# AN APPLICATION FOR FACE MASK DETECTION BASED ON DEEP LEARNING

**A PROJECT REPORT**

## Submitted by

**ARVINTH S**

**17ITR011**

**BHARANEESHWAR B 17ITR014**

**HARI PRASATH K V**

**17ITR033**

***in partial fulfilment of the requirements for the award of the degree***

***of***

**BACHELOR OF TECHNOLOGY IN**

**INFORMATION TECHNOLOGY**

# DEPARTMENT OF INFORMATION TECHNOLOGY SCHOOL OF COMMUNICATION AND COMPUTER SCIENCES

**KONGU ENGINEERING COLLEGE**

## (Autonomous)

**PERUNDURAI, ERODE – 638 060**

**DECEMBER 2020**

# I

# 

# DEPARTMENT OF INFORMATION TECHNOLOGY

# KONGU ENGINEERING COLLEGE

**(Autonomous)**

**PERUNDURAI, ERODE – 638 060**

**DECEMBER 2020**

# BONAFIDE CERTIFICATE

This is to certify that the Project Report entitled **AN APPLICATION FOR FACE MASK DETECTION BASED ON DEEP LEARNING** is the bonafide record of project work done by **ARVINTH S (17ITR011), BHARANEESHWAR B (17ITR014),** and **HARI PRASATH K V (17ITR033)** in partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology in **INFORMATION TECHNOLOGY** of Anna University, Chennai during the year 2020-2021.

## SUPERVISOR HEAD OF THE DEPARTMENT

**(Signature with seal)**

Date:

Submitted for the end semester viva voce examination held on

## INTERNAL EXAMINER EXTERNAL EXAMINER

# II

**DEPARTMENT OF INFORMATION TECHNOLOGY KONGU ENGINEERING COLLEGE**

**(Autonomous)**

**PERUNDURAI ERODE – 638 060**

**DECEMBER 2020**

**DECLARATION**

We affirm that the Project Report titled **AN APPLICATION FOR FACE MASK DETECTION BASED ON DEEP LEARNING** being submitted in partial fulfilment of the requirements for the award of Bachelor of Technology is the original work carried out by us. It has not formed the part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

## ARVINTH S

**(17ITR011)**

**BHARANEESHWAR B**

**(17ITR014)**

**HARI PRASATH K V**

**(17ITR033)**

**Date:**

I certify that the declaration made by the above candidates is true to the best of my knowledge.

Date: Name and Signature of the Supervisor with seal

# III

# ABSTRACT

Deep learning has shown tremendous potential in many real-life applications in different domains. One of these potentials is object detection using computer vision techniques. As the devastating COVID-19 pandemic outbreak continues, innovators are coming up with innovative digital solutions. One of them is a face mask detection system to identify people with and without proper face masks. On the other hand, it will be challenging to recognize faces with masks on any monitoring systems, while maintaining touch-less access control in buildings. The objective of this paper is to annotate and localize the face mask in real-life videos. Here we have designed a facemask detection system by applying CNN Deep Learning algorithm. The proposed model consists of two components. The first component is designed for the facial feature extraction process based on the FaceNet deep transfer learning model. While the second component is designed for the detection of face masks based on MobileNetv2. The model is able to predict the masked faces with 99.00% accuracy. Besides, we also explore the possibility of implementing FaceMaskDetector with a light-weighted neural network MobileNetV2 for embedded or mobile devices. The proposed detector achieved higher accuracy and precision than the related work.

# IV

# ACKNOWLEDGEMENT

First and foremost, we acknowledge the abundant grace and presence of Almighty throughout different phases of the project and its successful completion.

We wish to express our extreme gratefulness to our beloved Correspondent

**Thiru. P. SACHITHANANDAN,** and all the trust members of Kongu Vellalar Institute of Technology Trust for providing all the necessary facilities to complete the project successfully.

We express our deep sense of gratitude to our beloved Principal **Dr. V. BALUSAMY B.E(Hons)., M.Tech., Ph.D.,** for providing us an opportunity to complete the project.

We express our gratitude to **Dr. R. THANGARAJAN M.E., Ph.D.,** Head of the Department, Department of Information Technology, for his valuable suggestions.

We are thankful to our project coordinators **Dr. P. SURESH M.E., Ph.D.,** and **Dr. K. LALITHA M.E., Ph.D.,** for their valuable guidance and support to complete our project successfully.

We are highly indebted to **Dr. R. SHANTHA KUMARI M.E., Ph.D.,** Department of Information Technology, for her valuable supervision and advice for the fruitful completion of the project.

Finally, we extend our love and thanks to our Parents and Friends for their patience in teaching, love, encouragement and support.

**V**

# TABLE OF CONTENTS

## CHAPTER

**No.**

**ABSTRACT**

**TITLE PAGE**

**No.**

**III**

## LIST OF ABBREVIATIONS LIST OF FIGURES

1. **INTRODUCTION**
   1. INTRODUCTION
      1. DEEP LEARNING
      2. CONVOLUTIONAL NEURAL NETWORKS
   2. OBJECTIVE
   3. SCOPE

## SYSTEM ANALYSIS

* 1. LITERATURE REVIEW
  2. SUMMARY

**VII**

**VIII**

## 1

1

1

1

2

2

## 3

3

4

## PROBLEM DEFINITION 5

* 1. EXISTING SYSTEM 5
  2. PROBLEM STATEMENT 5
  3. PROPOSED SYSTEM 6

## SYSTEM REQUIREMENTS 8

* 1. HARDWARE REQUIREMENTS 8
  2. SOFTWARE REQUIREMENTS 8
  3. SOFTWARE DESCRIPTION 8
     1. PYTHON 3 8
     2. TENSORFLOW 9

**VI**

* + 1. OPENCV 9
    2. VISUAL STUDIO CODE 10

1. [SYSTEM DESIGN 11](#_TOC_250013)
   1. [PROPOSED METHODOLOGY 11](#_TOC_250012)
   2. [MODULE DESCRIPTION 12](#_TOC_250011)
      1. IMAGE ACQUISITION 13
      2. [FACENET 13](#_TOC_250010)
      3. [MASKNET 13](#_TOC_250009)
      4. [DETECTING FACES WITH MASK AND NO MASK 14](#_TOC_250007)
2. RESULTS AND DISCUSSION 16
3. [CONCLUSION AND FUTURE WORK 17](#_TOC_250004)

[APPENDICES 18](#_TOC_250003)

[APPENDIX-1 CODING 18](#_TOC_250002)

[APPENDIX-2 SCREENSHOTS 23](#_TOC_250001)

[REFERENCES 27](#_TOC_250000)

**VII**

|  |  |
| --- | --- |
| **LIST OF ABBREVIATIONS** | |
| **ABBREVIATION** | **EXPANSION** |
| YOLOV2 | You Only Look Once Version 2 |
| AI | Artificial Intelligence |
| CNN | Convolutional Neural Networks |
| ReLU | Rectified Linear Unit |
| RGB | Red Green Blue |
| RAM | Random Access Memory |
| PR | Pattern Recognition |

