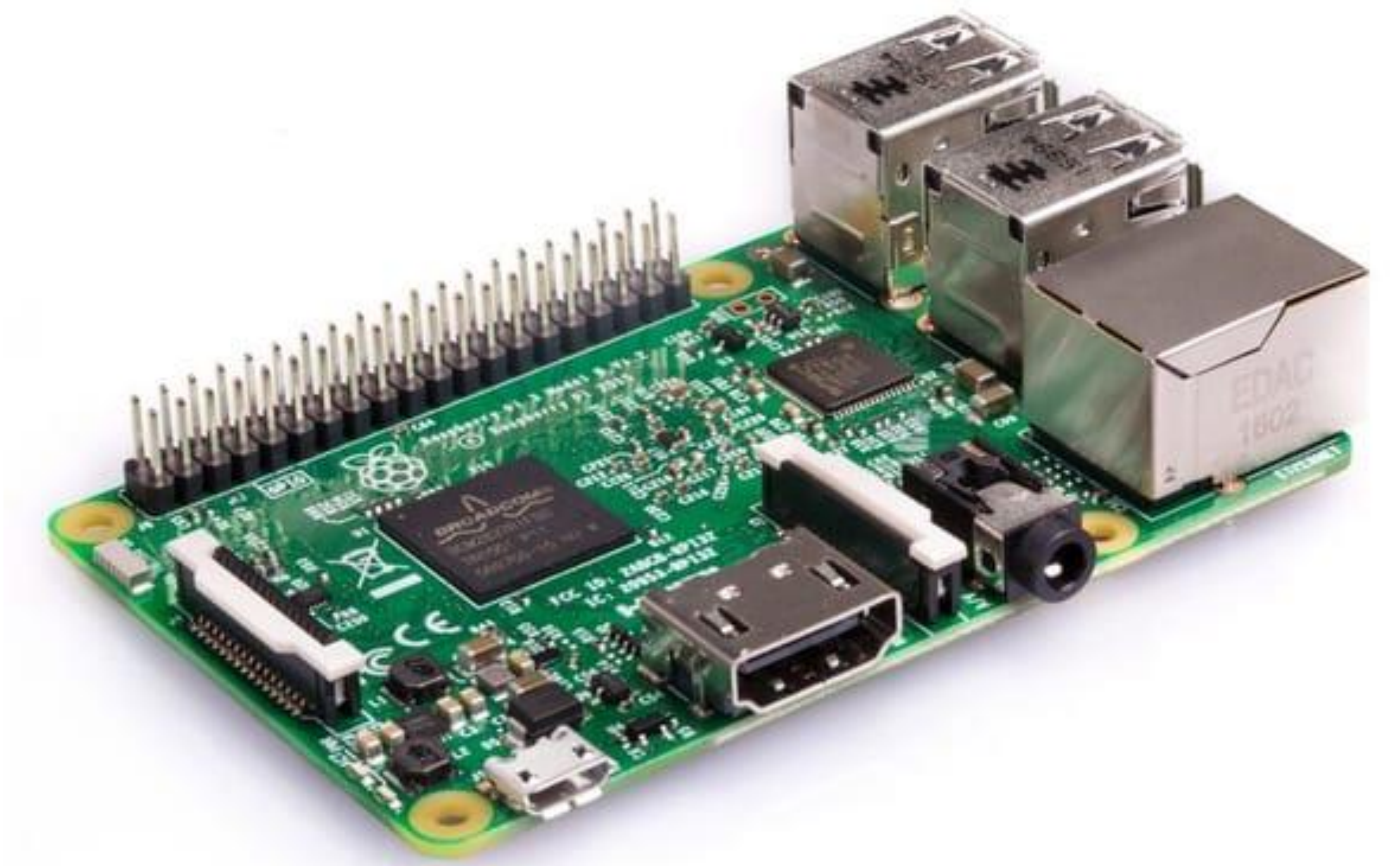


Figure 1 2.1 Raspberry Pi board

Given on the list below are the outstanding features about Raspberry Pi board.

It Comes With a 4.2 Bluetooth

bluetooth-42asdPlacing first on the Raspberry Pi 3 B+ outstanding features is that it comes with a 4.2 Bluetooth which can provide a much faster and greater connection of Ethernet.

Furthermore, it can also establish a significant compatibility of Ethernet when you choose to equip it with an accessory of PoE HAT. With this feature, the Raspberry 3 B+ has already gained a lot of potential buyers even though that it is just recently launched.

It Has a Dual-Band Wi-Fi Feature

wifi strong signalThe next outstanding feature that Raspberry Pi 3 B+ has is that the board is included with a dual-band Wi-Fi which means that it can provide both 2.4 GHz wireless and 5 GHz.

This great feature makes the Raspberry 3 B+ very versatile and adaptable to everybody.

It is also the reason why this new hardware is slowly making its way to the top, thus, beating the other models.

It Has a More Improved Processor

Another outstanding feature to know is that Raspberry Pi 3 B+ has a more improved processor as compared to the other available models in the market and even to the other Raspberry’s models.

The processor has a 64 – bit 1.4 GHz quad-core which can target much higher frequencies of the clock.

Not only that, it can also control and monitor the chip's temperature in a more precise and exact way.

Therefore, the Raspberry 3 B+ is definitely a great and smart choice that you should do when you are looking for this kind of product.

