**REAL-TIME FACE MASK DETECTION**

**USING MACHINE LEARNING**

*A project report submitted in partial fulfilment of the requirement for the award of the degree of*

**BACHELOR OF TECHNOLOGY**

**In**

**COMPUTER SCIENCE AND ENGINEERING**

Submitted by

V. Krishna Vamsi -5191411059

Zaheen Ansar -5191411061

Y. Anu Abhishek -5191411060

Under the Esteemed Guidance of

**Dr. B h. Padma *Assistant professor***



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**ENGINEERING AND TECHNOLOGY PROGRAM**

**GAYATRI VIDYA PARISHAD COLLEGE FOR DEGREE AND PG COURSES (A)**

**Rushikonda, Visakhapatnam - 45**

(Approved by AICTE| Accredited by NBA| Accredited by NAAC| Affiliated to Andhra University)

An ISO 9001:2015 Certified Institution

**2019 – 2023**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**GAYATRI VIDYA PARISHAD COLLEGE FOR DEGREE AND PG COURSES (A)**

**Rushikoda, Visakhapatnam - 45**



**CERTIFICATE**

|  |  |
| --- | --- |
| **Project Guide** | **Head of the Department** |
| D r. B h. Padma | D r.N. V. Ramana Murthy |
| Assistant Professor | Professor |
| Degrees | M. Tech, Ph. D |
|  |  |

This is to certify that the project report entitled “REAL-TIME FACE MASK DETECTION USING MACHINE LEARNING” being submitted by V. Krishna Vamsi (5191411059), Zaheen Ansar (5191411061), Y. Anu Abhishek (5191411060) in the partial fulfilment for the award of the Degree of Bachelor of Technology in Computer Science and Engineering to the Gayatri Vidya Parishad College for Degree and PG Courses, Visakhapatnam is a record of bonified work carried out under my guidance and supervision.

**Internal Examiner External Examiner**

**VISION AND MISSION OF THE INSTITUTE**

**VISION**

**Creating Human Excellence for a Better Society.**

**MISSION**

**Unfold into a World class organization with strong academic and research base, producing responsible citizens to cater to the changing needs of the society.**

**VISION AND MISSION OF THE DEPARTMENT**

**VISION**

**“Create and sustain as a Centre of excellence in Computer Science and Engineering and allied areas of research to pave the way for providing better technocrats to the society.”**

**MISSION**

**“To provide strong *conceptual base (M1)* for acquiring programming skills, to develop high *computational innovative skills (M2)* and to empower *leadership qualities* with strong *ethical base (M3)”.***

**PROGRAM EDUCATIONAL OBJECTIVES (PEOs)**

**PEO1: Ability to develop hardware and software problem solving capabilities by inculcating strong fundamental, analytical, and communication skills.**

**PEO2: Ability to build confidence to address the challenges in their work and to apply innovative technical tools through interaction with professional bodies.**

**PEO3: Ability to gain sufficient depth of knowledge, Self-learning skills, technical skills and leadership qualities through participating in personality development activities, so as to sustain in multidisciplinary environment.**

**PEO4: To be able to respond to societal needs in professional and ethical concerns.**

**PROGRAM SPECIFIC OUTCOMES (PSOs)**

**PSO 1: System Inception and Elaboration: Conceptualize the software and/or hardware systems, system components and processes/procedures through requirement analysis, Modelling/Design of the system using various architectural/design patterns, Standard Notations, procedures and algorithms.**

**PSO 2: System Construction: Implement the systems, Procedures and Processes using the state of, the art technologies, standards, tools and Programming Paradigms.**

**PSO 3: System Testing and Deployment: Verify and Validate the Systems, Procedures and Processes using various testing and verification techniques and tools.**

**PSO 4: Quality and Maintenance: Manage the quality through various product development strategies under revision, transition and operation through maintainability, flexibility, testability, portability, reusability, interoperability, correctness, reliability, efficiency, integrity and usability to adapt the system to the changing structure and behaviour of the systems/environments.**

**DECLARATION**

We hereby declare that the project entitled **“REAL-TIME FACE MASK DETECTION USING MACHINE LEARNING”** submitted in partial fulfilment of the requirements for the award of Bachelor of Technology in Computer Science and Engineering, to Engineering and Technology Program, Gayatri Vidya Parishad College for Degree & PG Courses (A). We assure that this project is not submitted in any other University or College.

**Name & Signature of the Students**

V. Krishna Vamsi -5191411059

Zaheen Ansar -5191411061

Y. Anu Abhishek -5191411060

**ACKNOWLEDGEMENTS**

With great pleasure we want to take this opportunity to express our heartfelt gratitude to all the people who helped in making project work a grand success.

First of all, we express our deep sense of gratitude to Smt. D r. B h. Padma, Assistant Professor, for his constant guidance throughout our project work.

We are grateful to DR. B h. Padma, Assistant Professor,for his valuable suggestions and guidance given by him during the execution of this project.

We would like to thank D r.N. V. Ramana Murthy, Professor, Head of the Department of Computer Science and Engineering, for being moral support throughout the period of our study.

We are highly indebted to D r.B. V. Ramana Murthy, Director and Prof. S. Rajani, Principal for giving us the permission to carry out this project in the college.

We would like to thank all the Teaching and Non-Teaching staff of CSE Department for sharing their knowledge with us.

V. Krishna Vamsi -5191411059

Zaheen Ansar -5191411061

Y. Anu Abhishek -5191411060

**ABSTRACT**

COVID-19 pandemic has rapidly affected our day-to-day life disrupting the world trade and movements. Wearing a protective face mask has become a new normal. In the near future, many public service providers will ask the customers to wear masks correctly to avail of their services. Therefore, face mask detection has become a crucial task to help global society. This project presents a simplified approach to achieve this purpose using some basic Machine Learning and Deep Learning packages. The proposed method detects the face from the image correctly and then identifies if it has a mask on it or not. As a surveillance task performer. The method attains accuracy up to 95.77% and 94.58% respectively on two different datasets. We explore optimized values of parameters using the Sequential Convolutional Neural Network model to detect the presence of masks correctly without causing over-fitting.

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| **CHAPTER** | **Page No** |
| 1. Introduction  1.1 Introduction to machine learning  1.2 Introduction to OpenCV  1.3 Applications  1.4 Introduction to Face Mask Detection | 10  10  11  12  13 |
| 2. Literature Survey | 15 |
| 3. Requirements and Analysis  3.1 Existing System  3.2 Proposed System  3.3 Functional Requirements  3.4 Non-Functional Requirements  3.5 Analysis | 16  16  18  18  19  20 |
| 4. System Design  4.1 Design Goals  4.2 System Architecture  4.3 System Design  4.4 UML Diagrams  4.5 Data Flow Diagrams  4.6 Flow Chart | 22  22  22  23  23  27  28 |
| 5. Implementation  5.1 Coding Approach  5.2 Information Handling  5.3 Programming Style  5.4 Verification and Validation  5.5 Implementation of Project  5.6 Hardware Implementation of  Project | 30  30  30  30  31  31  34 |
| 6. Testing  6.1 Testing Activities  6.2 Testing Type | 35  35  35 |
| 7. Results  7.1 Start webcam  7.2 Face detection with mask  7.3 Face detection without mask  7.4 Input and Output | 38  38  38  39  39 |
| 8. Sample Code | 40 |
| 9. Conclusion | 44 |
| 10. References | 45 |

