VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi – 590 014

DEPARTMET OF COMPUTER SCIENCE AND ENGINEERING

A Report on

COVID -19 FACE MASK DETECTION USING DEEP LEARNING

In partial fulfillment of Final Year Project

8th SEM COMPUTER SCIENCE AND ENGINEERING 2021-2022

Submitted by:

ABHISHEK M R	1GG19CS400
HARSHITHA T S	1GG18CS021
SAGAR N C	1GG19CS407
SANIYA BANU R	1GG19CS408

UNDER THE GUIDANCE OF

KOMALA K V

Asst professor, dept of CSE

GEC, Ramanagara.

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

GOVERNMENT ENGINEERING COLLEGE RAMANAGARA

GOVERNMENT ENGINEERING COLLEGE

Ramanagara-562159

Department of Computer Science & Engineering

CERTIFICATE

This is to certify that project entitled **“**COVID-19 FACE MASK DETECTION USINGDEEP LEARNING **”** is a bonafide work carried out by “ **ABHISHEK M R, HARSHITHA T S, SAGAR N C, SANIYA BANU R**” as a partial fulfillment for the award of Bachelor’s Degree in Computer Science and Engineering for **Project** as prescribed by **Visvesvaraya Technological University**, Belgaum for the year **2021-22**

Mrs. KOMALA K V
Asst.Prof & Guide,
Dept of CSE

Dr. VASANTH G
Professor & HOD,
Dept of CSE

Dr. G PUNDARIKA
Principal,
GECR

Name of the Examiners	Signature with Date
1. …………………………	……………………..
2. …………………………	……………………..

DECLARATION

We hereby declare that the project report entitled **"COVID-19 FACE MASK DETECTION USING DEEP LEARNING",** Submitted in partial fulfillment of the requirement for the award of the degree of **Bachelor of Engineering (B.E)** in **Computer Science and Engineering Department** is a record of bonafide project work carried on by us under the guidance of **"Mrs. KOMALA K V"** to **GOVERNMENT ENGINEERING COLLEGE RAMANAGAR**.
We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Signatures of the Candidates:

Abhishek M R (1GG19CS400)

ACKNOWLEDGEMENT

I consider it a privilege to whole heartedly express my gratitude and respect to each and every one who guided and helped me in the successful completion of this Internship Report.

I would greatly mention the enthusiastic influence provided by, **Mrs. KOMALA K V,** Assistant Professor as my Internship Guide, for her/his ideas and co-operation showed on me during my venture and making this Internship great success.

I am thankful to **Dr VASANTH G**, HOD, Department of Computer Science, for his co-operation and encouragement at all moments of `my approach.

I am very thankful to the principal **Dr G PUNDARIKA** for being kind enough to providing me an opportunity to work on a Internship in this institution.

I would also like to thank my parents and well-wishers as well as my classmates for their guidance and their kind co-operation.

Finally, it is my pleasure and happiness to the friendly co-operation showed by all the staff members of Computer Science Department, GECR.

ABSTRACT

In this pandemic government urges people to wear masks. If at all it is mandatory, some people are not properly wearing masks. This model is helpful to detect the people who are not wearing it properly, The corona virus COVID-19 pandemic is causing a global health crisis so the effective protection method is wearing a mask in public areas according to World Health Organization (WHO).Wearing a protective face mask has become a new normal. In the future, many public service providers will ask the customers to wear masks currently to avail of their services. Therefore, face mask detection has become a crucial task to help global society. The COVID-19 pandemic forced governments across the World to impose lockdown to prevent virus transmission. Reports indicate that wearing face masks while at work clearly reduces the risk of transmission. An efficient and economic approach of using AI to create a safe environment in manufacture setup. A face mask detection data set consist of people images mask and without mask images, OpenCV is used in real-time face detection from a live stream via our webcam. We will use the data set to build a COVID-19 face mask detector with computer vision using Python, OpenCV, and TensorFlow and Keras. Our goal is to identify whether the person on image/video stream is wearing a mask