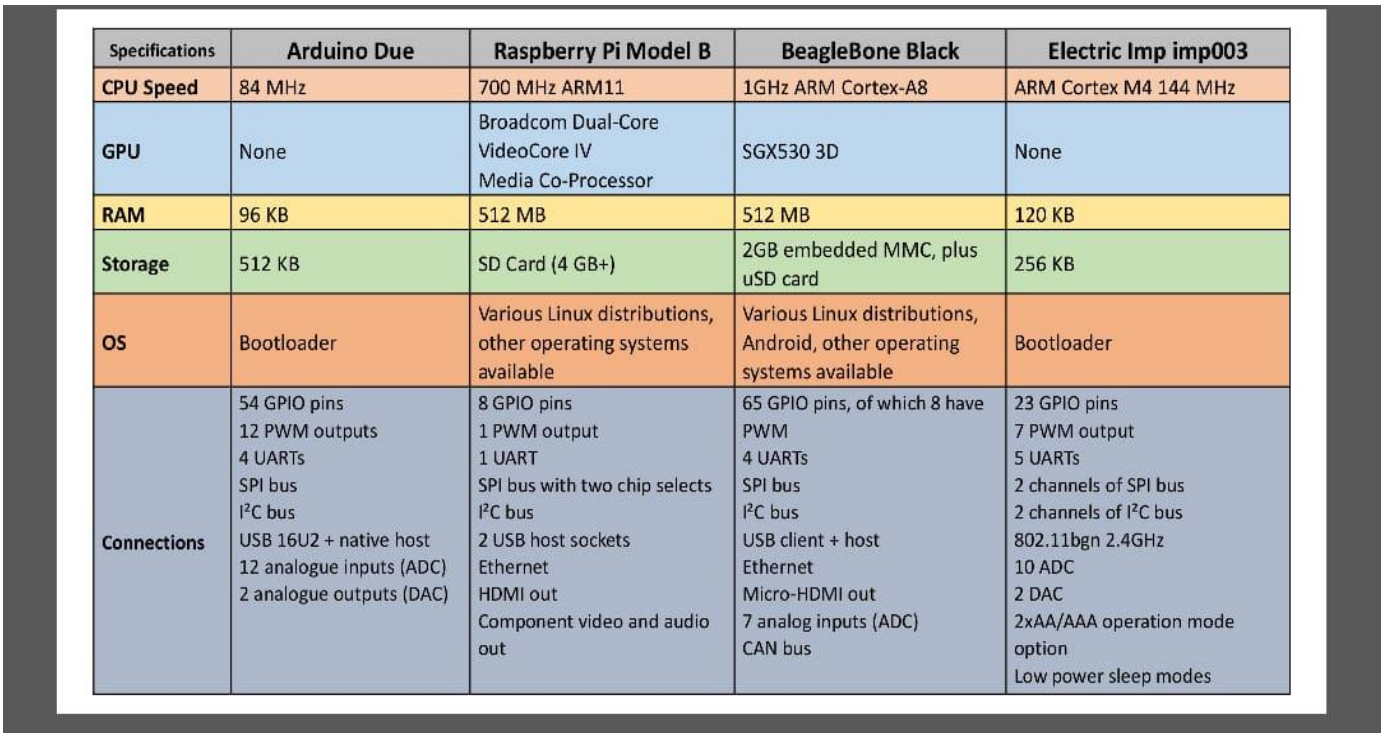
It Uses A 2.5A Power

Last, but not the least feature of Raspberry Pi 3 B+ is that this new hardware uses a 2.5A power.

In spite of its outstanding features, this latest product only comes at the affordable price.

These outstanding features on the list are just some of it and there are so much more for you to explore.

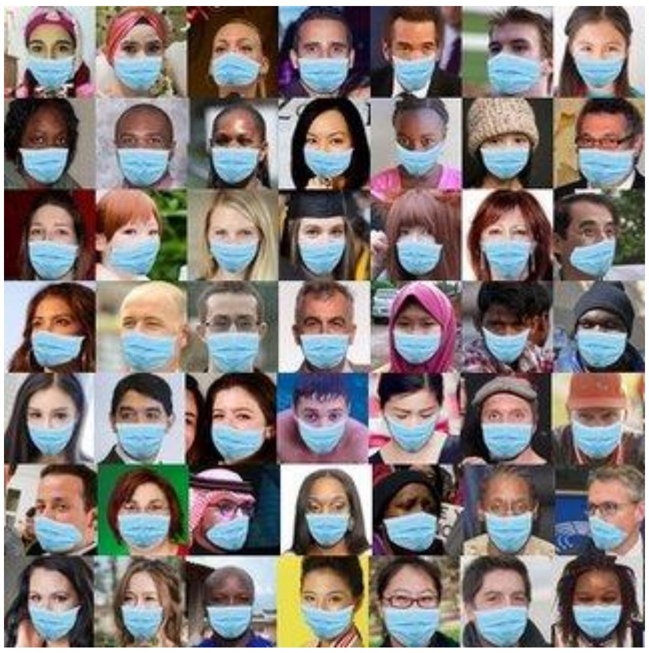
Comparison between Raspberry Pi 3 and other boards available



Specifications	Arduino Due	Raspberry Pi Model B	BeagleBone Black	Electric Imp imp003
CPU Speed	84 MHz	700 MHz ARM11	1GHz ARM Cortex-A8	ARM Cortex M4 144 MHz
GPU	None	Broadcom Dual-Core VideoCore IV Media Co-Processor	SGX530 3D	None
RAM	96 KB	512 MB	512 MB	120 KB
Storage	512 KB	SD Card (4 GB+)	2GB embedded MMC, plus uSD card	256 KB
OS	Bootloader	Various Linux distributions, other operating systems available	Various Linux distributions, Android, other operating systems available	Bootloader
Connections	54 GPIO pins 12 PWM outputs 4 UARTs SPI bus I^2C bus USB 16U2 + native host 12 analogue inputs (ADC) 2 analogue outputs (DAC)	8 GPIO pins 1 PWM output 1 UART SPI bus with two chip selects I^2C bus 2 USB host sockets Ethernet HDMI out Component video and audio out	65 GPIO pins, of which 8 have PWM 4 UARTs SPI bus I^2C bus USB client + host Ethernet Micro-HDMI out 7 analog inputs (ADC) CAN bus	23 GPIO pins 7 PWM output 5 UARTs 2 channels of SPI bus 2 channels of I^2C bus 802.11bgn 2.4GHz 10 ADC 2 DAC 2xAA/AAA operation mode option Low power sleep modes

Table 1 Difference between Raspberry Pi and others

Our COVID-19 face mask detection dataset



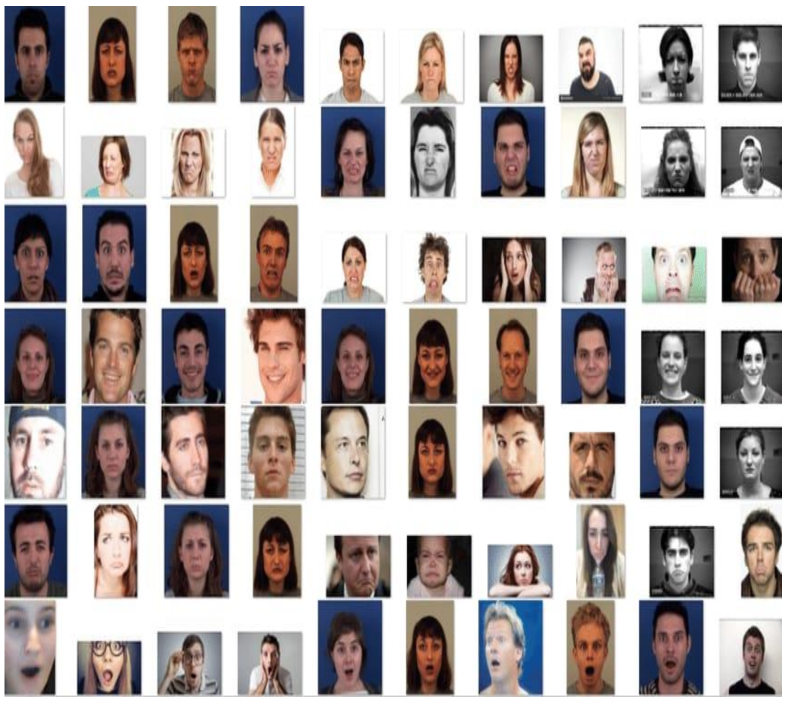


Figure 2 people with and without mask

The dataset we’ll be using here was created by PyImageSearch .