**LIST OF FIGURES**

|  |  |
| --- | --- |
| **Title** | **Page No** |
| 3.4.2 Software Specifications  4.3.1 System’s Block Diagram | 20 |
| 4.4.1 Use Case Diagram | 24 |
| 4.4.2 Class Diagram | 25 |
| 4.4.3 Activity Diagram | 26 |
| 4.4.4 Sequence Diagram  4.5.1 DFD Level 0  4.5.2 DFD Level 1  4.6.1 Flow chart of the system | 27  27  28  29 |
| 6.2.1 Black Box Testing | 37 |
| 7.1 Start webcam | 38 |
| 7.2 Face mask detection with mask | 38 |
| 7.3 Face mask detection without mask | 39 |
| 7.4 Input and Output  9. Conclusion | 39  44 |
|  |  |

**1.INTRODUCTION**

**1.1 Introduction to Machine Learning**

The term Machine Learning was coined by Arthur Samuel in 1959, an American pioneer in the field of computer gaming and artificial intelligence, and stated that “it gives computers the ability to learn without being explicitly programmed”.   
And in 1997, Tom Mitchell gave a “well-posed” mathematical and relational definition that “A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

Machine Learning is the latest buzzword floating around. It deserves to, as it is one of the most interesting subfields of Computer Science. So what does Machine Learning really mean?

Let’s try to understand Machine Learning in layman’s terms. Consider you are trying to toss a paper into a dustbin.

After the first attempt, you realize that you have put too much force into it. After the second attempt, you realize you are closer to the target but you need to increase your throw angle. What is happening here is basically after every throw we are learning something and improving the end result. We are programmed to learn from our experience.

This implies that the tasks in which machine learning is concerned to offer a fundamentally operational definition rather than defining the field in cognitive terms. This follows Alan Turing’s proposal in his paper “Computing Machinery and Intelligence”, in which the question “Can machines think?” is replaced with the question “Can machines do what we (as thinking entities) can do?”   
Within the field of data analytics, machine learning is used to devise complex models and algorithms that lend themselves to prediction; in commercial use, this is known as predictive analytics. These analytical models allow researchers, data scientists, engineers, and analysts to “produce reliable, repeatable decisions and results” and uncover “hidden insights” through learning from historical relationships and trends in the data set(input).

So if you want your program to predict, for example, traffic patterns at a busy intersection (task T), you can run it through a machine learning algorithm with data about past traffic patterns (experience E) and, if it has successfully “learned”, it will then do better at predicting future traffic patterns (performance measure P).   
The highly complex nature of many real-world problems, though, often means that inventing specialized algorithms that will solve them perfectly every time is impractical, if not impossible. Examples of machine learning problems include, “Is this cancer?”, “Which of these people are good friends with each other?”, “Will this person like this movie?” such problems are excellent targets for Machine Learning, and in fact, machine learning has been applied to such problems with great success.

* 1. **Introduction to OpenCV**

OpenCV stands for Open supply pc Vision Library is associate open supply pc vision and machine learning software system library. The purpose of creation of OpenCV was to produce a standard infrastructure for computer vision applications and to accelerate the utilization of machine perception within the business product [6]. It becomes very easy for businesses to utilize and modify the code with OpenCV as it is a BSD-licensed product. It is a rich wholesome library as it contains 2500 optimized algorithms, which also includes a comprehensive set of both classic and progressive computer vision and machine learning algorithms. These algorithms is used for various functions such as discover and acknowledging faces. Identify objects classify human actions. In videos, track camera movements, track moving objects. Extract 3D models of objects, manufacture 3D purpose clouds from stereo cameras, sew pictures along to provide a high-resolution image of a complete scene, find similar pictures from a picture information, remove red eyes from images that are clicked with the flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality. Officially launched in 1999 the OpenCV project was initially an Intel Research initiative to advance CPU-intensive applications, part of a series of projects including real-time ray tracing and 3D display walls The main contributors to the project included a number of optimization experts in Intel Russia, as well as Intel's Performance Library Team. In the early days of OpenCV, the goals of the project were described as:

• Advance vision research by providing not only open but also optimized code for basic vision infrastructure. No more reinventing the wheel.

• Disseminate vision knowledge by providing a common infrastructure that developers could build on, so that code would be more readily readable and transferable.

• Advance vision-based commercial applications by making portable, performance-optimized code available for free – with a license that did not require code to be open or free itself.

**1.3. Applications**

Machine learning is a buzzword for today's technology, and it is growing very rapidly day by day. We are using machine learning in our daily life even without knowing it such as Google Maps, Google assistant, Alexa, etc. Below are some most trending real-world applications of Machine Learning:

1. Image Recognition:

Image recognition is one of the most common applications of machine learning. It is used to identify objects, persons, places, digital images, etc. The popular use case of image recognition and face detection is, **Automatic friend tagging suggestion**: Facebook provides us a feature of auto friend tagging suggestion. Whenever we upload a photo with our Facebook friends, then we automatically get a tagging suggestion with name, and the technology behind this is machine learning's **face detection** and **recognition algorithm**. It, is based on the Facebook project named "**Deep Face**," which is responsible for face recognition and person identification in the picture

2. Speech Recognition

While using Google, we get an option of "**Search by voice**," it comes under speech recognition, and it's a popular application of machine learning. Speech-recognition is a process of converting voice instructions into text, and it is also known as "**Speech to text**", or "**Computer speech recognition**." At present, machine learning algorithms are widely used by various applications of speech recognition. **Google assistant**, **Siri**, **Cortana**, and **Alexa** are using speech recognition technology to follow the voice instructions.

3. Traffic prediction:

If we want to visit a new place, we take help of Google Maps, which shows us the correct path with the shortest route and predicts the traffic conditions.

It predicts the traffic conditions such as whether traffic is cleared, slow-moving, or heavily congested with the help of two ways:

* **Real Time location** of the vehicle form Google Map app and sensors

**Average time has taken** on past days at the same time.

Everyone who is using Google Map is helping this app to make it better. It takes information from the user and sends back to its database to improve the performance.

**1.4 Introduction to face mask detection**

Face mask detection refers to detect whether a person is wearing a mask or not. In fact, the problem is reverse engineering of face detection where the face is detected using different machine learning algorithms for the purpose of security, authentication and surveillance. Face detection is a key area in the field of Computer Vision and Pattern Recognition. A significant body of research has contributed sophisticated to algorithms for face detection in past. The primary research on face detection was done in 2001 using the design of handcraft feature and application of traditional machine learning algorithms to train effective classifiers for detection and recognition. The problems encountered with this approach include high complexity in feature design and low detection accuracy.

Although numerous researchers have committed efforts in designing efficient algorithms for face detection and recognition but there exists an essential difference between ‘detection of the face under mask’ and ‘detection of mask over face’. As per available literature, very little body of research is attempted to detect mask over face. Thus, our work aims to a develop technique that can accurately detect mask over the face in public areas (such as airports. railway stations, crowded markets, bus stops, etc.) to curtail the spread of Coronavirus and thereby contributing to public healthcare. Further, it is not easy to detect faces with/without a mask in public as the dataset available for detecting masks on human faces is relatively small leading to the hard training of the model. So, the concept of transfer learning is used here to transfer the learned kernels from networks trained for a similar face detection task on an extensive dataset.