**VIII**

# LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **FIGURE No.** | **FIGURE NAME** | **PAGE No.** |
|  |  |  |
| 3a | Flow Chart of Face Mask Detection System | 6 |
| 3b | CNN layers of MaskNet model | 7 |
| 7a | Comparison between Algorithms | 17 |
| 7b | Training/Testing Accuracy Graph | 18 |
| 7c | Training/Testing Loss Graph | 18 |
| A 2.1 | Evaluating the Performance of Networks | 25 |
| A 2.2 | Detection of Masked Face with 99.55% | 25 |
| A 2.3 | Detection of Unmasked Face with 100.00% | 26 |
| A 2.4 | Detection of improper Masked Face as No Mask with 84.53% | 26 |
| A 2.5 | Detection of Face covered by hands as No Mask with 97.34% | 27 |
| A 2.6 | Detection of Masked Face and Unmasked face both with 99.7% | 27 |
| A 2.7 | Detection of all unmasked Faces above 99% | 28 |

**CHAPTER 1**

**INTRODUCTION**

## INTRODUCTION

* + 1. **DEEP LEARNING**

In the area of image recognition and classification, the most successful results were obtained using convolutional neural networks. These networks form the basis for most deep learning models. Deep learning is a class of machine learning algorithms that use multiple layers that contain nonlinear processing units. Each level learns to transform its input data into a slightly more abstract and composite representation. Deep neural networks have managed to outperform other machine learning algorithms. They also achieved the first superhuman pattern recognition in certain domains. This is further reinforced by the fact that deep learning is considered as an important step towards obtaining Strong AI. Secondly, deep neural networks specifically convolutional neural networks have been proved to obtain great results in the field of image recognition.

* + 1. **MACINE LEARNING**

Machine learning (ML) is the study of computer algorithms that improve automatically through experience and by the use of data. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks. Machine learning involves computers discovering how they can perform tasks without being explicitly programmed to do so. It involves computers learning from data provided so that they carry out certain tasks. For simple tasks assigned to computers, it is possible to program algorithms telling the machine how to execute all steps required to solve the problem at hand; on the computer's part, no learning is needed. For more advanced tasks, it can be challenging for a human to manually create the needed algorithms. In practice, it can turn out to be more effective to help the machine develop its own algorithm, rather than having human programmers specify every needed step.

## CONVOLUTIONAL NEURAL NETWORKS

Convolutional neural networks (CNN) are part of the deep learning models. Such a network can be composed of convolutional layers, pooling layers, ReLU layers, fully connected layers and loss layers. A characteristic that sets apart the CNN from a regular neural network is taking into account the structure of the images while processing them.

**Convolutional layers :** These layers are named after the convolution operation. In mathematics convolution is an operation on two functions that produces a third function that is the modified (convoluted) version of one of the original functions. The resulting function gives an integral of the pointwise multiplication of the two functions as a function of the amount that one of the original functions is translated.

**Pooling layers :** These layers are used on one hand to reduce the spatial dimensions of the representation and to reduce the amount of computation done in the network. The other use of pooling layers is to control overfitting. The most used pooling layer has filters of size 2 x 2 with a stride 2. This effectively reduces the input to a quarter of its original size.

**Fully connected layers :** Each neuron from a fully connected layer is linked to each output of the previous layer. The operations behind a convolutional layer are the same as in a fully connected layer.

**Loss layers :** These layers are used to penalize the network for deviating from the expected output. This is normally the last layer of the network. Various loss functions exist: softmax is used for predicting a class from multiple disjunct classes, sigmoid cross-entropy is used for predicting multiple independent probabilities (from the [0, 1] interval).

## OBJECTIVE

The objective is creating a framework established by [TensorFlow](https://tensorflow.org/lite), OpenCV, camera, dataset, and pre-trained model. FaceNet and MobileNetV2 are used to improve the efficiency of identifying faces with and without mask with a better detection system for reducing computational complexity. The purpose of the system is to minimize the number of human computer interactions, speed up the identification process and improve the usability of the graphical user interface compared to existing manual systems. The system is initially developed using a Python Application for the detection of faces with and without mask based on certain features. To detect and classify a face, different convolutional neural networks have been tested and retrained.

## SCOPE

A unified approach that can detect and classify the faces as mask and no mask, where all features are simply concatenated and fed independently to the algorithm. Detection of faces with and without masks are implemented in different areas. The most common areas are identification in the public places, and in areas where the purpose is to monitor the wearing of mask throughout their presence. The main scope of the project is to prevent the rigorous spread of the virus through proper wearing of mask.

one of the original functions is translated.

**Pooling layers :** These layers are used on one hand to reduce the spatial dimensions of the representation and to reduce the amount of computation done in the network. The other use of pooling layers is to control overfitting. The most used pooling layer has filters of size 2 x 2 with a stride 2. This effectively reduces the input to a quarter of its original size.

**Fully connected layers :** Each neuron from a fully connected layer is linked to each output of the previous layer. The operations behind a convolutional layer are the same as in a fully connected layer.

**Loss layers :** These layers are used to penalize the network for deviating from the expected output. This is normally the last layer of the network. Various loss functions exist: softmax is used for predicting a class from multiple disjunct classes, sigmoid cross-entropy is used for predicting multiple independent probabilities (from the [0, 1] interval).

## OBJECTIVE

The objective is creating a framework established by [TensorFlow](https://tensorflow.org/lite), OpenCV, camera, dataset, and pre-trained model. FaceNet and MobileNetV2 are used to improve the efficiency of identifying faces with and without mask with a better detection system for reducing computational complexity. The purpose of the system is to minimize the number of human computer interactions, speed up the identification process and improve the usability of the graphical user interface compared to existing manual systems. The system is initially developed using a Python Application for the detection of faces with and without mask based on certain features. To detect and classify a face, different convolutional neural networks have been tested and retrained.

## SCOPE

A unified approach that can detect and classify the faces as mask and no mask, where all features are simply concatenated and fed independently to the algorithm. Detection of faces with and without masks are implemented in different areas. The most common areas are identification in the public places, and in areas where the purpose is to monitor the wearing of mask throughout their presence. The main scope of the project is to prevent the rigorous spread of the virus through proper wearing of mask.