This dataset consists of 1,376 images belonging to two classes:

with_mask: 690 images

without_mask: 686 images

Our goal is to train a custom deep learning model to detect whether a person is or is not wearing a mask.

How is our face mask dataset created?

To create this dataset, PyImageSearch reader had the solution of:

- Taking normal images of faces
- Then creating a custom computer vision Python script to add face masks to them, thereby creating an artificial (but still real-world applicable) dataset
- This method is actually a lot easier once you apply facial landmarks to the problem.
- Facial landmarks allow us to automatically infer the location of facial structures, including:

  Eyes

  Eyebrows

  Nose

  Mouth

  Jawline

- To use facial landmarks to build a dataset of faces wearing face masks, we need to first start with an image of a person not wearing a face mask.

From there, we apply face detection to compute the bounding box location of the face in the image.

- Once we know where in the image the face is, we can extract the face Region of Interest (ROI).

Next, we need an image of a mask (with a transparent background) And from there, we apply facial landmarks, allowing us to localize the eyes, nose, mouth, etc.

- This mask will be automatically applied to the face by using the facial landmarks (namely the points along the chin and nose) to compute where the mask will be placed.

- The mask is then resized and rotated, placing it on the face.
- We can then repeat this process for all of our input images, thereby creating our artificial face mask dataset

Figure 3 location of facial structure

CHAPTER 3: REQUIRMENTS AND ANALYSIS

3.1 Problem Definition

The trend of wearing face masks in public is rising due to the COVID- 19 corona virus epidemic all over the world. Before Covid-19, People used to wear masks to protect their health from air pollution.

Scientists proofed that wearing face masks works on impeding COVID-19 transmission. COVID-19 (known as corona virus) is the latest epidemic virus that hit the human health in the last century. In 2020, the rapid spreading of COVID-19 has forced the World Health Organization to declare COVID- 19 as a global pandemic.

More than five million cases were infected by COVID-19 in less than 6 months across 188 countries. The virus spreads through close contact and in crowded and overcrowded areas.

Artificial Intelligence (AI) based on Machine learning and Deep Learning can help to fight Covid-19 in many ways. Machine learning allows researchers and clinicians evaluate vast quantities of data to forecast the distribution of COVID-19, to serve as an early warning mechanism for potential pandemics, and to classify vulnerable populations.The provision of healthcare needs funding for emerging technology such as artificial intelligence, IoT, big data and machine learning to tackle and predict new diseases.

People are forced by laws to wear face masks in public in many countries. These rules and laws were developed as an action to the exponential growth in cases and deaths in many areas. However, the process of monitoring large groups of people is becoming more difficult. The monitoring process involves the detection of anyone who is not wearing a face mask.

Here we introduce a mask face detection model that is based on computer vision and deep learning. The proposed model can be integrated with surveillance cameras to impede the COVID-19 transmission by allowing the detection of people who are wearing masks not wearing face masks. The model is integration between deep learning and classical machine learning techniques with opencv, tensor flow and keras.

3.2 REQUIRMENTS SPECIFICATION

Neural Network (CNN) model using *TensorFlow* with *Keras* library and *OpenCV* to detect if you are wearing a face mask to protect yourself.

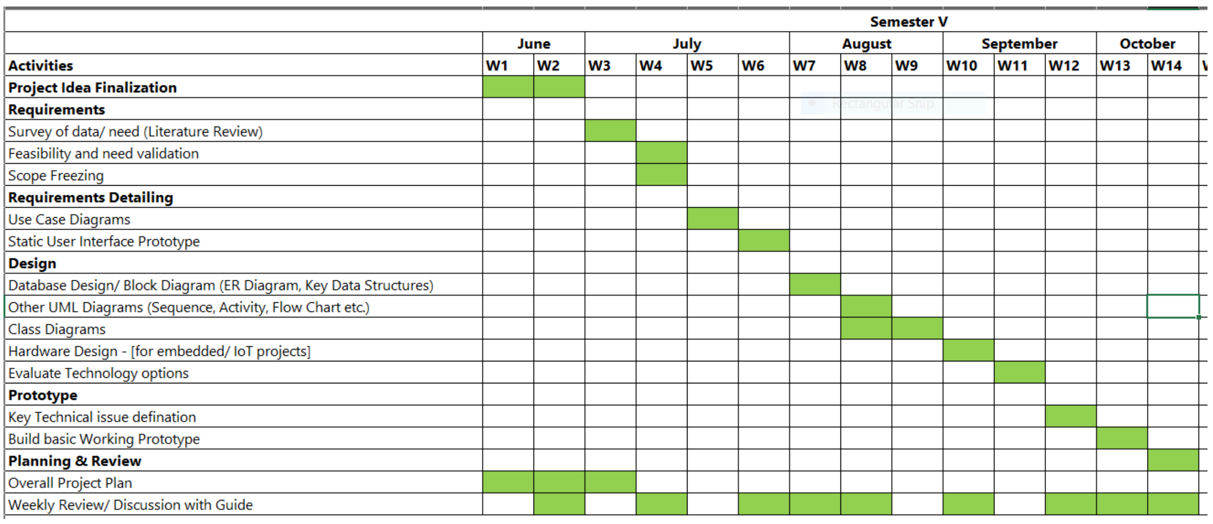
In this project, use our two-phase COVID-19 face mask detector, detailing how our computer vision/deep learning pipeline will be implemented.

From there, we'll review the dataset we'll be using to train our custom face mask detector.

Then implement a Python script to train a face mask detector on our dataset using Keras and TensorFlow.

We'll use this Python script to train a face mask detector and review the results. Given the trained COVID-19 face mask detector, we'll proceed to implement two more additional Python scripts used to Detect face masks in real-time video streams.