**2. Literature survey**

MAMATA S. KALAS, REAL TIME FACE DETECTION AND TRACKING USING OPENCV,

International Journal of Soft Computing and Artificial Intelligence, ISSN: 2321-404X, Volume-2, Issue-1, May- 2014

RELATED WORK

Face detection is defined as the procedure that has many applications like face tracking, pose estimation or compression. Face detection is a two-class problem where we have to decide if there is a face or not in a picture. This approach can be seen as a simplified face recognition problem. Ada-Boost: Ada-boost is an algorithm for constructing a strong classifier as a linear combination. Ada-boost, short for Adaptive Boosting, is a machine learning algorithm. It is a meta-algorithm and can be used in conjunction with many other learning algorithms to improve their performance. Ada-boost is adaptive in the sense that subsequent classifiers built are tweaked in favour of those instances misclassified by previous classifiers. Ada-boost generates and calls a new weak classifier in each of a series of rounds. For from a set of training images. This method can be used for both face detection and face locations. In this method, a standard face (such as frontal) can be used. The advantages of this method are that it is very simple to implement the algorithm, and it is easily to determine the face locations such as nose, eyes, mouth, etc. based on the correlation values.

**3. Requirement Analysis**

**3.1 Existing System**

# 1.Real-Time Implementation of AI-Based Face Mask Detection and Social Distancing Measuring System for COVID-19 Prevention:

**Abstract**

Since the infectious coronavirus disease (COVID-19) was first reported in Wuhan, it has become a public health problem in China and even around the world. This pandemic is having devastating effects on societies and economies around the world. The increase in the number of COVID-19 tests gives more information about the epidemic spread, which may lead to the possibility of surrounding it to prevent further infections. However, wearing a face mask that prevents the transmission of droplets in the air and maintaining an appropriate physical distance between people, and reducing close contact with each other can still be beneficial in combating this pandemic. Therefore, this research paper focuses on implementing a Face Mask and Social Distancing Detection model as an embedded vision system. The pretrained models such as the Mobile Net, Res-Net Classifier, and VGG are used in our context. People violating social distancing or not wearing masks were detected. After implementing and deploying the models, the selected one achieved a confidence score of 100%. This paper also provides a comparative study of different face detection and face mask classification models. The system performance is evaluated in terms of precision, recall, F1-score, support, sensitivity, specificity, and accuracy that demonstrate the practical applicability. The system performs with F1-score of 99%, sensitivity of 99%, specificity of 99%, and an accuracy of 100%. Hence, this solution tracks the people with or without masks in a real-time scenario and ensures social distancing by generating an alarm if there is a violation in the scene or in public places. This can be used with the existing embedded camera infrastructure to enable these analytics which can be applied to various verticals, as well as in an office building or at airport terminals/gates.

# 2.Real-time Face Mask Detection Using Machine Learning/ Deep Feature-Based Classifiers for Face Mask Recognition:

**Abstract**

In this research, we put forward an architecture that combines recent deep learning algorithms with geometry techniques to create robust models that can handle aspects like detection, tracking and validation. This dissertation focuses on creating a model that efficiently utilizes a mix of conventional machine learning and deep learning techniques to categorize the facemasks effectively. Our model comprises two aspects in this hour of need: For dimensionality reduction through feature extraction, the initial element is produced with InceptionV3. The facemask classification procedure is developed with the Logistic Regression (LR) method. Deep learning (DL) is used to efficiently train an architecture using a dataset of photos of people's faces with and without and partial face masks to extract features. The retrieved traits are now passed into various classification algorithms namely Random Forest, Logistic Regression, CNN, Support Vector Machine, AdaBoost and K-Nearest Neighbours to appropriately classify the masks position. Hence, we can project that employing Transfer Learning (TL) and Deep Learning together can detect a properly or improperly worn face mask with high accuracy. This system design stops transmitting this fatal virus by detecting individuals in urban areas who are not wearing facemasks effectively.

**3. Facial Recognition and Face Mask Detection Using Machine Learning Techniques:**

**Abstract**

Facial recognition, as a biometric system, is a crucial tool for the identification procedures. When using facial recognition, an individual's identity is identified using their unique facial features. Biometric authentication system helps in identifying individuals using their physiological and behavioural features. Physiological biometrics utilize human features such as faces, irises, and fingerprints. In contrast, behavioural biometric rely on features that humans do, such as voice and handwritings. Facial recognition has been widely used for security and other law enforcement purposes. However, since COVID-19 pandemic, many people around the world had to wear face masks. This thesis introduces a neural network system, which can be trained to identify people’s facial features while half of their faces are covered by face masks. The Convolutional Neural Network (CNN) model using transfer learning technique has achieved remarkable accuracy even the original dataset is very limited. One large Face mask detection dataset was first used to train the model, while the original much smaller Face mask detector dataset was used to adapt and finetune this model that was previously generated. During the training and testing phases, network structures, and various parameters were adjusted to achieve the best accuracy results for the actual small dataset. Our adapted model was able to achieve a 97.1% accuracy. Keywords: Biometrics, Facial Recognition, Face Mask Detection, CNN, Transfer learning.

**3.2 Proposed system**

The existing problem can be solved by using the computerized detections of face masks. So, if we are able to put down digital cameras in the over populated areas it is much easier to detect whether a person is wearing a mask or not. After the detection then we can implement alarm systems or email systems for asking the individual to wear a mask. The computer vision is in the way of mask detection is a reviving factor to get our lives back on track. Real time mask detection can solve the monitoring issues in geographies with high population.

3.3 **Functional requirements**

Requirements of Face Mask Detector:

R1. The system must be correctly able to load the face mask classifier model.

R2. The system must be able to detect faces in images or video stream.

R3. The system must be able to extract each face’s Region of Interest (ROI).

R4. There must not be any object between the system and the face of the user for a successful face detection and hence the face mask detection.

R5. The end position of the face must be fit inside the webcam frame and must be closer to the camera.

R6. Correctly able to detect masks in ‘png’, ‘jpg’, ‘jpeg’, and ‘gif’ format images.

R7. The system must be able to detect face masks on human faces on every frame in a live video.

R8. The results must be viewed by showing.

3.4 **Non-Functional requirements**

This project is intended to meet the following non-functional requirements:

1) This face recognition software should be available on the Internet, to enable the users to use, download it any time.  
2) The program should be platform independent.

* + 1. **Hardware Specifications**
* Intel Core i3 3rd gen processor or later.
* 512 MB Disk Space
* 512 MB RAM
* Any external or inbuild camera with minimum pixel resolution 200 x 200 (300ppi or 150lpi) 4-megapixel cameras and up.

**3.4.2 Software Specifications**

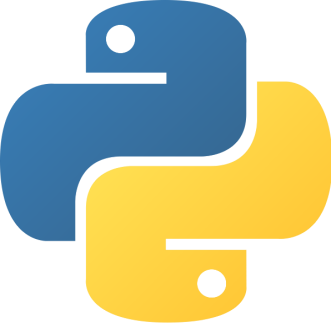
Operating System : Windows 7/ any other

Coding language : Python

**Python**

Python is an interpreted, high-level, general purpose programming language. Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural. It also has a comprehensive standard library.

Python interpreters are available for many operating systems. C, Python, the reference implementation of Python, is open-source software and has a community-based development model, as do nearly all of Python’s other implementations. Python uses dynamic typing, and a combination of reference counting and a cycle-detecting garbage collector for memory management. It also features dynamic name resolution (late binding), which binds method and variable names during program execution.