TABLE OF CONTENTS

CHAPTER NAME **PAGE NO**

1. **INTRODUCTION.. 1**
   1.1 Existing System 3
   1.2 Proposed System..3
   1.3 Objectives 4
   1.4 Application.. 4
   1.5 Advantages.. 5
   1.6 Scope of Project 5
2. **LITERATURE SURVEY…...6**
3. **SYSTEM REQUIREMENTS AND SPECIFICATIONS 13**
   3.1 Hardware & Software requirements 14
   3.2 Functional requirements.. 14
   3.3 Non-Functional requirements 15
4. **SYSTEM DESIGN..16**
   4.1 Data flow diagram.. 17
   4.2 Use case diagram 19
   4.3 Class diagram.. 20
   4.4 Activity diagram 21
   4.5 Interaction diagram / Sequence diagram..22
5. **IMPLEMENTATION 23**
   5.1 Module 1: Mask detection from live video stream 28
   5.2 Module 2: Mask detection for uploaded image 31
   5.3 Caffe module..32
6. **TESTING AND CODING..33**
   6.1 Train module 40
   6.2 Code for mask detection through live video stream 44
   6.3 Results 46
7. **CONCLUSION 48**
8. **REFERENCES...50**

INTRODUCTION

INTRODUCTION:

According to the World Health Organization (WHO)’s official Situation Report Corona Virus disease 2019(COVID-19) has globally infected over 20 million people causing over 0.7million deaths. Individuals with COVID-19 have had a wide scope of symptoms reported to serious illness. The COVID-19 pandemic forced governments across the World to impose lockdown to prevent virus transmission. Reports indicate that wearing face masks while at work clearly reduces the risk of transmission. Respiratory problems like shortness of breath or difficulty in breathing is one of them. Some common human Corona Viruses that infect public around the world are NL63 AND 299E. Persons having respiratory problems can expose (anyone who is close contact with them) to infective beads. Surroundings of a tainted individual can cause contact transmission as droplets carrying virus may withal arrive on his adjacent surface. To curb respiratory viral ailment, including COVID-19, wearing a clinical mask is very necessary. The public should be aware of whether to put on the mask for source control or aversion of COVID-19. Potential points of interest of the utilization of masks lie in reducing vulnerability of risk from a noxious individual during the “pre-symptomatic” period and stigmatization of discrete persons putting on masks to restraints the spread of virus. WHO stresses on prioritizing medical masks and respirators for health care assistants. Therefore, face mask detection become a crucial task in present global society. Face mask detection involves detecting the location of the face and then determining whether it has a mask on it or not. The issue is “proximately connate” to general object detection to detect the classes of objects.

Face identification categorically deals with distinguishing a specific group of entities that is, face. The project project involves developing a deep learning model for face mask detection using python, Keras, OpenCV. We will train the model using Keras. The face detection system involves capturing people’s images using webcam, Face detection is carried out by calculating the location of the bounding box in the image, On face detection face Region of Interest(ROI) is extracted and facial landmarks is applied, This allows us to localize mouth, face and eyes**.**

EXISTING SYSTEM:

In the present environment going outside without the mask is a risk. And even the government also passed the strict rules that every citizen has to wear a mask while coming outside. But some are not following the rules, and it's not possible to identify them when they are in a crowded area. To detect those people, we are using face mask detection. The detection, which is using the existing model, is not much efficient while processing the images, and comparatively, it takes more parameters than other models. It is required to detect faster to stop them, but on the off chance that the pictures contain some level of tilt or turn, at that point, model normally experience issues in classifying the image.

PROPOSED SYSTEM:

The proposed system Centre around how to recognize the individual on picture/video transfer wearing a face mask with the assistance of PC vision and profound learning calculation by utilizing the OpenCV, Tensorflow, Keras, and Python library. Data at Source Most of the pictures were increased by OpenCV. The arrangement of pictures was at that point named "mask" and "no mask". The pictures that were available were of various sizes and resolutions, presumably extricated from various sources or from machines (cameras) of various resolutions. Data pre-processing ventures as referenced underneath was applied to all the raw input images to change over them into clean forms, which could be taken care of to a neural network model. Altering the size of an inserted image (256 x 256). Put in the color Computing (RGB) over the transmission. Scaling/Normalizing pictures utilizing the standard mean of Python work in loads. Centre trimming the picture with the pixel estimation of 224x224x3.Finally transform them into tensors. Detecting face mask. System includes Input face mask dataset. Training the dataset with python libraries, Rearrange the dataset to disk, Load dataset from disk. Detect faces from image/video stream and determine with or without mask and output the result.