3.3 PLANNING AND SCHEDULING



	Semester V													
	June		July				August			September			October	
Activities	**W1**	**W2**	**W3**	**W4**	**W5**	**W6**	**W7**	**W8**	**W9**	**W10**	**W11**	**W12**	**W13**	**W14**
Project Idea Finalization	🟩	🟩												
Requirements														
Survey of data/ need (Literature Review)			🟩											
Feasibility and need validation				🟩										
Scope Freezing				🟩										
Requirements Detailing														
Use Case Diagrams					🟩									
Static User Interface Prototype						🟩								
Design														
Database Design/ Block Diagram (ER Diagram, Key Data Structures)							🟩							
Other UML Diagrams (Sequence, Activity, Flow Chart etc.)								🟩						
Class Diagrams								🟩	🟩					
Hardware Design - [for embedded/ IoT projects]										🟩				
Evaluate Technology options											🟩			
Prototype														
Key Technical issue defination												🟩		
Build basic Working Prototype													🟩	
Planning & Review														🟩
Overall Project Plan	🟩	🟩	🟩											
Weekly Review/ Discussion with Guide		🟩		🟩		🟩	🟩	🟩		🟩		🟩	🟩	🟩

Figure 4 Gantt chart

In this project, we are using the waterfall model because it helps to implement each phase of system requirements.

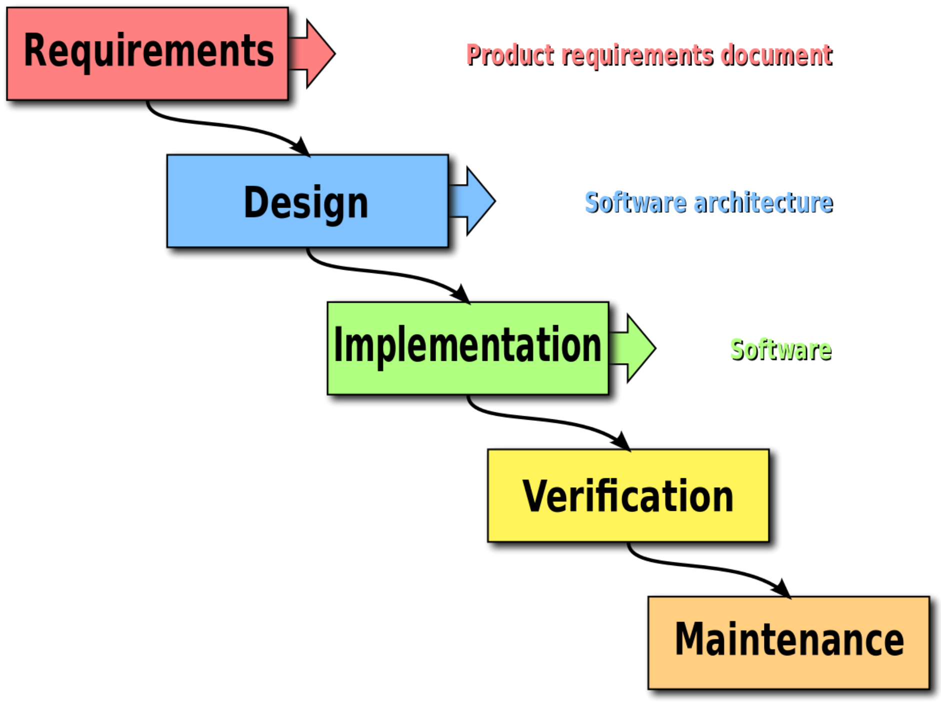


Figure 5 Waterfall model

System Requirements Phase

The purpose of this phase is to determine project's main
goal and how the system will function. To gather system requirements information, these are common questions that have to be answered:

Why the system needs to be developed?
Who are the users?
How will they use the system?
What are they using the system for?
What are the input and output of the system?

This question needs to be answered thoroughly to come up with clear functionality of the system describing the functions that the system should perform. All possible requirements of the system to be developed are captured in this phase. After the requirements are understood. Software Requirement Specification (SRS) document is prepared to serve as a guideline for the next phase of the model.

Design Phase:

This is the plan of how the system will look like and how it works. It describes the desired features and operations in detail and may include screen layouts, process diagrams, pseudocode and other documentation. A sample of the project is developed in this phase.

Design focuses on high level design like, what programs are needed and how are they going to interact, low-level design (how the individual programs are going to work),interface design (what are the interfaces going to look like) and data design (what data will be required).

During these phases, the software's overall structure is defined and the logical system of the product is developed in this phase. It also helps in specifying hardware and system requirements and also helps in defining overall system architecture.

Implementation and Unit Testing Phase

This phase is considered to be the longest phase of the software development life cycle. This is so because this is where the code is created and work is divided into small programs that are referred to as units.

This phase include unit testing whereby units will be tested individually for their functionality before the whole system. Unit testing mainly verifies if the modules also known as units meet project specifications.

3.4 SOFTWARE AND REQUIREMENTS

1 Hardware Requirements:

1 Raspberry Pi Camera

2 Raspberry Pi model 4

2 Software Requirements:

Python with:

A. **OpenCV**

B. **TensorFlow**

C. **Keras**

D. **PyTorch**

A. OpenCV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc.

OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 18 million. The library is used extensively in companies, research groups and by governmental bodies.Along with well-established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota that employ the library, there are many startups such as Applied Minds, Video Surf, and Zeitera, that make extensive use of OpenCV. OpenCV's deployed uses span the range from stitching street view images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan.

It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. A full-featured CUDA and OpenCL interfaces are being actively developed right now. There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms. OpenCV is written natively in C++ and has a template interface that works seamlessly with STL containers.

A. TensorFlow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks.

It is used for both research and production at Google, TensorFlow is Google Brain's second-generation system. Version 1.0.0 was released on February 11, While the reference implementation runs on single devices, TensorFlow can run on multiple CPUs and GPUs (with optional CUDA and SYCL extensions for general-purpose computing on graphics processing units).

Tensor Flow is available on 64-bit Linux, macOS, Windows, and mobile computing platforms including Android and iOS. Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices.

The name Tensor Flow derives from the operations that such neural networks perform on multidimensional data arrays, which are referred to as tensors. During the Google I/O Conference in June 2016, Jeff Dean stated that 1,500 repositories on GitHub mentioned TensorFlow, of which only 5 were from Google.Unlike other numerical libraries intended for use in Deep Learning like Theano, TensorFlow was designed for use both in research and development and in production systems, not least RankBrain in Google search and the fun DeepDream project.It can run on single CPU systems, GPUs as well as mobile devices and large scale distributed systems of hundreds of machines.

B. Keras

Keras is an API designed for human beings, not machines. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear & actionable error messages.

It also has extensive documentation and developer guides. Keras contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code.

The code is hosted on GitHub, and community support forums include the GitHub issues page, and a Slack channel. Keras is a minimalist Python library for deep learning that can run on top of Theano or Tensor Flow.