# CHAPTER 2

**SYSTEM ANALYSIS**

## LITERATURE REVIEW

1. M. M. Rahman, M. M. H. Manik, Saifuddin M, Jong-Hoon Kim, and M. M. Islam, “An Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network”, proposes a system for face mask detection using a CNN that is customized accordingly for better results yet it fails to distinguish between a mask and hands covering the face.
2. Mohamed Loey, Gunasekaran Manogaran, Mohamed Hamed N. Taha, and Nour Eldeen M. Khalifa, “Fighting against COVID-19: A novel deep learning model based on YOLO-v2 with ResNet-50 for medical face mask detection” Sustainable Cities and Society, Elsevier Ltd, published on November 12, 2020, this paper works on two components namely the ResNet-50 and YOLO-V2 together making the face mask detector for predicting the output. Yet, the accuracy tends to be less.
3. Jose sigut, M. castro, R. Arnay, and M. sigut, “OpenCV Basics: A Mobile Application to Support the Teaching of Computer Vision Concepts,” IEEE Transactions on Education, this is an application designed with the purpose of facilitating the initiation of field of computer vision, making the learning process easier.
4. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, Google Inc. “Mobilenetv2: Inverted residuals and linear bottlenecks” – 2018 IEEE conference on computer vision and pattern recognition (pp. 4510-4520), this is a python framework of under tensorflow based on Convolutional Neural Networks, which is proved as one of the fastest working object detection algorithms for small number of classes.
5. C. Jagadeeswari, and M. Uday Theja, – “Performance Evaluation of Ingelligent Face Mask Detection System with various Deep Learning Classifiers” – International Journal of Advanced Science and Technology, vol. 29, No. 11s, (2020). Pp. 3074-3082, this paper uses an ADAM optimizer along with YOLO, ResNet, and MobileNetV2 for providing a face mask detection system and this system genrates a reminder for the person for not wearing mask and an alarm.
6. L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, “Deep learning for generic object detection: A survey,” International journal of computer vision, proposes a object detection system regardless of the different types of classes, which in turn can be used for face detection as well.

## SUMMARY

The previous existing systems namely the YOLO-V2 and ResNet-50, CNN for PR in images are less accurate and have some drawbacks. They have good performance but the drawbacks of having difficulty in differentiating between hands and masks covering the face. In order to improve the functionality and flexibility of the detection system, facial and mask features can be combined together with color and texture features with image augmentation along with different postures of face. By increasing the number of images in the data set the detection rate can be increased. If an object size in a testing image is significantly less than that of a training set, it misses the detection. So, high-quality and more training image sets covering multiple scale levels are required. The proposed method is not robust to noise.

# CHAPTER 3 PROBLEM DEFINITION

## EXISTING SYSTEM

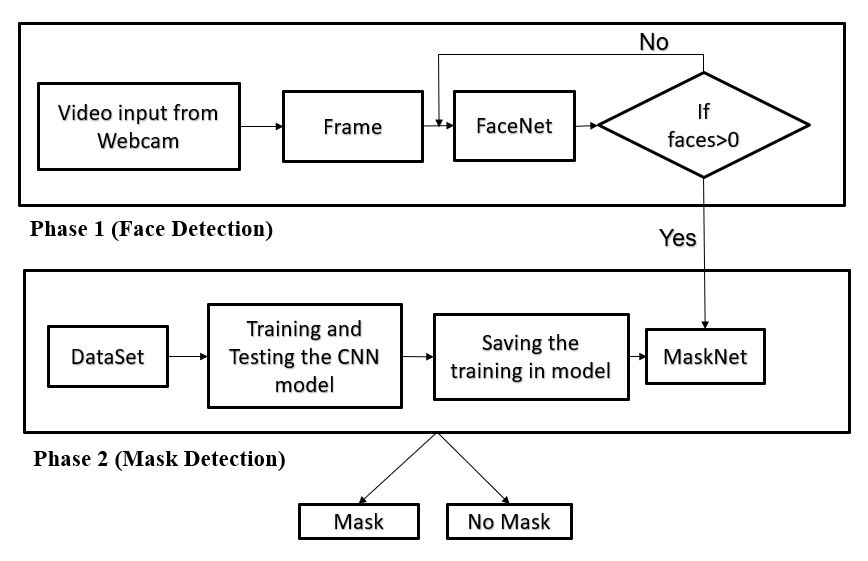
Convolutional neural networks for face mask detection models are available, which scales to realistic image sizes are used in the face mask detection and not for complex, high dimensional data. In order to improve the functionality and flexibility of the detection system, facial and mask features can be combined together with color and texture features with image augmentation along with different postures of face. Color and texture are the fundamental characteristics of natural images, and play an important role in visual perception. Color has been a great help in identifying objects for many years. It is often useful to simplify a monochrome problem by improving contrast or separation. The process of face detection and classification based on the presence of mask involves extraction of useful information concerning the spectral properties of the face and discovering the best match from a set of known descriptions or class models to implement the recognition task. If an object size in a testing image is significantly less than that of a training set, it misses the detection. So, high-quality and more training image sets covering multiple scale levels are required.

## PROBLEM STATEMENT

Detection and Recognition system has emerged as a ‘grand challenge' for computer vision, with the longer-term aim of being able to achieve near human levels of recognition for tens of thousands of categories under a wide variety of conditions. On the off chance that apply further convolutional layers, the range is additionally decreased, and, accordingly the picture range is definitely diminished which produces defeat of data and the disappearing angle issue. Cushioning builds the range of input information by satisfying constants approximately input information. Most of the existing datasets with images contain both the object and the noisy background that lead to cases where changing the background will lead to the incorrect classification of the object. If an object size in a testing image is significantly less than that of a training set, it misses the detection.

## PROPOSED SYSTEM

The proposed Face Mask Detection system, shown in Figure 3a, comprises of mainly the two components. They are FaceNet for the face detection and the Convolutional neural network layer using MobileNetV2 named as MaskNet for the detection of proper presence mask in the face. The FaceNet is a pre trained model proposed by Google Inc., which is regarded as one of the best systems for face detection and recognition. This system is used to detect the faces and then it is fed into the MaskNet model to classify the face as mask or no mask.

Figure. 3a Face Mask Detection System

The MaskNet model is deployed by developing a customized MobileNetV2 layer, CNN layers, ReLU layers, and Softmax layers processed through max-pooling, flattening, and dropout regularization. The model is developed, then it is trained using various images of dataset by using the image augmentation technique. The model is trained for 100 epochs and graphs are plotted for loss and accuracy over training and testing.

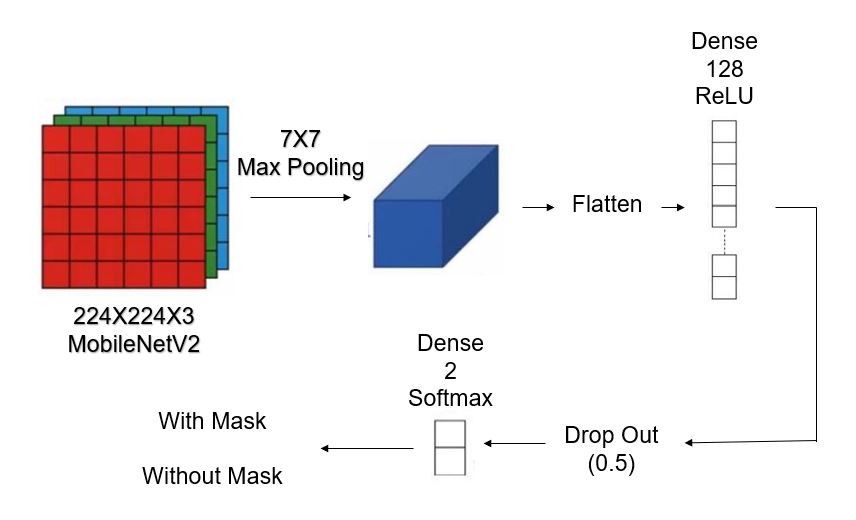


Figure. 3b CNN layers of MaskNet model

The Dropout regularization is used to minimize the loss during the detection. Softmax is used to get better performance in case of small number of output classes thus, softmax is deployed as output layer.