OBJECTIVES:

The objectives of our project are:

- Recognize the individual on picture/video transfer wearing a face mask with the assistance of PC vision and profound learning calculation by utilizing the Python library.
- It ensures a lightweight representation that makes real-world masked face recognition a feasible task. Moreover, the masked regions vary from face to another, which leads to informative images from different sizes.
- It has to display the percentage of how much accurately the person wear a mask
- Provide an alarm if a person not wear a mask
- The proposed model can be incorporated with observation cameras to block the COVID-19 transmission by permitting the identification of individuals who are wearing face masks not wearing a facemask.
- We are using deep learning architecture to learn various important nonlinear features from the given samples.

APPLICATIONS:

- Ensuring Face Mask Detection
- sound Alert when not worn mask
- display the percentage of correctly wearing mask
- In Airports, Hospitals, and Offices the Face Mask Detection System can be used at airports to detect travellers without masks. Face data of travellers can be captured in the system at the entrance. If a traveller is found to be without a face mask, it plays a sound then security will stop that person.

ADVANTAGES

- Manual Monitoring is very difficult for officers to check whether the peoples are wearing mask or not. So in our technique, We are using web cam to detect people's faces and to prevent from virus transmission.
- It has fast and high accuracy.
- This system can be implemented in ATMs, Banks.
- We can keep peoples safe from our technique.
- It provides buzzer sound to wear mask.

SCOPE OF PROJECT:

This project can be used in schools, hospitals, banks, airports, and etc. as a digitalized scanning tool. The technique of detecting people's faces and segregating them into two classes namely the people with masks and people without masks is done with the help of image processing and deep learning.

LITERATURE SURVEY

LITERATURE SURVEY :

[1] ArjyaDas, Mohammad Wasif Ansari, RohiniBasak are saying that 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. Face mask detection involves in detecting the location of the face and then determining whether it has a mask on it or not. The issue is proximately cognate to general object detection to detect the classes of objects. Face identification categorically deals with distinguishing a specific group of entities i.e. Face It has numerous applications, such as autonomous driving, education, surveillance, and so on. This paper presents a simplified approach to serve the above purpose using the basic Machine Learning (ML) packages such as TensorFlow, Keras, OpenCV and Scikit-Learn.

[2] Pinki, Prof. SachinGarg says The Face Mask Detector can be used to analyze whether people are wearing face masks or not in a workplace as well as in a public area and carry out necessary actions to better deal with the pandemic. Automating the task will lead in effective actions taken in short time hence equipping us better to deal with the situation. This Face Mask Detector will make sure that the virus does not spread too much by taking necessary measures like Face Masks. Successfully identify the person on image/video stream wearing face mask or not. If the person doesn't wear a mask, the notification will be sent to the respected admin with the help of Python and deep learning algorithm by using the Convolutional Neural Network, Keras Framework and OpenCV.

[3] KarthikSrivathsa D S , Dr. A. Rengarajan, Nikhil Kumar H S are say that Corona Virus causes respiratory disappointments including injury of the heart liver, and kidneys, and its leadings to death. Most by far spoiled with the Corona virus diseases will experience delicate to coordinate respiratory affliction and recover without requiring uncommon treatment. For all this reason, one should close a face with mask. So they presented a detecting of face mask model that depends on PC vision and deep learning. They proposed a model that can be incorporated with observation cameras to block the COVID-19 transmission by permitting the identification of individuals who are wearing face masks not wearing a facemask. That model is a combination deep learning and traditional machine learning techniques with the tensor flow, Keras, and OpenCV. They are introduced their own dataset and denoted that as MAFA. That dataset consists of various face orientations and occlusion degrees, and the face to be closed at one side in each face. LLE-CNN has been used to identify the masked face. Their proposed system centre's around how to recognize the individual on picture/video transfer wearing a face mask with the assistance of PC vision and profound learning calculation by utilizing the OpenCV, Tensorflow, Keras, and Python library. Most of their pictures were increased by OpenCV. The arrangement of pictures was at that point named "mask" and "no mask". The pictures that were available were of various sizes and resolutions, presumably extricated from various sources or from machines (cameras) of various resolutions. That particular model could be utilized as a utilization case for edge examination. Moreover, that technique accomplishes cutting edge results on a public face mask dataset. By the advancement of face mask recognition, they can distinguish if the individual is wearing a face cover, and permit their entrance would be of incredible assistance to society.