It was developed to make implementing deep learning models as fast and easy as possible for research and development. It runs on Python 2.7 or 3.5 and can seamlessly execute on GPUs and CPUs given the underlying frameworks.

It is released under the permissive MIT license. Keras was developed and maintained by François Chollet, a Google engineer using four guiding principles:

 Modularity: A model can be understood as a sequence or a graph alone. All the concerns of a deep learning model are discrete components that can be combined in arbitrary ways.
 Minimalism: The library provides just enough to achieve an outcome, no frills and maximizing readability.
 Extensibility: New components are intentionally easy to add and use within the framework, intended for researchers to trial and explore new ideas.
 Python: No separate model files with custom file formats. Everything is native Python. Keras is designed for minimalism and modularity allowing you to very quickly define deep learning models and run them on top of a Theano or TensorFlow backend.

C. PyTorch

PyTorch is an open source machine learning library based on the Torch library, used for applications such as computer vision and natural language processing, primarily developed by Face book’s AI Research lab (FAIR).

It is free and open-source software released under the Modified BSD license. Although the Python interface is more polished and the primary focus of development, PyTorch also has a C++ interface. Tensor computing (like NumPy) with strong acceleration via graphics processing units (GPU).

- Deep neural networks built on a tape-based automatic differentiation system

PyTorch defines a class called Tensor (torch.Tensor) to store and operate on homogeneous multidimensional rectangular arrays of numbers. PyTorch Tensors are similar to NumPy Arrays, but can also be operated on a CUDA-capable Nvidia GPU. PyTorch supports various sub-types of Tensors.

Figure 6 logos of OpenCV, TensorFlow, Keras , PyTorch

3.5 PRELIMINARY PRODUCT DESCRIPTION

The face mask recognition system uses AI technology to detect the person with or without a mask. It can be connected with any surveillance system installed at your premise. The authorities or admin can check the person through the system to confirm their identity.

The system sends an alert message to the authorized person if someone has entered the premise without a face mask.

The accuracy rate of detecting a person with a face mask is 95-97% depending on the digital capabilities. The data has been transferred and stored automatically in the system to enable reports whenever you want.

Features of the Face Mask Detection System

- The system is easy to implement in any existing organizational system
- Custom alerts can be sent to the person with or without a face mask or the one whose face is unrecognizable in the admin system.
- The system can be used easily with any camera or hardware like surveillance cameras.
- The system restricts access for those not wearing the masks and notifies the authorities.
- You can customize the face mask detection system based on your business requirements.
- Easy to access and control the movements from any device through face mask detection applications.

CHAPTER 4 SYSTEM DESIGNS

4.1 Basic Modules

4.1.1 Activity Diagram

The proposed system helps to ensure the safety of the people at public places by automatically monitoring them whether they maintain a safe social distance, and also by detecting whether or not and individual wears face mask.

This section briefly describes the solution architecture and how the proposed system will automatically functions in an automatic manner to prevent the coronavirus spread.

The system uses a transfer learning approach to performance optimization with a deep learning algorithm and a computer vision to automatically monitor people in public places with a camera integrated with a raspberry pi4 and to detect people with mask or no mask.

We also do fine tuning, which is another form of transfer learning, more powerful than just the feature extraction.

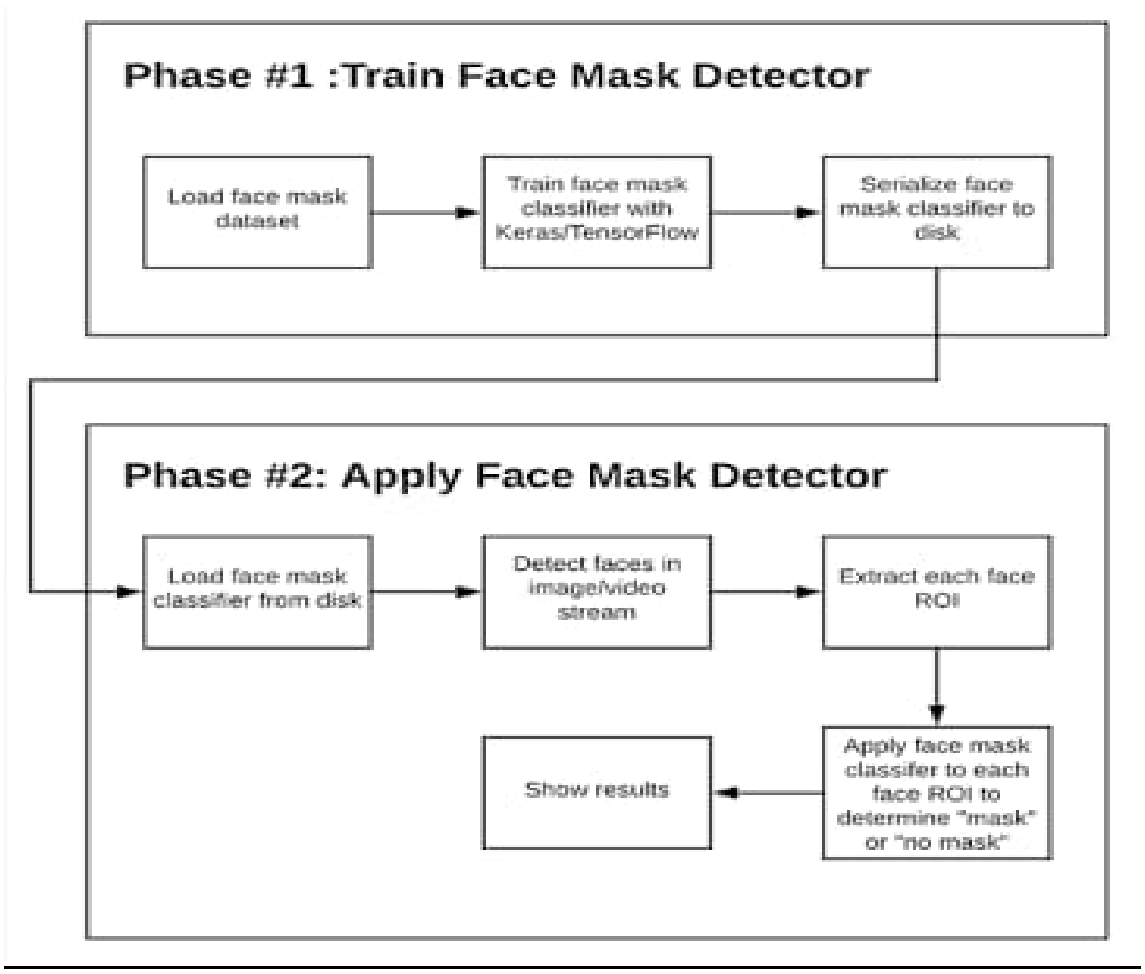

Phase #1 :Train Face Mask Detector
Load face mask dataset
Train face mask classifier with Keras/TensorFlow
Serialize face mask classifier to disk
Phase #2: Apply Face Mask Detector
Load face mask classifier from disk
Detect faces in image/video stream
Extract each face ROI
Apply face mask classifer to each face ROI to determine "mask" or "no mask"
Show results