# CHAPTER 4 SYSTEM REQUIREMENTS

## HARDWARE REQUIREMENTS

Processor : Intel Core i5 Speed : 2.4 GHz

RAM : 8 GB

Hard Disk : 50 GB

## SOFTWARE REQUIREMENTS

Operating System : Windows 10 Programming Languages : Python 3

IDE : Visual Studio Code

Framework : TensorFlow, OpenCV

## SOFTWARE DESCRIPTION

## PYTHON 3

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input never causes a segmentation fault. Instead, when the

interpreter discovers an error; it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

## TENSORFLOW

TensorFlow is a [free](https://en.wikipedia.org/wiki/Free_software) and [open-source](https://en.wikipedia.org/wiki/Open-source_software) [software library](https://en.wikipedia.org/wiki/Library_(computing)) for [machine learning](https://en.wikipedia.org/wiki/Machine_learning). It can be used across a range of tasks but has a particular focus on [training](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#Training) and [inference](https://en.wikipedia.org/wiki/Statistical_inference) of [deep neural networks](https://en.wikipedia.org/wiki/Deep_neural_networks). TensorFlow is a symbolic math library based on [dataflow](https://en.wikipedia.org/wiki/Dataflow_programming) and [differentiable programming](https://en.wikipedia.org/wiki/Differentiable_programming). It is used for both research and production at [Google](https://en.wikipedia.org/wiki/Google). TensorFlow was developed by the [Google Brain](https://en.wikipedia.org/wiki/Google_Brain) team for internal [Google](https://en.wikipedia.org/wiki/Google) use. It was released under the [Apache License 2.0](https://en.wikipedia.org/wiki/Apache_License_2.0) in 2015.

## OPENCV

OpenCV (Open Source Computer Vision Library) is a [library of programming functions](https://en.wikipedia.org/wiki/Library_(computing)) mainly aimed at real-time [computer vision](https://en.wikipedia.org/wiki/Computer_vision). Originally developed by [Intel](https://en.wikipedia.org/wiki/Intel_Corporation), it was later supported by [Willow Garage](https://en.wikipedia.org/wiki/Willow_Garage) then Itseez (which was later acquired by Intel). The library is [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) and free for use under the [open-source](https://en.wikipedia.org/wiki/Open-source_software) [Apache 2 License](https://en.wikipedia.org/wiki/Apache_License). Starting with 2011, OpenCV features GPU acceleration for real-time operations the goals of the project were described as:

* Advance vision research by providing not only open but also [optimized code](https://en.wikipedia.org/wiki/Code_optimization) for basic vision infrastructure. No more [reinventing the wheel](https://en.wikipedia.org/wiki/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](https://en.wikipedia.org/wiki/Portability_(computer_science)), performance-optimized code available for free – with a license that did not require code to be open or free itself.

## VISUAL STUDIO CODE

VSCode (Visual Studio Code) is a free [source-code editor](https://en.wikipedia.org/wiki/Source-code_editor) made by [Microsoft](https://en.wikipedia.org/wiki/Microsoft) for [Windows](https://en.wikipedia.org/wiki/Windows), [Linux](https://en.wikipedia.org/wiki/Linux) and [macOS](https://en.wikipedia.org/wiki/MacOS). Features include support for [debugging](https://en.wikipedia.org/wiki/Debugging), [syntax highlighting](https://en.wikipedia.org/wiki/Syntax_highlighting), [intelligent code completion](https://en.wikipedia.org/wiki/Intelligent_code_completion), [snippets](https://en.wikipedia.org/wiki/Snippet_(programming)), [code refactoring](https://en.wikipedia.org/wiki/Code_refactoring), and embedded [Git](https://en.wikipedia.org/wiki/Git). Users can change the [theme](https://en.wikipedia.org/wiki/Theme_(computing)), [keyboard shortcuts](https://en.wikipedia.org/wiki/Keyboard_shortcut), preferences, and install [extensions](https://en.wikipedia.org/wiki/Plug-in_(computing)) that add additional functionality. Microsoft has released Visual Studio Code's [source code](https://en.wikipedia.org/wiki/Source_code) on the VSCode repository of [GitHub.com](https://en.wikipedia.org/wiki/GitHub.com), under the permissive [MIT License](https://en.wikipedia.org/wiki/MIT_License), while the compiled releases are [freeware](https://en.wikipedia.org/wiki/Freeware). In the [Stack Overflow](https://en.wikipedia.org/wiki/Stack_Overflow) 2019 Developer Survey, Visual Studio Code was ranked the most popular developer environment tool, with 50.7% of 87,317 respondents reporting that they use it. Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including [Java](https://en.wikipedia.org/wiki/Java_(programming_language)), [JavaScript](https://en.wikipedia.org/wiki/JavaScript), [Go](https://en.wikipedia.org/wiki/Go_(programming_language)), [Node.js](https://en.wikipedia.org/wiki/Node.js) and [C++](https://en.wikipedia.org/wiki/C%2B%2B). It is based on the [Electron](https://en.wikipedia.org/wiki/Electron_(software_framework)) framework,[[18]](https://en.wikipedia.org/wiki/Visual_Studio_Code#cite_note-ars-electron-18) which is used to develop [Node.js](https://en.wikipedia.org/wiki/Node.js) [Web applications](https://en.wikipedia.org/wiki/Web_application) that run on the [Blink layout engine](https://en.wikipedia.org/wiki/Blink_layout_engine). Visual Studio Code employs the same editor component (codenamed "Monaco") used in [Azure DevOps](https://en.wikipedia.org/wiki/Azure_DevOps_Server) (formerly called Visual Studio Online and Visual Studio Team Services). Instead of a project system, it allows users to open one or more directories, which can then be saved in workspaces for future reuse. This allows it to operate as a [language-agnostic](https://en.wikipedia.org/wiki/Language-agnostic) code editor for any language. It supports a number of programming languages and a set of features that differs per language. Unwanted files and folders can be excluded from the project tree via the settings. Many Visual Studio Code features are not exposed through menus or the user interface but can be accessed via the command palette. It has the features like

* Language support,
* Data collection, and
* Version Control.

# CHAPTER 5 SYSTEM DESIGN

## PROPOSED METHODOLOGY

CNN can extract features from sample images automatically and does not only depend on color features. A variety of new CNN models have been proposed in recent years. For example, MobileNetV2 is popular for its good performance. It has been used to detect objects. In addition, a MobileNetV2 model for detecting faces with and without mask was constructed and trained.

In this study, the FaceNet model is used for detecting faces which is a pretrained one by involving transfer learning technique. The facial features such as eyes, nose, chin, etc., are used to train the model to accurately detect the faces. MobileNetV2 is another new CNN model, which can detect objects at higher speed. It was applied to detect face mask. The detection rate of the model reaches 83 fps. However, the training of CNNs needs massive labeled images. Collecting and labeling so many images require a great amount of time and labor costs. Moreover, the training and running of CNNs also require better hardware.