**3.5 Analysis**

The analysis phase defines the requirements of the system, independent of how these requirements will be accomplished. This phase defines the problem that the customer is trying to solve. The deliverable result at the end of this phase is a requirement document. Analysis object model is represented by class and object diagrams. Analysis focuses on producing a model of the system, called the Analysis model, which is correct, complete, consistent, and verifiable. The analysis model is composed of three individual models: The Functional Model represented by use cases and scenarios, the Analysis Object Model, represented by class and object diagrams, and the Dynamic Model, represented by state chart and sequence diagrams. The analysis model represents the system under development from the user’s point of view. The analysis object model is a part of the analysis and focuses on the individual concepts that are manipulated by the system, their properties and their relationships.

**3.5.1 Entity Objects**

The Analysis object model consists of entity, boundary and control objects. Entity objects represent the persistent information tracked by the system. Participating objects form the basis of the analysis model.

**3.5.2 Boundary Objects**

Boundary object is the object used for interaction between the user and the system. Moreover, it is an interface used to communicate with the system. Boundary objects represent the system interface with the actors. In each use case, each actor interacts with at least one boundary object. The boundary object collects the information from the actor and translates into an interface model from that can be used by the objects and by the control objects.

**3.5.3 Control Objects**

Control objects are responsible for coordinating entity objects and boundary objects. A control object is creating at the beginning of the use cases and ceases to exist at its end. Control objects usually do not have a concrete counterpart in the real world. Control object is a responsible for collecting information from the boundary objects and dispatching it to entity object. Here only authorized users can operate, and details obtained are verified. The control objects in this project are admission, result, placement, company and eligibility details.

**3.5.4 Object Interaction**

Interaction diagrams model the behaviour of use cases by describing the way groups of objects interact to complete the task. The two kinds of interaction diagrams are sequence and collaboration diagrams. Sequence diagrams generally show the sequence of events that occur.

**3.5.5 Object Behaviour**

State chart diagrams are used to describe the behaviour of a system. They define different states of an object during its lifetime. Each diagram usually represents objects of a single class and tracks the different states of its objects through the system and these states are changed by events. State chart diagram describes the flow of control from one state to another state. State chart diagrams are very important for describing the states. States can be identified as the condition of objects when an event occurs.

**4. SYSTEM DESIGN**

System Design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. In System design, developers:

* Define design goals of the project
* Decompose the system into smaller sub systems
* Design hardware/software strategies
* Design persistent data management strategies
* Design global control flow strategies
* Design access control policies and
* Design strategies for handling boundary conditions.

System design is not algorithmic. It is decomposed of several activities. They are:

* Identify Design Goals
* Design the initial subsystem decomposition
* Refine the subsystem decomposition to address the design goals.

System Design is the transform of analysis model into a system design model. Developers define the design goals of the project and decompose the system into smaller subsystems that can be realized by individual teams. Developers also select strategies for building the system, such as the hardware/software platform on which the system will run, the persistent data management strategy, the goal control flow the access control policy and the handling of boundary conditions. The result of the system design is model that includes a clear description of each of these strategies.

**4.1 Design Goals**

Design goals are the qualities that the system should focus on. Many design goals can be inferred from the non-functional requirements or from the application domain.

**User friendly:** The system is user friendly because it is easy to use and understand.

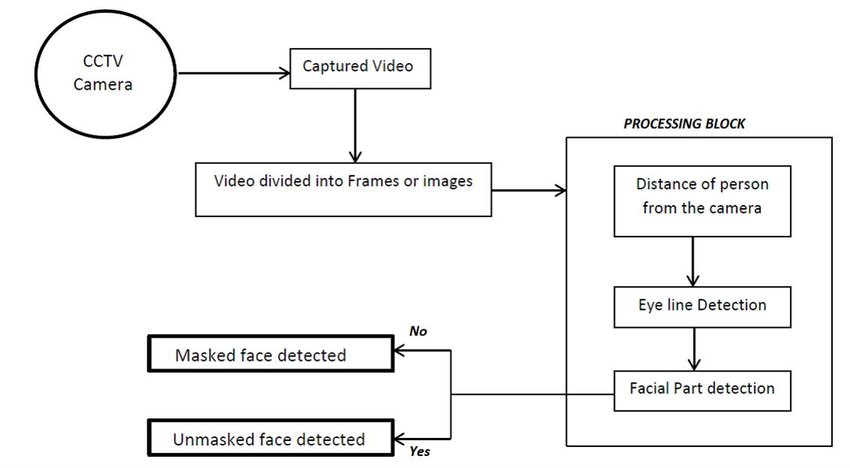
**Reliability:** Proper checks are there for any failure in the system if they exist.

**4.2 System Architecture**

As the complexity of systems increases, the specification of the system decomposition is critical. Moreover, subsystem decomposition is constantly revised whenever new issues are addressed. Subsystems are merged into alone subsystem, a complex subsystem is split into parts, and some subsystems are added to take care of new functionality. The first iterations over the subsystem decomposition can introduce drastic changes in the system design model.

**4.3 System Design**

The overall project aim is taking the first step in flow of this system involves the starting of the camera. Once the camera recognises the hand gesture of the user it will extract the gesture of the hand of the user. If the camera doesn’t recognizes the gesture it will not further proceeded. The extracted hand is transformed into frames and by using the convolutional neural networks algorithm it will predict the sign and display the sign to the user.



**Fig.4.3. System’s Block Diagram**

**4.4 UML Diagrams**

A UML diagram is a diagram based on the UML (Unified Modelling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

UML is a modern approach to modelling and documenting software. In fact, it’s one of the most popular business process modelling techniques. It is based on diagrammatic representations of software components. As the old proverb says: “a picture is worth a thousand words”. By using visual representations, we are able to better understand possible flaws or errors in software or business processes.

Mainly UML has been used as a general-purpose modelling language in the field of software engineering. However, it has row found its way into the documentation of several business processes or workflows. For example, activity diagrams, a type of UML diagram, can be used as a replacement for flowcharts. They provide both a more standardized way of modelling workflows as well as a wider range of features to improves readability.