Figure 7 4.1.1 Activity Diagram

4.2 Data Design

4.2.1 E-R Diagram

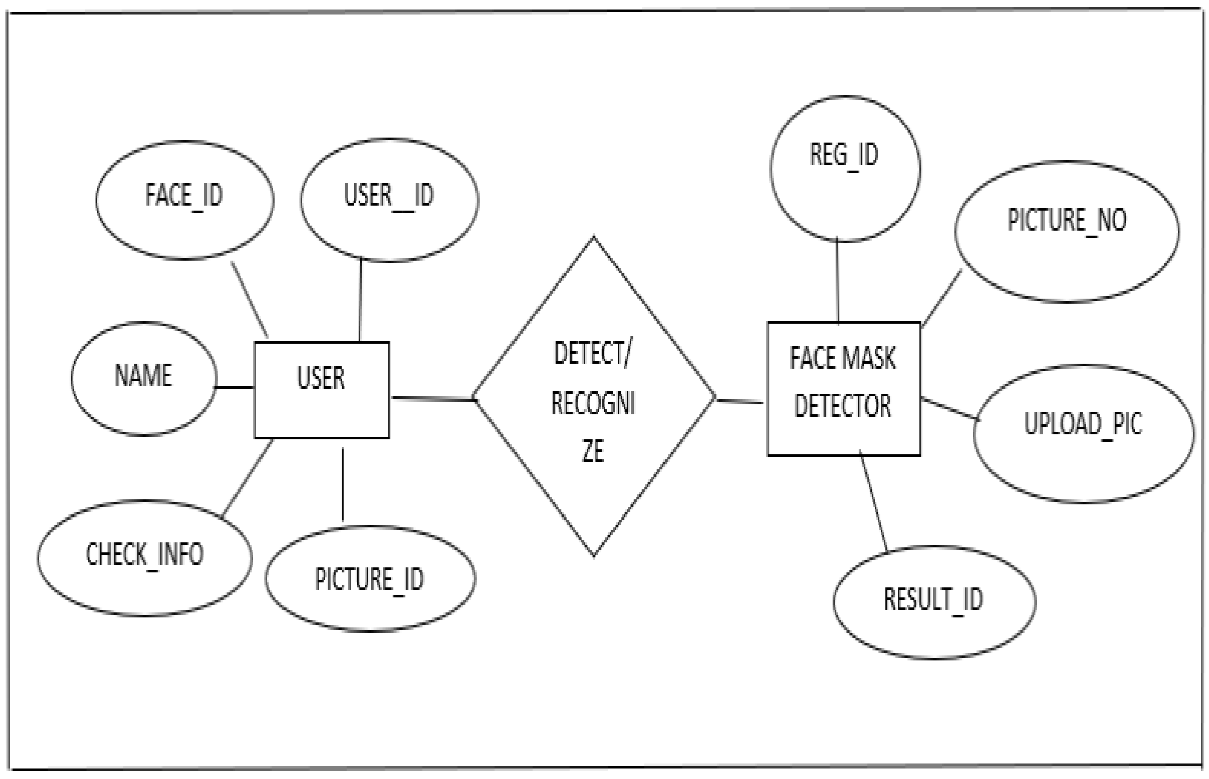


Figure 8 4.2.1 E-R diagram

4.2.2 Sequence Diagram

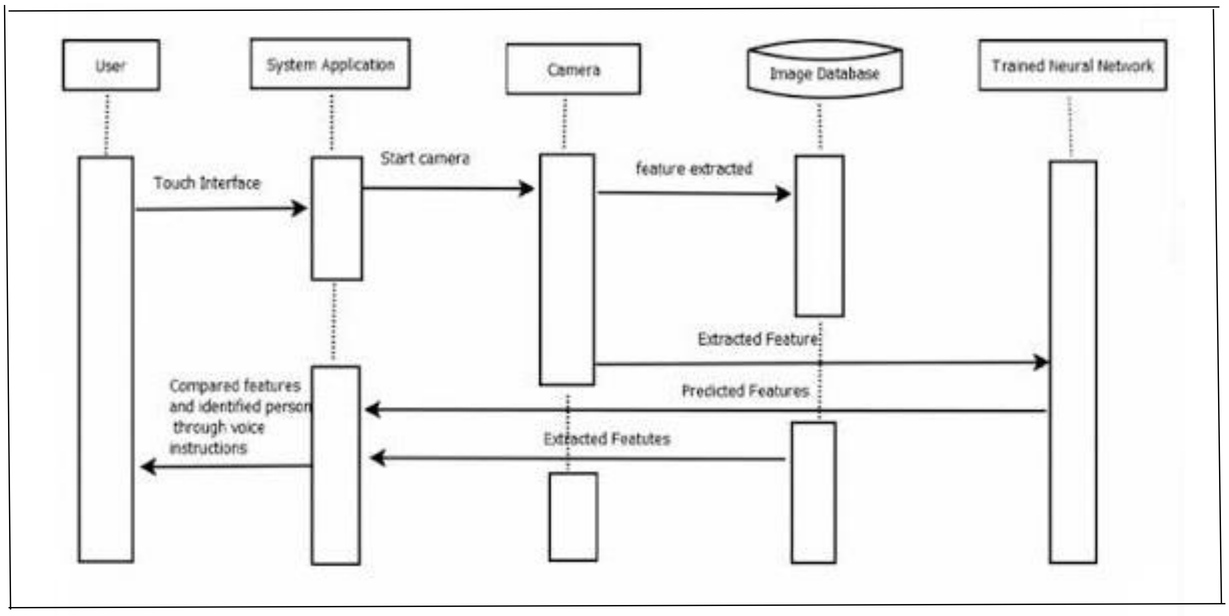