The gathering of neurons that build up pieces is composed of convolutional layers. The pieces have a little range however they generally have a similar profundity as the contribution. The neurons from a bit are associated with a little locale of the contribution, called the open field, since it is profoundly wasteful to connect all neurons to every past yield on account of contributions of high measurements, for example, pictures. The yield of a convolutional layer is through by stacking the come about actuation maps which in bowed is utilized to characterize the contribution of the next layer.

* + - **Input Layer** The picture holds the unfinished pixel estimations of the picture, right now picture of width 224, tallness 224, and with three shading channels R, G, B.
    - **The CONV Layer** turns the yield of neurons that are connected with nearby districts in the information, each registering an item sandwiched between their loads and a little locale they are related with in the info volume. This may bring about a degree, for example, on the off chance that chose to make the most of 12 channels.
    - **RELU Layer** applies a factor wise initiation work, for example, the max (0, x) thresholding at zero. This leaves the size of the level unaltered
    - **POOL Layer** plays out a down sampling activity along the spatial measurements (width, tallness), bringing about volume.
    - **Softmax Layer** is the output layer, which is regarded as the good output providing the activation function. Since, the softmax is better for low number of classes and here we have only two classes namely mask and no mask.

## MODULE DESCRIPTION

Complex and tedious self-administration frameworks may bring about clients picking another supermarket. Since clients are the explanation organizations endure, their fulfillment is the organization’s key to progress. The need of frameworks which diminishes the procedure time exists due to buyer’s desires for their steady undertaking to spare time. As needs be, the reason for this work is to improve the recognizable proof procedure of foods grown from the ground performed by the self-administration frameworks in the retail showcase. All the more explicitly, the development should comprise a quicker procedure and an easier to use framework. The motivation behind executing PC vision to the framework is to limit the choice of conceivable items and in this manner lessen the injury to the client. Moreover, the utilization of PC ideas in self-administration frameworks can disentangle the way toward recognizing objects by poignant the procedure from a person to a PC.

Hypothetically, this could hurry the procedure to recognize items and limit the measure of mistakes by expelling the human being factor. As far as equipment, the good times-damentals are a camera, a presentation, an actuation instrument speaking to the level, and a processor to run the framework. A picture classifier has been prepared and assessed to order pictures of foods grown from the ground from the camera. A UI has been created also to deal with client connection by means of the presentation.

## IMAGE ACQUISITION

Various pictures and faces with and without masks are caught through the camera for the training purpose in this module. The training images should be without impediment, occlusion, and illuminating condition. The Convolutional Neural Networks are evaluated by two properties: propagation time, which is the time it takes for an image to be classified, and accuracy, which is how accurate the prediction is. To evaluate the propagation time of the networks, a series of 100 images was captured and classified. The propagation time is the time between start and end of the classification.

## FACENET

The features are extracted by the proposed algorithm where it is the initial set of processes which reduces the raw set of data to the manageable process. It is difficult to segment faces from the background completely based on the facial features. In order to locate the position of face further, the shape feature of face is employed. It is pre-trained in order to reduce the complexity of the working process. Since the model is pretrained it also paves the way for better output. This also reduces the time taken for training the model, which saves a lot of time.

## MASKNET

The MaskNet model is the final output layers. Here, we deploy a technique of training only the output layers which reduces the time complexity problem. We create a customized dataset of images with two classes namely the with\_mask and without\_mask. The MobileNetV2 framework is used for input layers, max pooling layers, ReLU layers, and softmax layers are involved in developing the MaskNet model.

## DETECTING FACE WITH MASK AND NO MASK

In this step, the video stream is taken as single frames as input for the FaceNet model. The FaceNet identifies and detects all the faces in the input video. These faces are fed as input for MaskNet. The faces are checked against for the presence of maks. The faces with mask are bounded with green boxes with the percentage of precision and if no mask is worn, or if the mask is not worn properly, or if the face is covered with hands or covered with objects other than mask the face is identified as No with a red bounding boxes with the respective precision percentage.

# CHAPTER 6 RESULTS AND DISCUSSION

The proposed method can process, analyze and detect faces with and without mask based on facial features and mask objects. In order to improve the functionality and flexibility of the detection system, facial and mask features can be combined together with color and texture features with image augmentation along with different postures of face. This algorithm can be used for smart self-service scales. In order to verify the performance of the proposed method, it will be compared with three other methods. The method is similar to the proposed method with some changes in algorithms.

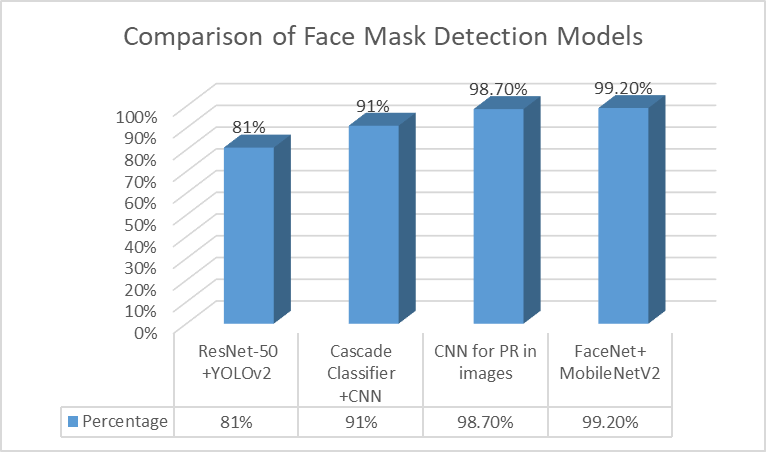


Figure 7a Comparison between Algorithms

The above figure 7a graph depicts comparisons of different algorithms for face mask detection. It is obvious that the accuracy of the proposed system, is more when compared to other methods. This proves that the proposed system is better than the previous existing systems.

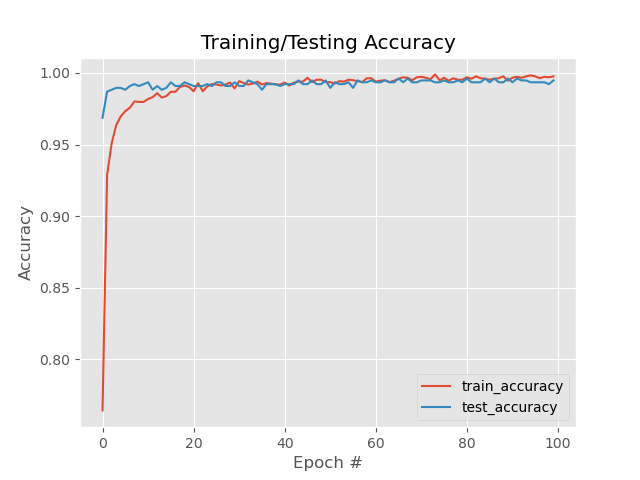
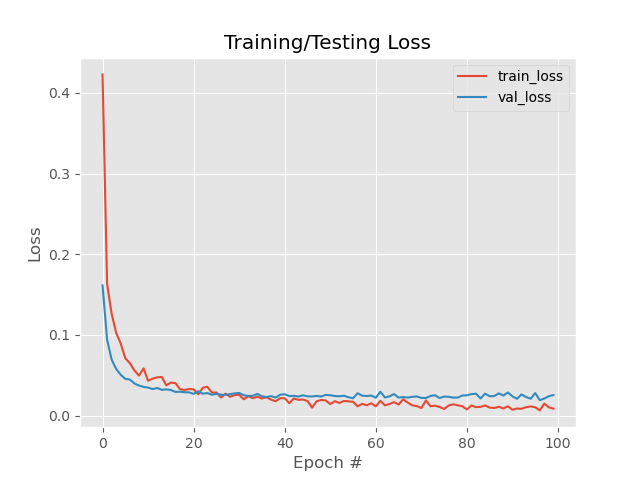
 