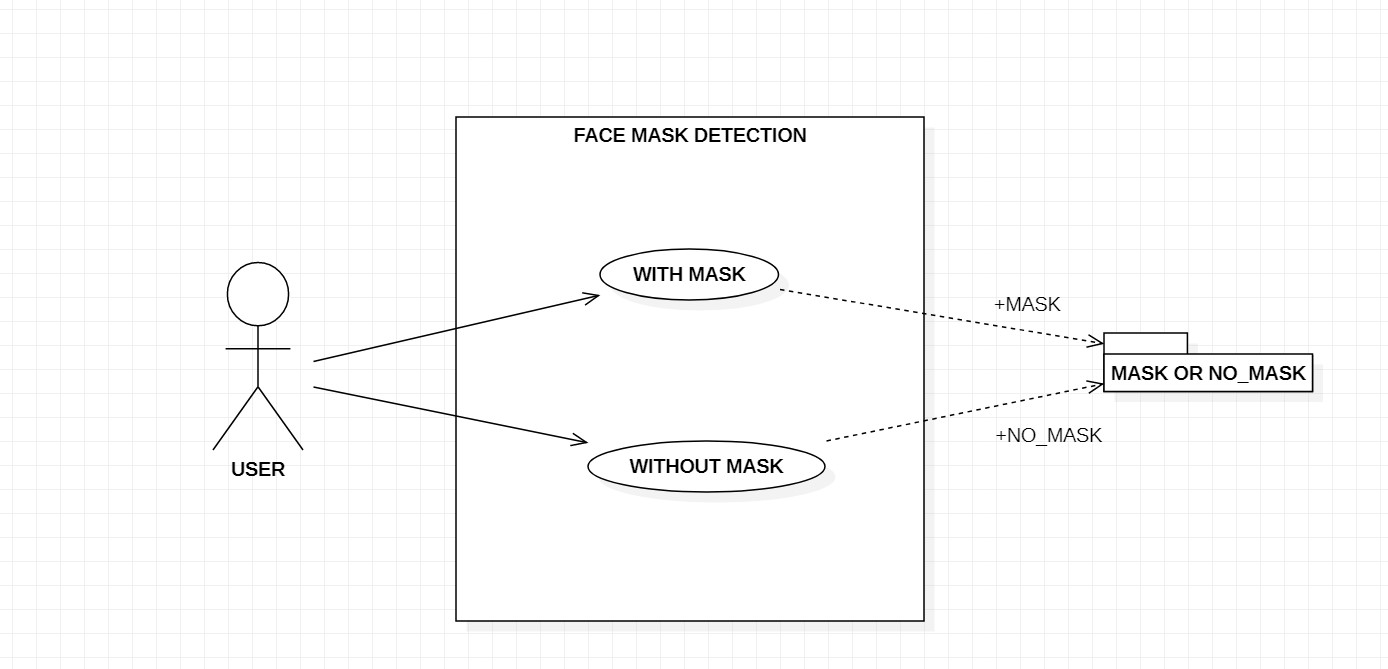
**4.4.1 Use Case Diagram**

In the Unified Modelling Language (UML), a use case diagram can summarize the details of your system’s users (also known as actors) and their interactions with the system. To build one, use a set of specialized symbols and connectors. An effective use case diagram can help the team discuss and represent:

* Scenarios in which system or application interacts with people, organizations, or external systems.
* Goals that your system or application helps those entities (known as actors) achieve.
* The scope of the system.

UML use case diagrams are ideal for:

* Representing the goals of system-user interactions.
* Defining and organizing functional requirements in a system.
* Specifying the context and requirements of a system.
* Modelling the basic flow of events in a use case.



**4.4.2 Class Diagram**

Class diagrams are one of the most useful types of diagrams in UML as they clearly map out the structure of a particular system by modelling its classes, attributes, operations, and relationships between objects. It is a static diagram that represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

The class shape itself consists of a rectangle with three rows. The top row contains the name of the class, the middle row contains the attributes of the class, and the bottom section express the methods or operations that the class may use. Classes and subclasses are grouped together to show the static relationship between each object.

The purpose of the class diagram can be summarized as:

* Analysis and design of the static view of an application.
* Describe responsibilities of a system.
* Base for component and deployment diagrams.
* Forward and reverse engineering.

**4.4.3 Activity Diagram**

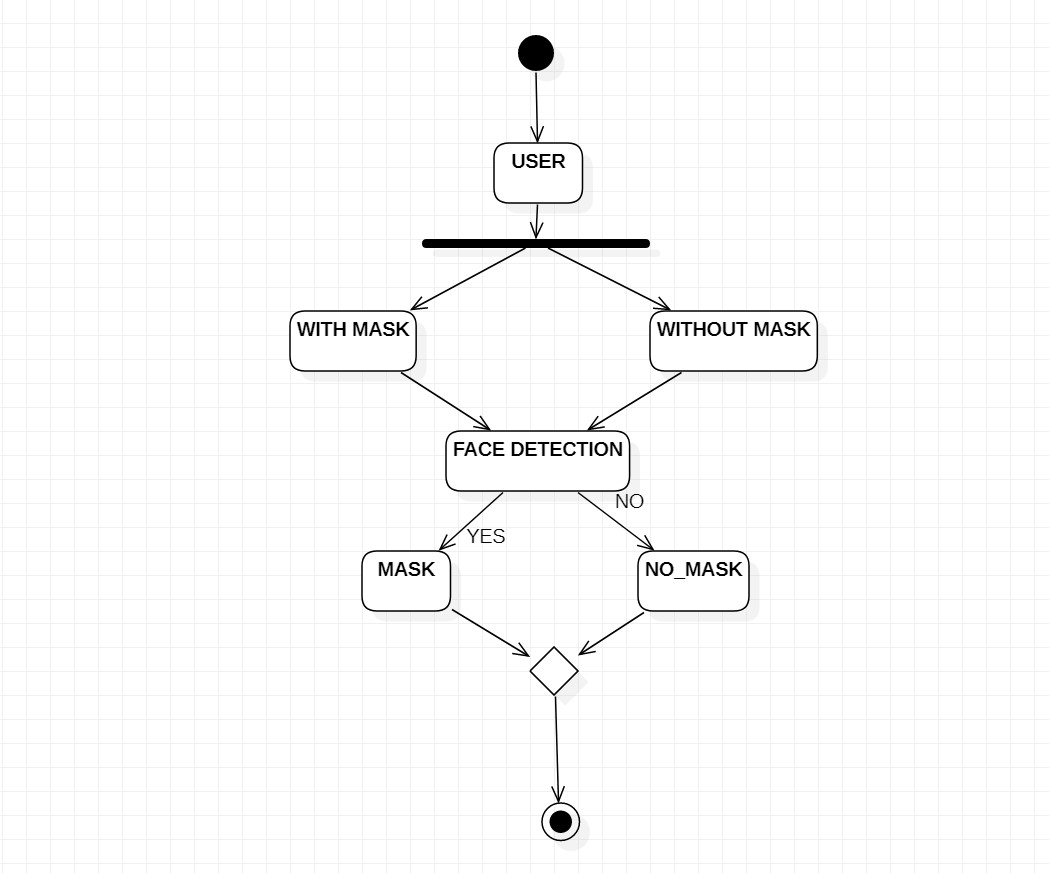
Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc. Activity is a particular

operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques.

The purpose of an activity diagram can be described as

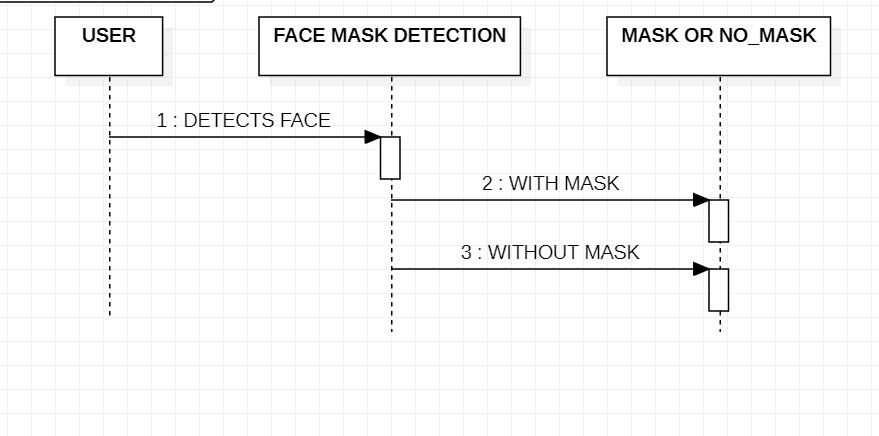
* Draw the activity flow of a system.
* Describe the sequence from one activity to another.
* Describe the parallel, branched and concurrent flow of the system.