[4] **MadhuraInamdar, NinadMehendale** are say Deep learning technique has been useful for big data analysis and has its applications in computer vision, pattern and speech recognition, etc. their work focuses on some commonly implemented deep learning architectures and their applications. The auto encoder, the Convolutional neural network, Boltzmann machine, the deep belief networks are the networks that are presented in detail. Deep learning can be used in unsupervised learning algorithms to process the unlabeled data. A CNN model for speedy face detection has been introduced by Li et al. that evaluates low resolution an input image and discards non-face sections and accurately processes the regions that are at a greater resolution for precise detection. Calibration nets are used to stimulate detection. Their proposed method was least complex and it does not demand segmentation, bounding-box regression, or SVM classifiers and can recognize faces at numerous angles. A novel data augmentation approach for mask detection from speech was proposed by Ristea et al. that could be used for communication amongst surgeons, used in forensic fields or infectious diseases like corona virus. Their work distinguishes face masks from images and live video streams. On training the model using Facemask net, they got an accuracy of 98.6 %. This classifiers was then implemented to images and live video streams. The faces were recognized in images and videos and these faces were extracted. Then, their face mask classifier was applied to achieve the required results. The green and yellow rectangular frame respectively represents the detected face and mask. Their face mask identifier is least complicated in structure and gives instantaneous results and hence can be used in CCTV footages to identify whether a person is wearing a mask correctly so that he does not pose any hazard to others.

[5] Prof. Dr. ChristophLippert, Benjamin Bergner, RazaAli, SaniyaAdee ,Akhyar Ahmed , MdHasanShahriar, MdShohelMojumder are say The basic aim of their project was to detect the presence of a face mask on human faces on live streaming video as well as on images. They have used deep learning to develop their face detector model. That architecture used for the object detection purpose was Single Shot Detector (SSD) because of that's good performance accuracy and high speed. Two stage detectors use two neural networks to detect objects, for instance region-based Convolutional neural networks (R- CNN) and faster R-CNN. The first neural network was used to generate region proposals and the second one refines those region proposals; performing a coarse-to- fine detection. That strategy results in high detection performance compromising on speed. The seminal work R- CNN is proposed by R. Girshick et al. R-CNN uses selective search to propose some candidate regions which may contain objects.

[6] SnehaSen, Dr. Harish Patidar are says proposed a mask detection system for the health care personal inside the operation theatre. As the health care personal need to wear a mask in the operation theatre and their proposed system will alert for any personal not wearing the mask. There were used two detection system for face and medical mask wearing. Their system achieved almost 90% recall and less than 5% of false positive rate. They had worked for the medical mask detection from the images that are taken from 5m distance by cameras the authors have worked on the masked face detection from the video. The masked person was detected in that presented approach and mainly 4 steps are performed for the detection that are estimation of distance between camera and person, detection of eye line, detection of part of face and detection of eye. They have analyzed their algorithm on various video surveillance systems and achieved a fine accuracy. That project presents a model for masked face detection for the security purpose. For that they had presented a special cascade CNN which works on three layers of CNN for the detection of masked face. Also, that authors have worked on the self created MASKED FACE dataset as the already present masked face dataset was not that much sufficient to evaluate algorithms more precisely. Their proposed CNN model worked well and the detection of masked faces was done accurately .One of the major precautions is to wear mask for the stopping of the spread of the respiratory droplets of infected peoples through cough or sneeze as well the healthy people should be covered with mask. So they had presented an approach that uses deep learning algorithm and the framework of MObileNetV2 is used for implementation along with the PyTorch and OPENCV of python. The results state that the proposed model is capable of detecting the peoples with or without masks from the images as well from the video streams.