Figure 7b Training/Testing Accuracy Graph Figure 7c Training/Testing Loss Graph

The above figures 7b and 7c illustrates the graphical representation of accuracy and loss of the model over training and testing of the model. We can clearly see that the accuracy is increasing as the training through the number of epochs increases and vice versa. Also, the loss of the model while training and testing decreases exponentially through the number of the epoch is increased. This experiment just indicates that the proposed method has better performance in the same conditions because of candidate regions in general.

# CHAPTER 8 CONCLUSION AND FUTURE WORK

Finally, faces with masks and without masks are detected by surrounding the faces with bounding boxes and the performance of the proposed method is good. However, it should be noted that it is not robust to noise. This system has increased the accuracy of the face mask detection and the faster performance of the detection system by using the pre-trained FaceNet model and MaskNet through MobileNetV2 framework of the Tensorflow parent framework. The proposed method combines the facial feature and mask object to detect face with and without masks and can improve the accuracy of face mask Detection. Nevertheless, it just detects faces with and without masks by square bounding boxes. Although this system detects faces and classify them based on masks, but not recognize them. Future work includes the recognition of faces even in the presence of mask.

## APPENDIX-1

**CODING**

**Training and saving the model:**

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.applications import MobileNetV2

from tensorflow.keras.layers import AveragePooling2D

from tensorflow.keras.layers import Dropout

from tensorflow.keras.layers import Flatten

from tensorflow.keras.layers import Dense

from tensorflow.keras.layers import Input

from tensorflow.keras.models import Model

from tensorflow.keras.optimizers import Adam

from tensorflow.keras.applications.mobilenet\_v2 import preprocess\_input

from tensorflow.keras.preprocessing.image import img\_to\_array

from tensorflow.keras.preprocessing.image import load\_img

from tensorflow.keras.utils import to\_categorical

from sklearn.preprocessing import LabelBinarizer

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

from sklearn.metrics import classification\_report

from imutils import paths

import matplotlib.pyplot as plt

import numpy as np

import os

INIT\_LR = 1e-4

EPOCHS = 100

BS = 100

DIRECTORY = r"E:\Face-Mask\dataset"

CATEGORIES = ["with\_mask", "without\_mask"]

print("[INFO] loading images...")

data = []

labels = []

for category in CATEGORIES:

path = os.path.join(DIRECTORY, category)

for img in os.listdir(path):

img\_path = os.path.join(path, img)

image = load\_img(img\_path, target\_size=(224, 224))

image = img\_to\_array(image)

image = preprocess\_input(image)

data.append(image)

labels.append(category)

lb = LabelBinarizer()

labels = lb.fit\_transform(labels)

labels = to\_categorical(labels)

data = np.array(data, dtype="float32")

labels = np.array(labels)

(trainX, testX, trainY, testY) = train\_test\_split(data, labels,

test\_size=0.20, stratify=labels, random\_state=42)

aug = ImageDataGenerator(

rotation\_range=20,

zoom\_range=0.15,

width\_shift\_range=0.2,

height\_shift\_range=0.2,

shear\_range=0.15,

horizontal\_flip=True,

fill\_mode="nearest")

baseModel = MobileNetV2(weights="imagenet", include\_top=False,

input\_tensor=Input(shape=(224, 224, 3)))

headModel = baseModel.output

headModel = AveragePooling2D(pool\_size=(7, 7))(headModel)

headModel = Flatten(name="flatten")(headModel)

headModel = Dense(128, activation="relu")(headModel)

headModel = Dropout(0.5)(headModel)

headModel = Dense(2, activation="softmax")(headModel)

model = Model(inputs=baseModel.input, outputs=headModel)

for layer in baseModel.layers:

layer.trainable = False

print("[INFO] compiling model...")

opt = Adam(lr=INIT\_LR, decay=INIT\_LR / EPOCHS)

model.compile(loss="binary\_crossentropy", optimizer=opt,

metrics=["accuracy"])

print("[INFO] training head...")

H = model.fit(

aug.flow(trainX, trainY, batch\_size=BS),

steps\_per\_epoch=len(trainX) // BS,

validation\_data=(testX, testY),

validation\_steps=len(testX) // BS,

epochs=EPOCHS)

print("[INFO] evaluating network...")

predIdxs = model.predict(testX, batch\_size=BS)

predIdxs = np.argmax(predIdxs, axis=1)

print(classification\_report(testY.argmax(axis=1), predIdxs,

target\_names=lb.classes\_))

print("[INFO] saving mask detector model...")

model.save("mask\_detector.model", save\_format="h5")

N = EPOCHS

plt.style.use("ggplot")

plt.figure()

plt.plot(np.arange(0, N), H.history["loss"], label="train\_loss")

plt.plot(np.arange(0, N), H.history["val\_loss"], label="val\_loss")

plt.plot(np.arange(0, N), H.history["accuracy"], label="train\_accuracy")

plt.plot(np.arange(0, N), H.history["val\_accuracy"], label="test\_accuracy")

plt.title("Training/Testing Accuracy")

plt.xlabel("Epoch #")

plt.ylabel("Accuracy")

plt.legend(loc="lower right")

plt.savefig("accuracy\_plot.png")

**Detect Faces with and without mask:**

from tensorflow.keras.applications.mobilenet\_v2 import preprocess\_input

from tensorflow.keras.preprocessing.image import img\_to\_array

from tensorflow.keras.models import load\_model

from imutils.video import VideoStream

import numpy as np

import imutils

import time

import cv2

import os

def detect\_and\_predict\_mask(frame, faceNet, maskNet):

(h, w) = frame.shape[:2]

blob = cv2.dnn.blobFromImage(frame, 1.0, (224, 224),

(104.0, 177.0, 123.0))

faceNet.setInput(blob)

detections = faceNet.forward()

print(detections.shape)

faces = []

locs = []

preds = []

for i in range(0, detections.shape[2]):

confidence = detections[0, 0, i, 2]

if confidence > 0.5:

box = detections[0, 0, i, 3:7] \* np.array([w, h, w, h])

(startX, startY, endX, endY) = box.astype("int")