**4.4.4 Sequence Diagram**

A sequence diagram is the most commonly used interaction diagram. It simply depicts interaction between objects in a sequential order i.e., the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems. Sequence diagrams can be useful references for businesses and other organizations. Purpose of sequence diagrams are:

* Represent the details of a UML use case.
* Model the logic of a sophisticated procedure, function, or operation.
* See how objects and components interact with each other to complete a process.
* Plan and understand the detailed functionality of an existing or future scenario.

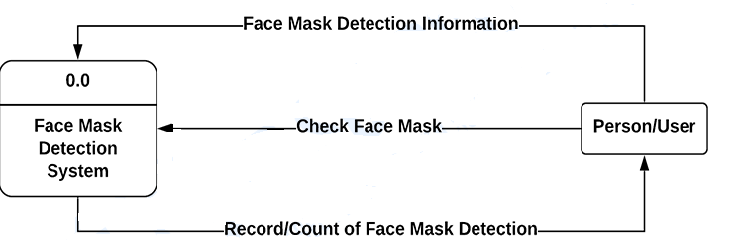


**4.5 Data Flow Diagrams**

* A Data Flow Diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled.

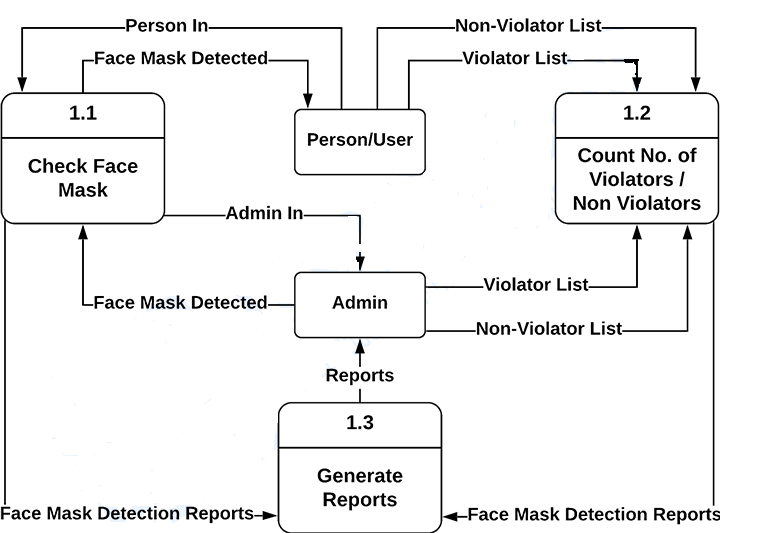
**4.5.1DFD Level-0**

* DFD Level 0 is also called as Context Diagram. It’s a basic overview of the whole system or process being analyzed or modelled. It’s designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities. In our project the external entity is driver

**Fig.4.5.1 DFD Level 0**

**4.5.2 DFD Level-1**

* DFD Level 1 provides a more detailed breakout of pieces of the Context Level Diagram. You will highlight the main functions carried out by the system, as you break down the high-level process of the Context Diagram into the subprocesses.



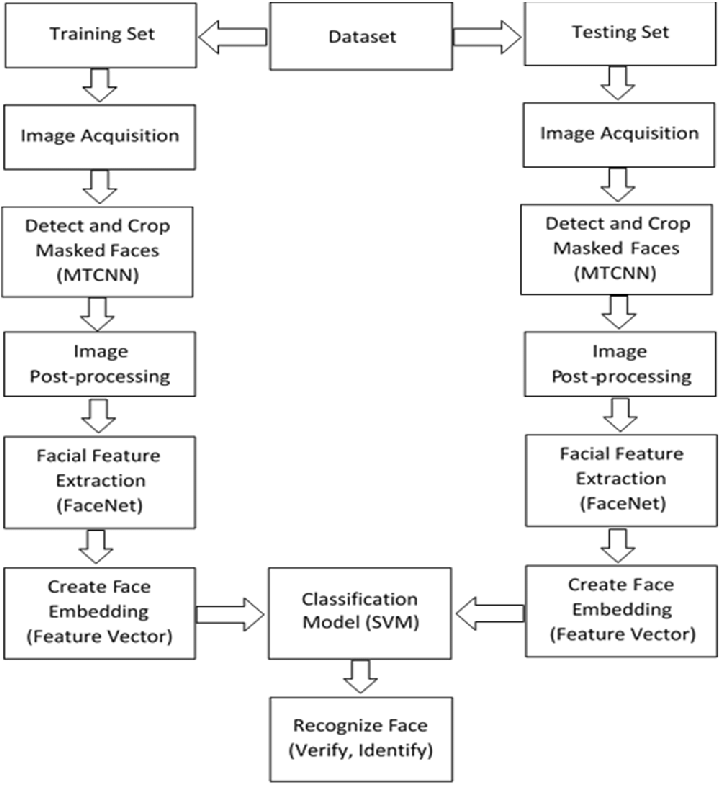
**Fig 4.5.2 DFD Level 1**

**4.6 Flow Chart**

A flowchart is a type of diagram that represents an algorithm, workflow or process. Flowchart can also be defined as a diagrammatic representation of an algorithm. The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analysing, designing, documenting or managing a process or program in various fields.

The first step in flow of this system involves the starting of the camera. Once the camera recognises the hand gesture of the user it will extract the gesture of the hand of the user. If the camera doesn’t recognizes the gesture it will not further proceeded.

The extracted hand is transformed into frames and by using the convolutional neural networks algorithm it will predict the sign and display the sign to the user.



**5. IMPLEMENTATION**

The objective of the coding or programming phase is to translate the design of the system produced during the design phase into code in a given programming language, which can be executed by a computer and that performs the computation specified by the design.

The coding phase affects both testing and maintenance. The goal of coding is not to reduce the implementation cost, but the goal should be to reduce the cost of later phases.

**5.1 Coding Approach**

There are two major approaches for coding any software system. They are Top-Down approach and Bottom-up approach.

Bottom-up Approach can best suit for developing the object-oriented systems. During system design phase, we decompose the system into an appropriate number of subsystems, for which objects can be modelled independently. These objects exhibit the way the subsystems perform their operations.

Once objects have been modelled, they are implemented by means of coding. Even though related to the same system as the objects are implemented of each other, the Bottom-Up approach is more suitable for coding these objects. In this approach, we first do the coding of objects independently and then we integrate these modules into one system to which they belong.

**5.2 Information Handling**

Any software system requires some amount of information during its operation selection of appropriate data structures can help us to produce the code so that objects of the system can better operate with the available information decreased complexity.

**5.3 Programming Style**

Programming style deals with act of rules that a programmer must follow so that the characteristics of coding such as Traceability, Understandability, Modifiability, and Extensibility can be satisfied. In this current system, we followed the coding rules for naming the variables and methods.

**5.4 Verification and Validation**

Verification is the process of checking the product built is right. Validation is the process of checking whether the right product is built. During the Development of the system, Coding for the object has been thoroughly verified from different aspects regarding their design, in the way they are integrated etc. The various techniques that have been followed for validation discussed in testing the current system.

Validations applied to the entire system at two levels:

**Form level Validation:**

Validations of all the inputs given to the system at various points in the forms are validated while navigating to the next form. System raises appropriate custom and pre-defined exceptions to alert the user about the errors occurred or likely to occur.

**Field level Validation:**

Validations at the level of individual controls are also applied wherever necessary. System pops up appropriate and sensuous dialogs wherever necessary. In this project**,** validations are performed on each individual control. If any one of text field is not filled or any wrong, click occurs then system will generate appropriate exceptions.