[7] Mohammad MarufurRahman, Md.MotalebHossenManik, Md.Milon Islam, SaifuddinMahmud, Jong-Hoon Kim are says that Their project aims was designing a system to find out whether a person is using a mask or not and informing the corresponding authority in a smart city network. Firstly, CCTV cameras ware used to capture real-time video footage of different public places in the city. From that video footage, facial images ware extracted and those images are used to identify the mask on the face. The learning algorithm Convolutional Neural Network CNN) was used for feature extraction from the images then those features ware learned by multiple hidden layers. Whenever the architecture identifies people without face mask that information was transferred through the city network to the corresponding authority to take necessary actions. Their proposed system appraised promising output on data collected from different sources. They also represented a system that can ensure proper enforcement of the law on people who were not following basic health guidelines in this pandemic situation. Their project presented a system for a smart city to reduce the spread of corona virus by informing the authority about the person who was not wearing a facial mask that is a precautionary measure of COVID-19. The motive of their work comes from the people disobeying the rules that are mandatory to stop the spread of corona virus. That system contains a face mask detection architecture where a deep learning algorithm was used to detect the mask on the face. To train the model, labeled image data are used where the images were facial images with masks and without a mask.

[8] Md. SanzidulIslam, Md. RafiuzzamanBhuiyan, SharunAkterKhushbu are says that They had attained that people who wear face masks or not, that’s trained by the face mask image and non face mask image. Under the experimental conditions, real time video data that finalized over detection, localization and recognition. They said that computer vision learning is the actual field to identify the image, convert descriptive image, output analyze and machine acquiring. According to image detection that was identified by the numeric number for ability to understand humans. And Vision learning is deeply related to YOLO for any type of image detection. Furthermore, YOLO detection is flexible in any place whether group into family members, colleagues and friends in a walk. Declared by the WHO that a potential speech by maintaining distance and wearing a mask is necessary. Wearing a mask captured by the image detection where the machine can cover and translate only the mouth portion of the face part. Computer vision is a following section of Deep learning particularly an area of convolution neural network (CNN) Added with one main thing is CNN supports very high configuration Graphic Processing Units (GPU) thus as real time image or video extraction of visualization.

[9] **Jagadeeswari, M. UdayTheja** are say that Their proposed System can be of great importance at airports to detect travelers without masks. Traveler's data can be captured as videos in the system at the entrance. Any traveler found to be without a face mask, an alarm alerting the airport authorities was sent so that they could take quick action. And that system can integrated with CCTV cameras and that data may be administered to see if their staff is wearing masks. If some health worker is found without a mask, they can receive a reminder notification to wear a mask. That system can help in maintaining safety standards to prevent the spread of Covid-19 or any such air borne disease. If some employee is not wearing a mask, they can receive a reminder notification to wear mask.

[10] **Vinitha, Velantina** are say that Their proposed system focused on how to identify the person on image/video stream wearing face mask with the help of computer vision and deep learning algorithm by using the OpenCV, Tensor flow, Keras and PyTorch library. Face Mask Detection in webcam stream: The flow to identify the person in the webcam wearing the face mask or not. The process is two-fold. 1. To identify the faces in the webcam 2.Classify the faces based on the mask identify the Face in the Webcam: To identify the faces a pre-trained model provided by the OpenCV framework was used. The model was trained using web images. OpenCV provides 2 models for this face detector: We used OpenCV, tensor flow, Keras, Pytorch and CNN to detect whether people were wearing face masks or not. The models were tested with images and real-time video streams. The accuracy of the model is achieved and, the optimization of the model is a continuous process and we are building a highly accurate solution by tuning the hyper parameters.

[11] **Kinjal Goswami, A.M Sowjanya** are says Their Proposed Facemask detector model aims at detecting whether a person is wearing a mask or not. The results were tested using a live video stream or when an image has been given as an input to the face mask detector. Some conventional facial recognition techniques like edge detection, template matching, color segmentation or image thresholding become ineffective in many cases as detecting a masked face image was a complex task. Therefore, that becomes very important to help improve the detection process images. Their research area was about face mask detection a little information about face mask detection was given ahead. Face mask detection task identifies whether a person is wearing a mask or not. Therefore, they had built a real time face mask detection model using deep learning that detects faces with or without mask and is deployed with OpenCV, TensorFlow and Keras. This face mask detection model has achieved 95% accuracy. Their particular model can be deployed at various places like the hospitals, offices, airports to name few, as this model detects whether a person is wearing the face mask or not, as that is one of the ways to help stop the spread of virus from one another during this pandemic and by monitoring the placement of the face mask on the face, we can make sure that an individual wears it the right way.