(startX, startY) = (max(0, startX), max(0, startY))

(endX, endY) = (min(w - 1, endX), min(h - 1, endY))

face = frame[startY:endY, startX:endX]

face = cv2.cvtColor(face, cv2.COLOR\_BGR2RGB)

face = cv2.resize(face, (224, 224))

face = img\_to\_array(face)

face = preprocess\_input(face)

faces.append(face)

locs.append((startX, startY, endX, endY))

if len(faces) > 0:

faces = np.array(faces, dtype="float32")

preds = maskNet.predict(faces, batch\_size=32)

return (locs, preds)

prototxtPath = r"face\_detector\deploy.prototxt"

weightsPath = r"face\_detector\res10\_300x300\_ssd\_iter\_140000.caffemodel"

faceNet = cv2.dnn.readNet(prototxtPath, weightsPath)

maskNet = load\_model("mask\_detector.model")

print("[INFO] starting video stream...")

vs = VideoStream(src=0).start()

while True:

frame = vs.read()

frame = imutils.resize(frame, width=800)

(locs, preds) = detect\_and\_predict\_mask(frame, faceNet, maskNet)

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

(startX, startY, endX, endY) = box

(mask, withoutMask) = pred

label = "Mask" if mask > withoutMask else "No Mask"

color = (0, 255, 0) if label == "Mask" else (0, 0, 255)

label = "{}: {:.2f}%".format(label, max(mask, withoutMask) \* 100)

cv2.putText(frame, label, (startX, startY - 10),

cv2.FONT\_HERSHEY\_SIMPLEX, 0.45, color, 2)

cv2.rectangle(frame, (startX, startY), (endX, endY), color, 2)

cv2.imshow("Frame", frame)

key = cv2.waitKey(1) & 0xFF

if key == ord("q"):

break

cv2.destroyAllWindows()

vs.stop()