**5.5 Implementation of Project**

**The Model**

A common job of machine learning algorithms is to recognize objects and being able to separate them into categories. This process is called classification, and it helps us segregate vast quantities of data into discrete values, i.e., distinct, like 0/1, True/False, or a pre-defined output label class.

Before we dive into Classification, let’s take a look at what Supervised Learning is. Suppose you are trying to learn a new concept in mathematics and after solving a problem, you may refer to the solutions to see if you were right or not. Once you are confident in your ability to solve a particular type of problem, you will stop referring to the answers and solve the questions put before you by yourself.

This is also how Supervised Learning works with machine learning models. In Supervised Learning, the model learns by example. Along with our input variable, we also give our model the corresponding correct labels. While training, the model gets to look at which label corresponds to our data and hence can find patterns between our data and those labels.

**Classification**

Classification is defined as the process of recognition, understanding, and grouping of objects and ideas into preset categories a.k.a “sub-populations.” With the help of these pre-categorized training datasets, classification in machine learning programs leverage a wide range of algorithms to classify future datasets into respective and relevant categories.

Classification algorithms used in machine learning utilize input training data for the purpose of predicting the likelihood or probability that the data that follows will fall into one of the predetermined categories. One of the most common applications of classification is for filtering emails into “spam” or “non-spam”, as used by today’s top email service providers.

In short, classification is a form of “pattern recognition,”. Here, classification algorithms applied to the training data find the same pattern (similar number sequences, words or sentiments, and the like) in future data sets.

We will explore classification algorithms in detail, and discover how a text analysis software can perform actions like sentiment analysis - used for categorizing unstructured text by opinion polarity (positive, negative, neutral, and the like).

# Support Vector Machine (SVM)

Support Vector Machine” (SVM) is a supervised machine learning algorithm that can be used for both classification or regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is a number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well (look at the below snapshot).

Support Vector Machine (SVM) is a supervised machine learning algorithm used for both classification and regression. Though we say regression problems as well its best suited for classification. The objective of SVM algorithm is to find a hyperplane in an N-dimensional space that distinctly classifies the data points. The dimension of the hyperplane depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a 2-D plane. It becomes difficult to imagine when the number of features exceeds three.

**SVM Kernel:**

The SVM kernel is a function that takes low dimensional input space and transforms it into higher-dimensional space, i.e., it converts not separable problem to separable problem. It is mostly useful in non-linear separation problems. Simply put the kernel, it does some extremely complex data transformations then finds out the process to separate the data based on the labels or outputs defined.

**Advantages of SVM:**

* Effective in high dimensional cases
* Its memory efficient as it uses a subset of training points in the decision function called support vectors
* Different kernel functions can be specified for the decision functions and its possible to specify custom kernels

**Example:**

SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify it as a cat.

## Types of SVM

**SVM can be of two types:**

* **Linear SVM:** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
* **Non-linear SVM:** Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

**5.6 Hardware Implementation of the Project**

No hardware devices used to implement the project

**6. TESTING**

**6.1 Testing Activities**

Testing is the process of finding differences between the expected behaviour specified by system models and the observed behaviour of the system. Testing is a critical role in quality assurance and ensuring the reliability of development and these errors will be reflected in the code, so the application should be thoroughly tested and validated.

Unit testing finds the differences between the object design model and its corresponding components. Structural testing finds differences between the system design model and a subset of integrated subsystems. Functional testing finds differences between the use case model and the system.

Finally, performance testing, finds differences between non-functional requirements and actual system performance. From modelling point of view, testing is the attempt of falsification of the system with respect to the system models. The goal of testing is to design tests that exercise defects in the system and to reveal problems.

Testing a large system is a complex activity and like any complex activity. It has to be broke into smaller activities. Thus, incremental testing was performed on the project i.e., components and subsystems of the system were tested separately before integrating them to form the subsystem for system testing.

**6.2 Testing Types**

**Unit Testing**

Unit testing focuses on the building blocks of the software system that is the objects and subsystems. There are three motivations behind focusing on components. First unit testing reduces the complexity of overall test activities allowing focus on smaller units of the system, second unit testing makes it easier to pinpoint and correct faults given that few components are involved in the rest. Third unit testing allows parallelism in the testing activities, that is each component are involved in the test. Third unit testing allows parallelism in the testing activities that is each component can be tested independently of one another. The following are some unit testing techniques.

* **Equivalence testing**:

It is a black box testing technique that minimizes the number of test cases.The possible inputs are partitioned into equivalence classes and a test case is selected for each class.

* **Boundary testing:**

It is a special case of equivalence testing and focuses on the conditions atthe boundary of the equivalence classes. Boundary testing requires that the elements be selected from the edges of the equivalence classes.

* **Path testing:**

It is a white box testing technique that identifies faults in the implementation ofthe component the assumption here is that exercising all possible paths through the code at least once. Most faults will trigger failure. This acquires knowledge of source code.

**Integrating Testing**

Integration testing defects faults that have not been detected. During unit testing by focusing on small groups on components two or more components are integrated and tested and once tests do not reveal any new faults, additional components are added to the group. This procedure allows testing of increasing more complex parts on the system while keeping the location of potential faults relatively small. I have used the following approach to implements and integrated testing.

Top-down testing strategy unit tests the components of the top layer and then integrated the components of the next layer down. When all components of the new layer have been tested together, the next layer is selected. This was repeated until all layers are combined and involved in the test.

**Validation Testing**

The systems completely assembled as package, the interfacing have been uncovered and corrected, and a final series of software tests are validation testing. The validation testing is nothing but validation success when system functions in a manner that can be reasonably expected by the customer. The system validation had done by series of Black-box test methods.

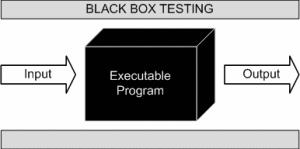
**System Testing**

1. System testing ensures that the complete system compiles with the functional requirements and non-functional requirements of the system, the following are some system testing activities.
2. Functional testing finds differences between the functional between the functional requirements and the system. This is a black box testing technique. Test cases are divided from the use case model.
3. Performance testing finds differences between the design and the system the design goals are derived from the functional requirements.
4. Pilot testing the system is installed and used by a selected set of users – users exercise the system as if it had been permanently installed.
5. Acceptance testing, I have followed benchmarks testing in a benchmark testing the client prepares a set of test cases represent typical conditions under which the system operates. In our project, there are no existing benchmarks.

Installation testing, the system is installed in the target environment.