Figure 9 4.2.2 Sequence Diagram

4.2.3 Data Flow Diagram

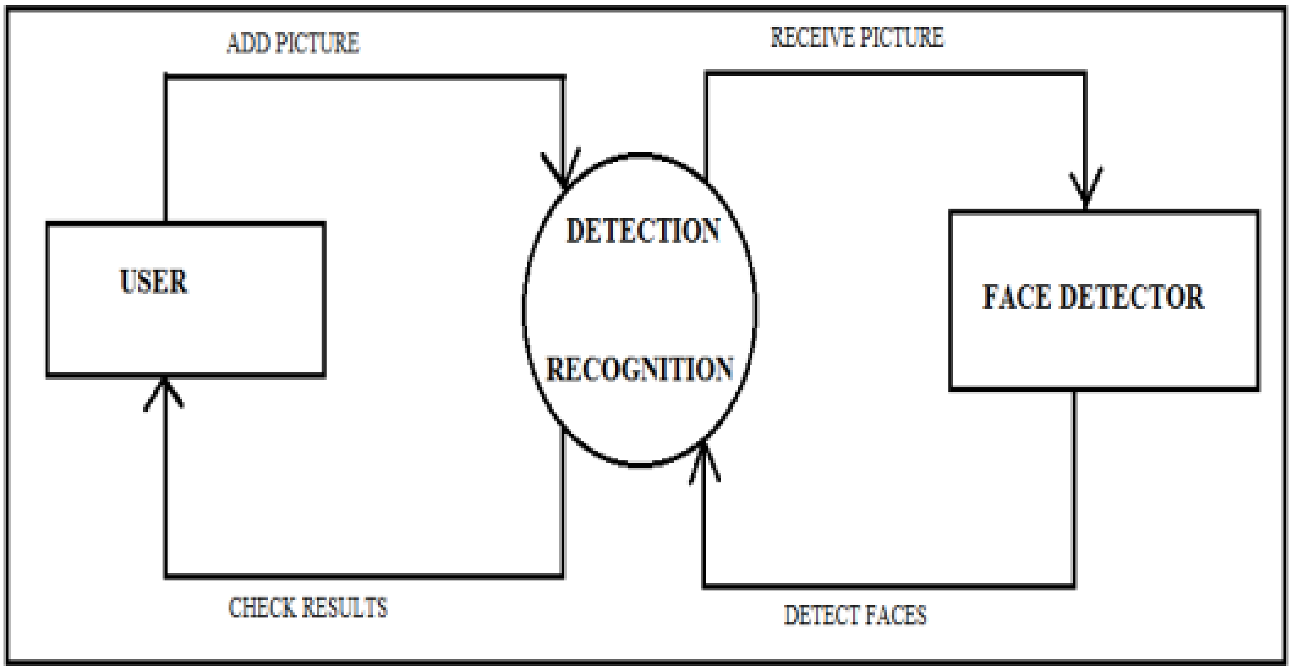
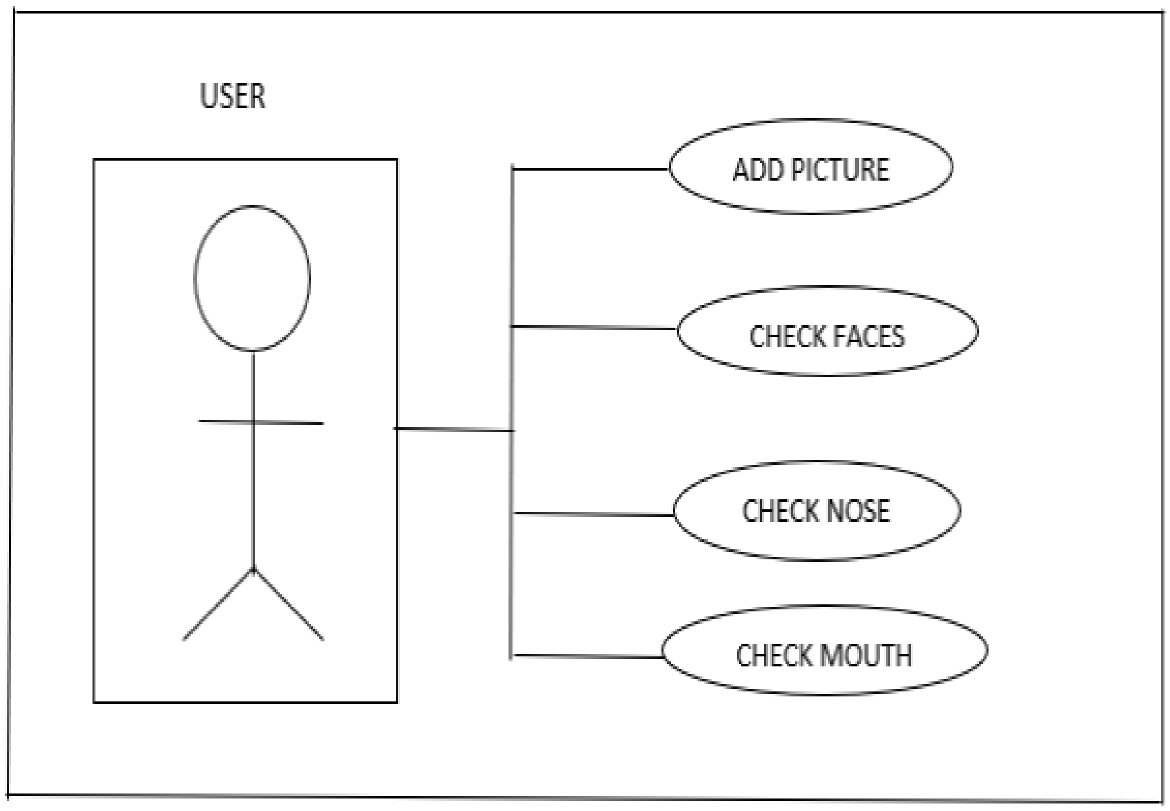


Figure 10 4.2.3 Data Flow Diagram

4.2.4 Use Case Diagram

User Module:



Software Module:

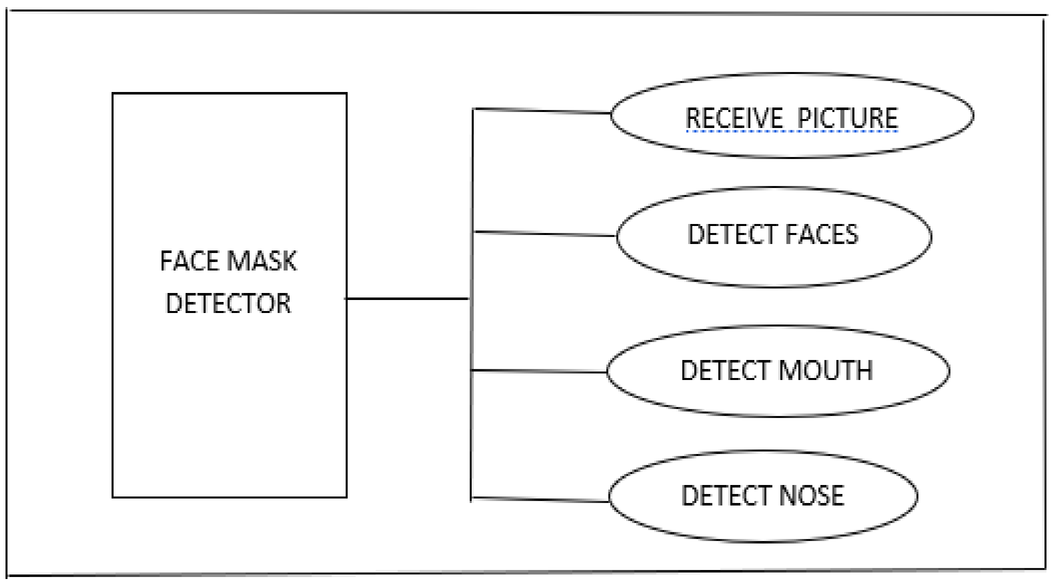


Figure 11 4.2.4 Use Case diagram

4.2.5 Fundamental steps in image processing

Fundamental steps in image processing are

1. Image acquisition: to acquire a digital image

2. Image pre-processing: to improve the image in ways that increases the chances for success of the other

processes.

3. Image segmentation: to partitions an input image into its constituent parts of objects.

4. . Image description: to extract the features that result in some quantitative information of interest of

features that are basic for differentiating one class of objects from another.

5. Image recognition: to assign a label to an object based on the information provided by its description.

6. Image segmentation: to convert the input data to a from suitable for computer processing.

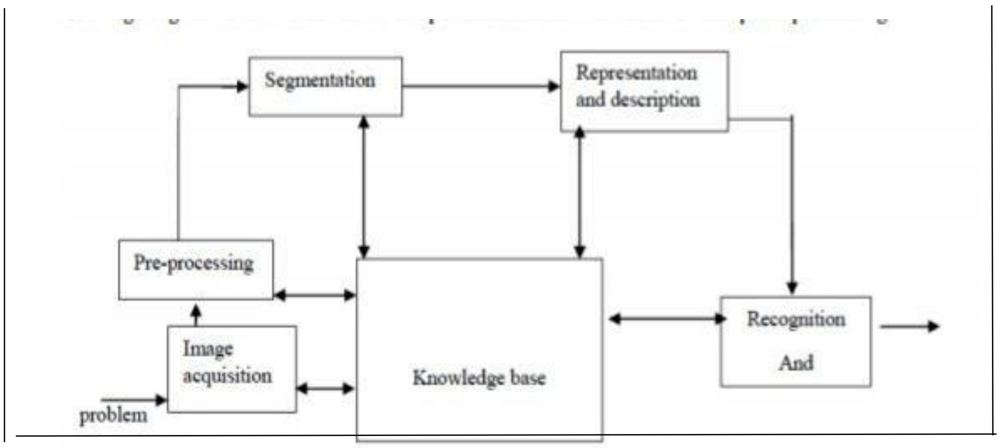


Figure 12 4.2.5 Fundamental steps in image processing

4.3.10 Security Issues

Nowadays, deep learning is becoming increasingly important in our daily life. The appearance of deep learning in many applications in life relates to prediction and classification such as self-driving, product recommendation, advertisements and healthcare.

Therefore, if a deep learning model causes false predictions and misclassification, it can do great harm. This is basically a crucial issue in the deep learning model.

In addition, deep learning models use large amounts of data in the training/learning phases, which contain sensitive information. Therefore, when deep learning models are used in real-world applications, it is required to protect the privacy information used in the model. In this article, we carry out a brief review of the threats and defenses methods on security issues for the deep learning models and the privacy of the data used in such models while maintaining their performance and accuracy.

References And Bibliography

Brunelli, R. and Poggio, T. (1993), Face Recognition: Features versus Templates. IEEE Transactions on Pattern Analysis and Machine Intelligence, 15(10):1042-1052

• AAFPRS(1997). A newsletter from the American Academy of Facial Plastic and Reconstructive Surgery. Third Quarter 1997, Vol. 11, No. 3. Page 3.

• Baron, R. J. (1981). Mechanisms of human facial recognition. International Journal of Man Machine Studies, 15:137-178

• Beymer, D. and Poggio, T. (1995) Face Recognition From One Example View, A.I. Memo No. 1536, C.B.C.L. Paper No. 121. MIT

• Bichsel, M. (1991). Strategies of Robust Objects Recognition for Automatic Identification of Human Faces. PhD thesis, , Eidgenossischen Technischen Hochschule, Zurich.

• Brennan, S. E. (1982) The caricature generator. M.S. Thesis. MIT.

• Brunelli, R. and Poggio, T. (1993), Face Recognition: Features versus Templates. IEEE Transactions on Pattern Analysis and Machine Intelligence, 15(10):1042-1052

• Craw, I., Ellis, H., and Lishman, J.R. (1987). Automatic extraction of face features. Pattern Recognition Letters, 5:183-187, February.

. M.Hunke and A. Waibel, “Face Locating And Tracking for Human Computer

Interaction”,in Proc. Conf. Signals Systems and Computers, Nov 1994, vol2 pp 1277-

1281.

. D. Cahi and K.N. Ngan, “Face segmentation using skin color map”, IEEE Trans. On

Circuit and Systems for Video Technology, vol 9, no 4, pp. 551-564, Jun. 1999.

Abbas El Gamal, “Introduction To Statistical Signal Processing”, Lecture Notes.

Richard Duda, Peter Hart and David Stork, “Pattern Classification”, 2nd ed., John Wiley and Sons.

Sirovich, L. and Kirby, M.: 1987, Low dimensional procedure for the characterization of human faces, J. Opt. Soc. Am. A 4, 519–524.

Summery

As the technology are blooming with emerging trends the availability so we have novel face mask detector which can possibly contribute to public healthcare.

The architecture consists of Mobile Net as the backbone it can be used for high and low computation scenarios. In order to extract more robust features, we utilize transfer learning to adopt weights from a similar task face detection, which is trained on a very large dataset.

We used OpenCV, tensor flow, keras , Pytorch and CNN to detect whether people were wearing face masks or not. The models were tested with images and real-time video streams.

The accuracy of the model is achieved and, the optimization of the model is a continuous process and we are building a highly accurate solution by tuning the hyper parameters.

This specific model could be used as a use case for edge analytics. Furthermore, the proposed method achieves state-of-the-art results on a public face mask dataset.

By the development of face mask detection we can detect if the person is wearing a face mask and allow their entry would be of great help to the society.