## APPENDIX-2

**SCREENSHOTS**

## 

## Figure A 2.1 Evaluating the Performance of Networks

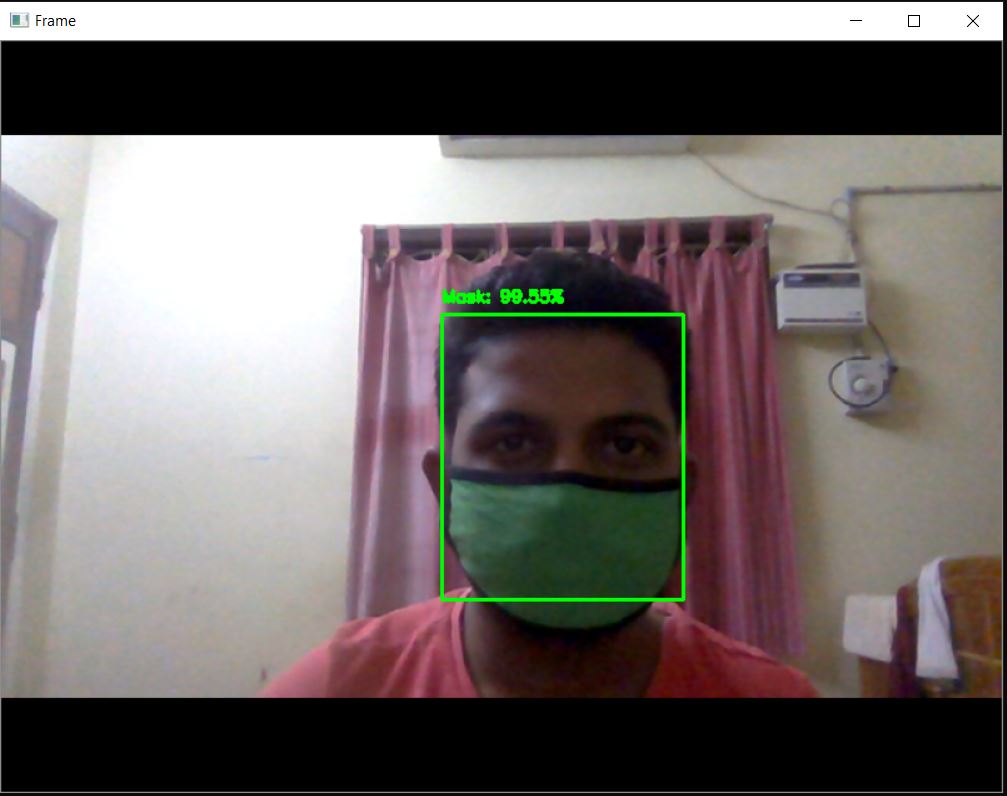


Figure A 2.2 Detection of Masked Face with 99.55%

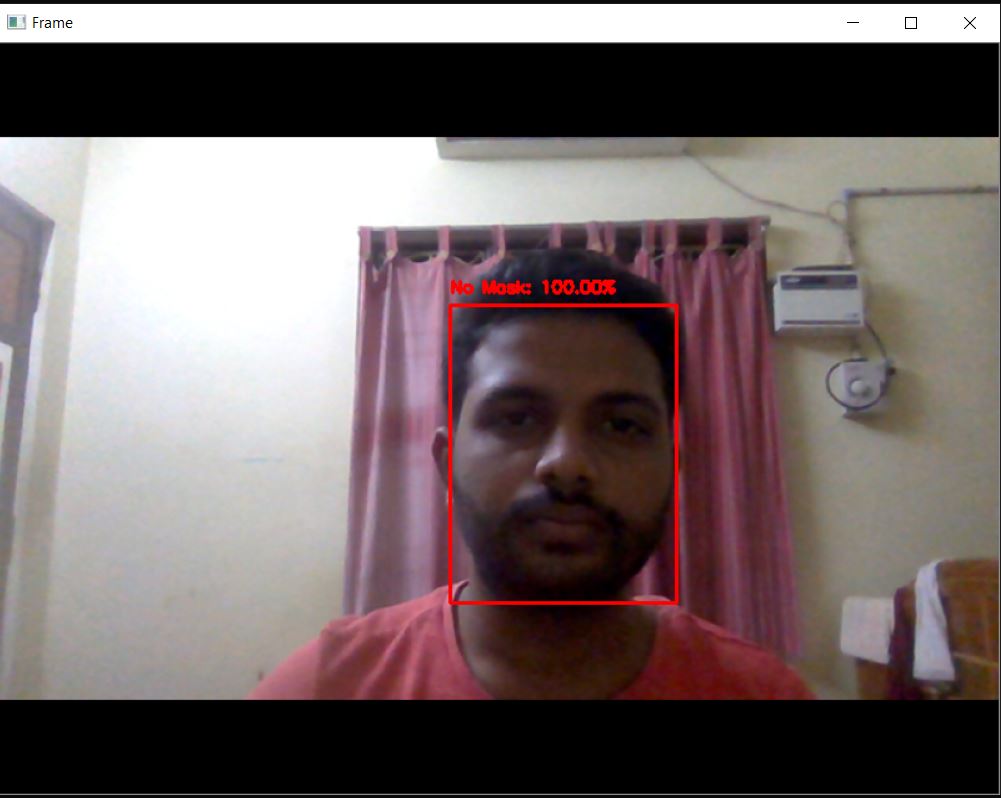


Figure A 2.3 Detection of Unmasked Face with 100.00%



Figure A 2.4 Detection of improper Masked Face as No Mask with 84.53%

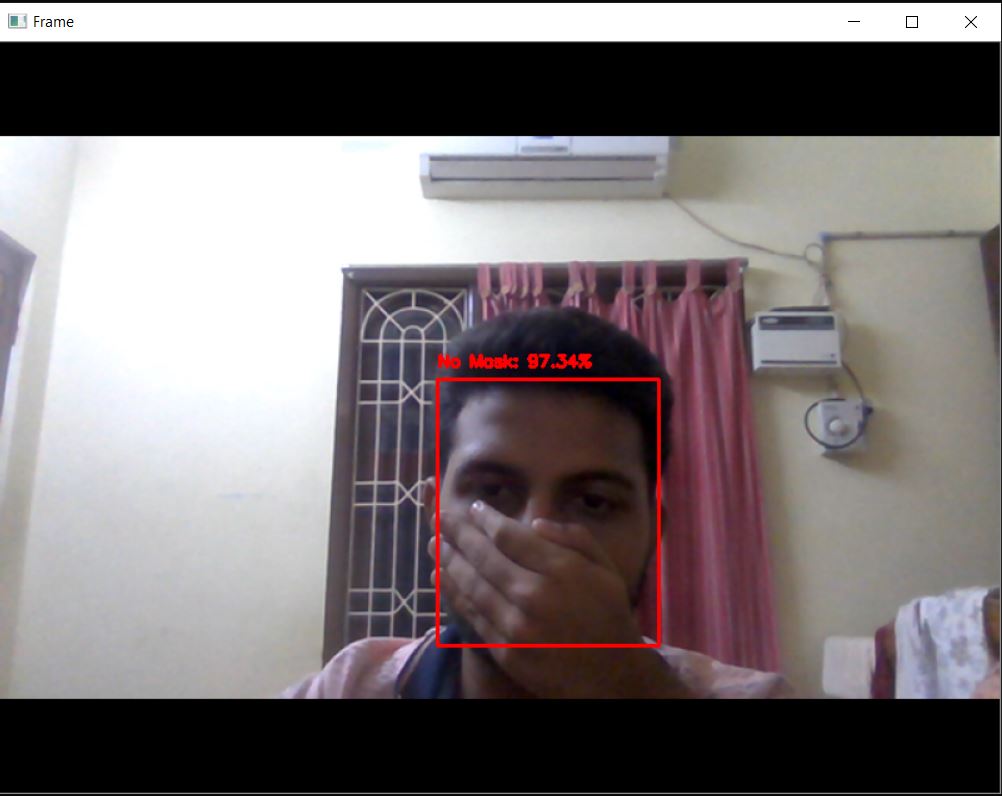


Figure A 2.5 Detection of Face covered by hands as No Mask with 97.34%

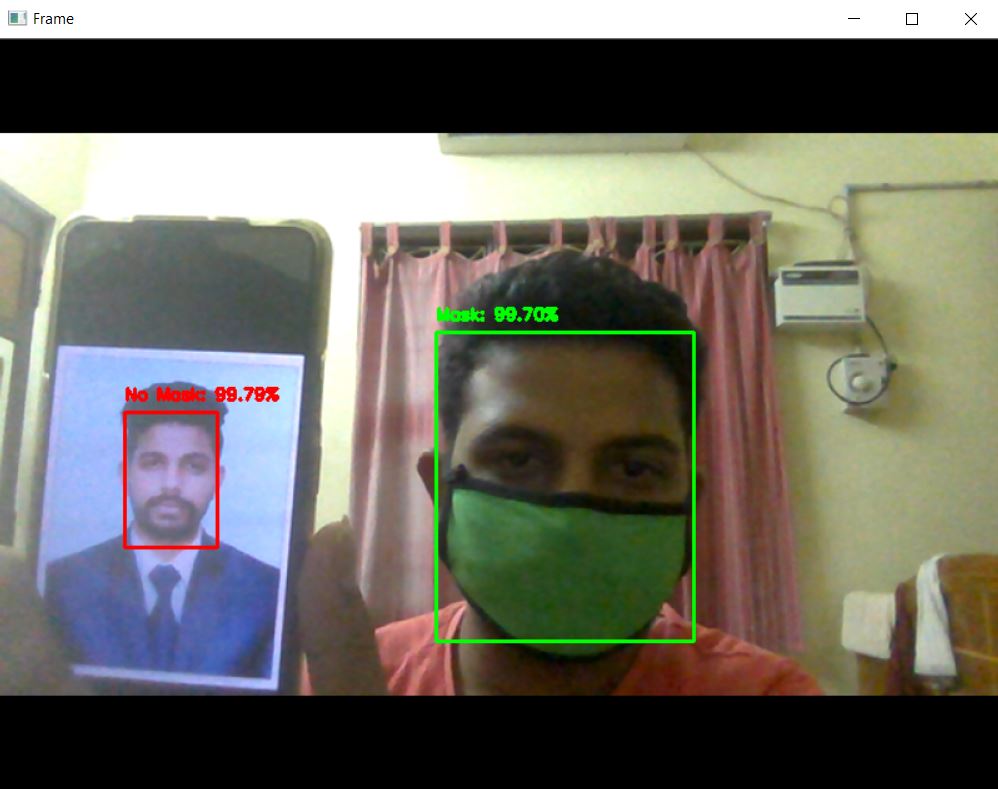


Figure A 2.6 Detection of Masked Face and Unmasked face both with 99.7%



## Figure A 2.7 Detection of all unmasked Faces above 99%

## REFERENCES

* + 1. M. M. Rahman, M. M. H. Manik, Saifuddin M, Jong-Hoon Kim, and M. M. Islam, “An Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network” 2020 IEEE International IEMTRONICS pp. 1-5.
    2. Mohamed Loey, Gunasekaran Manogaran, Mohamed Hamed N. Taha, and Nour Eldeen M. Khalifa, “Fighting against COVID-19: A novel deep learning model based on YOLO-v2 with ResNet-50 for medical face mask detection” Sustainable Cities and Society, Elsevier Ltd, published on November 12, 2020
    3. “OpenCV Basics: A Mobile Application to Support the Teaching of Computer Vision Concepts,” IEEE Transactions on Education (Volume: 63, Issue: 4, Nov. 2020)
    4. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, Google Inc. “Mobilenetv2: Inverted residuals and linear bottlenecks” – 2018 IEEE conference on computer vision and pattern recognition (pp. 4510-4520).
    5. C. Jagadeeswari, and M. Uday Theja, – “Performance Evaluation of Ingelligent Face Mask Detection System with various Deep Learning Classifiers” - 2018 IEEE
    6. L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, “Deep learning for generic object detection: A survey,” International journal of computer vision, vol. 128, no. 2, pp. 261–318, 2020.”