**Black Box Testing** also known as Behavioural Testing, is a software testing method in which the internal structure/design/implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional. This method is named so because the software program, in the eyes of the tester, is like a black box; inside which one cannot see. This method attempts to find errors in the following categories:

* Incorrect or missing functions
* Interface errors
* Errors in data structures or external database access
* Behavior or performance errors
* Initialization and termination errors



1. **RESULTS**

**7.1 Start webcam**

****

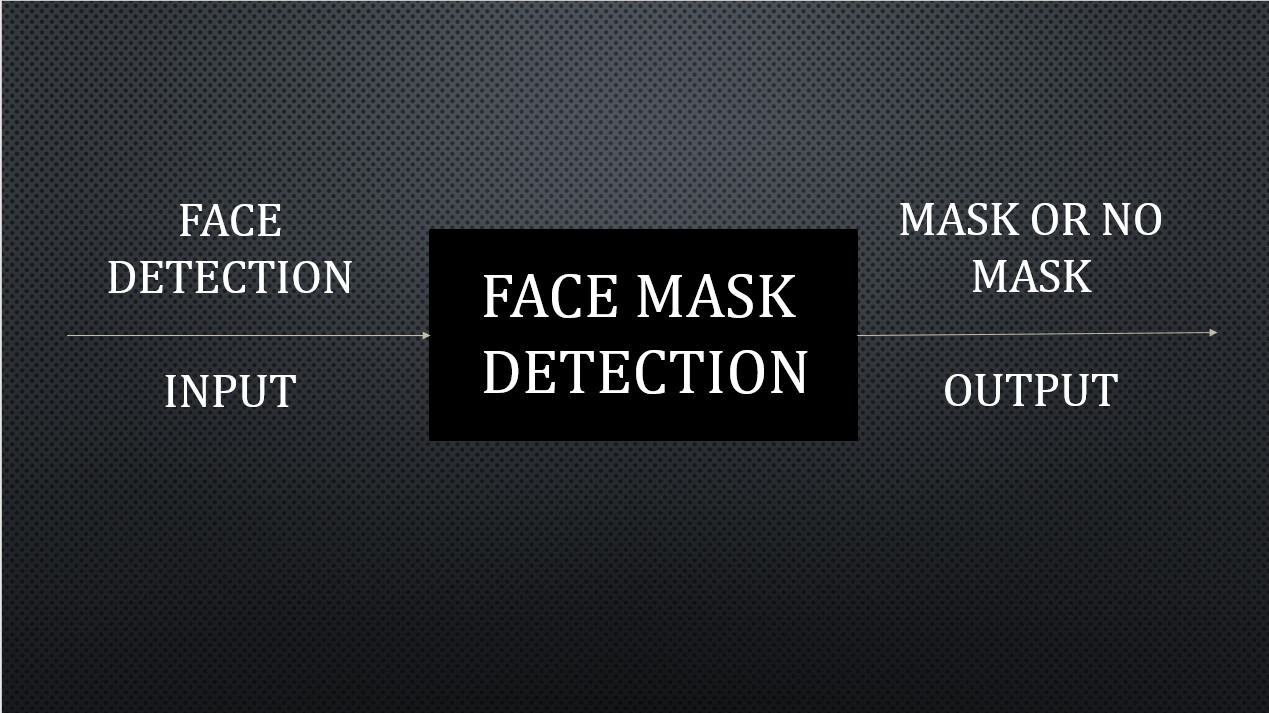
**7.2 Face detection with mask:**

****

**7.3 Face detection without the mask:**



**7.4 Input and Output:**



1. **SAMPLE CODE**

! pip install OpenCV-python

import cv2

import numpy as np

img = cv2.imread("C:/Users/user/OneDrive/Desktop/twillight.jpg")

img.shape

import matplotlib.pyplot as plt

plt.imshow(img)

haar\_data = cv2.CascadeClassifier('data.xml')

while True:

faces = haar\_data.detectMultiScale(img)

for x, y, w, h in faces:

cv2.rectangle(img, (x,y), (x+w, y+h), (255,0,255), 4)

cv2.imshow('result', img)

if cv2.waitKey(2) == 27:

break

cv2.destroyAllWindows()

capture = cv2.VideoCapture(0)

data = []

while True:

flag, img = capture.read()

if flag:

faces = haar\_data.detectMultiScale(img)

for x, y, w, h in faces:

cv2.rectangle(img, (x,y), (x+w, y+h), (255,0,255), 4)

face = img[y:y+h, x:x+w, :]

face = cv2.resize(face, (50,50))

print(len(data))

if (len(data)) < 800:

data.append(face)

cv2.imshow('result', img)

if cv2.waitKey(2) == 27 or len(data) >= 400:

break

capture.release()

cv2.destroyAllWindows()

np.save('without\_mask.npy',data)

np.save('With\_Mask.npy',data)

plt.imshow(data[0])

import numpy as np

import cv2

with\_mask = np.load('with\_mask.npy')

without\_mask = np.load('without\_mask.npy')

with\_mask.shape

without\_mask.shape

with\_mask = with\_mask.reshape(400, 50\*50\*3)

without\_mask = without\_mask.reshape(400, 50\*50\*3)

with\_mask.shape

without\_mask.shape

X = np.r\_[with\_mask, without\_mask]

X.shape

labels = np.zeros(X.shape[0])

labels[400 : ] = 1.0

names = {0 : "MASK", 1 : "N0\_MASK"}

from sklearn.svm import SVC

from sklearn.metrics import accuracy\_score

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(X,labels, test\_size = 0.25)

x\_train.shape

from sklearn.decomposition import PCA

pca = PCA(n\_components = 3)

x\_train = pca.fit\_transform(x\_train)

x\_train[0]

x\_train.shape

svm = SVC()

svm.fit(x\_train, y\_train)

x\_test = pca.fit\_transform(x\_test)

y\_pred = svm.predict(x\_test)

accuracy\_score(y\_test, y\_pred)

haar\_data = cv2.CascadeClassifier('data.xml')

capture = cv2.VideoCapture(0)

data = []

font = cv2.FONT\_HERSHEY\_COMPLEX

while True:

flag, img = capture.read()

if flag:

faces = haar\_data.detectMultiScale(img)

for x, y, w, h in faces:

cv2.rectangle(img, (x,y), (x+w, y+h), (255,0,255), 4)

face = img[y:y+h, x:x+w, :]

face = cv2.resize(face, (50,50))

face = face.reshape(1,-1)

pred = svm.predict(face)[0]

n = names[int(pred)]

cv2.putText(img, n, (x,y), font, 1, (244,250,250))

print(n)

cv2.imshow('result', img)

if cv2.waitKey(2) == 27 or len(data) >= 400:

break

capture.release()

cv2.destroyAllWindows()

**9. CONCLUSION**

With the increasing number of COVID cases all over the world, a system to replace humans to check masks on the faces of people is greatly needed. This system satisfies that need. This system can be employed in public places like railway stations and malls. It will be of a great help in companies and huge establishments where there will be a lot of workers. This system will be of a great help there because it is easy to obtain and store the data of the employees working in that Company and will very easy find the people who are not wearing the mask and a mail will sent to that respective person to take Precautions not wearing mask.



**10. REFERENCES**

1. M. S. Ejaz, M. R. Islam, M. Sifatullah and A. Sarker, "Implementation of Principal Component Analysis on Masked and Non-masked Face Recognition", 2019 1st International Conference on Advances in Science Engineering and Robotics Technology (ICASERT), pp. 1-5, 2019.
2. BOSHENG QIN and DONGXIAO LI, Identifying Facemask-wearing Condition Using Image Super-Resolution with Classification Network to Prevent COVID- 19, May 2020, [online] Available: <https://doi.org/10.21203/rs.3.rs-28668/v1>+].

**3**. Kaihan Lin, Huimin Zhao, Jujian Lv(&)Jin Zhan, Xiaoyong Liu, Rongjun Chen, Canyao Li, Zhihui Huang et al., "Face Detection and Segmentation with Generalized Intersection over Union Based on Mask R- CNN", Advances in Brain Inspired Cognitive System, pp. 106-116, 2020.

**4**. Ud Din, K. Javed, S. Bae and J. Yi, "A Novel GAN-Based Network for Unmasking of Masked Face", in IEEE Access, vol. 8, pp. 44276-44287, 2020.

1. Nicolae-Cătălin Ristea and Radu Tudor Ionescu, are you wearing a mask? Improving mask detection from speech using augmentation by cycle-consistent GANs.