[12] Aniruddha Srinivas Joshi, Shreyas Srinivas Joshi, GouthamKanahasabai, RudrakshKapil, Savyasachi Gupta are says they developed a deep learning methodology to detect masked faces using LLE-CNNs, which outperforms state-of-the-art detectors by at least 15%. In the given work, the authors introduced a new dataset called Masked Faces (MAFA), containing 35,806 images of masked faces having different orientations and occlusion degrees. The proposed LLE-CNNs consist of three modules - proposal module, embedding module and verification module. Their proposal module first combines two CNNs to extract candidate facial regions from the input image and represents them with high dimensional descriptors. After that, the embedding module is turns these descriptors into similarity based descriptors using Locally Linear Embedding algorithms and dictionaries trained on a set of faces, comprised of masked and unmasked images. Finally, the verification module is used to identify candidate facial regions and refine their positions with the help of classification and regression tasks. That proposed framework aims to detect whether people in the video footage of a public area are wearing face masks or not. In order to do so, we first detect the face of the person and then determine if a facial mask is present on the face. That is to be noted that the terms ‘face mask’ and ‘facial mask’ are used interchangeably throughout this work. A highly effective face detection model was used for obtaining facial images and cues. A distinct facial classifier was built using deep learning for the task of determining the presence of a face mask in the facial images detected. The resulting approach was robust and is evaluated on a custom dataset obtained for this work. The proposed approach was found to be effective as it portrayed high precision, recall, and accuracy values on the chosen dataset which contained videos with varying occlusions and facial angle

SYSTEM REQUIREMENTS SPECIFICATION

SYSTEM REQUIREMENTS SPECIFICATION:

Hardware Requirements

Processor	:	2.5 gigahertz (GHz) frequency or above.
RAM	:	A minimum of 4 GB of RAM.
Hard disk	:	A minimum of 20 GB of available space.
Monitor	:	Minimum Resolution 1024 X 768.

Software Requirements

Operating System	:	Windows 7 and above.
Programming language	:	Python 2.7 and above
Platform	:	IDLE 3.8 and Anaconda prompt
Supporting libraries	:	tensorflow,sklearn,numpy,imutils, and matplotlib etc.

Functional Requirements:

Functional requirements describe the system functionality, while the nonfunctional requirements describe system properties and constraints. Functional requirements capture the intended behaviour of the system. This behaviour may be expressed as services, tasks, or the functions the system is required to perform. This lays out important concepts and discusses capturing functional requirements in such a way they can drive architectural decisions and be used to validate the architecture. Features may be additional functionality, or differ from basic functionality along some quality attribute. In the proposed system, concert assesses the compliance of a workflow by analysing the five established elements. required to check for the rule adherence in workflows: activities, data, location, resources, and time limits. A rule describes which activities may, must or must not be performed on what objects by which roles. In addition, a rule can further prescribe the order of activities i.e. which activities have to happen before or after other activities.

Non-Functional Requirements:

1. **Security**

- System needs to control the user access and session
- It needs to store the data in a secure location and stored in a secure format
- It requires a secure communication channel for the data.

2. **Concurrency and Capacity**

System should be able to handle multiple computations executing simultaneously, and potentially interacting with each other.

3. **Performance**

Performance is generally perceived as a time expectation. This is one of the most important considerations especially when the project is in the architecture phase.

4. **Reliability**

It is necessary to ensure and notify about the system transactions and processing as simple as keep a system log will increase the time and effort to get it done from the very beginning. Data should be transferred in a reliable way and using trustful protocols.

5. **Maintainability**

Well-done system is meant to be up and running for long time. Therefore, it will regularly need preventive and corrective maintenance. Maintenance might signify scalability grow and improve the system features and functionalities.

6. **Usability**

End user satisfaction and acceptance is one of the key pillars that support a project success. Considering the user experience requirements from the project conception is a win bet, and it will especially save a lot of time at the project release, as the user will not ask for changes or even worst misunderstandings.

7. **Documentation**

All projects require a minimum of documentation at different levels. In many cases the users might even need training on it, so keeping good documentation practices and standards will do this task spread along the project development; but as well this must be establish since the project planning to include this task in the list

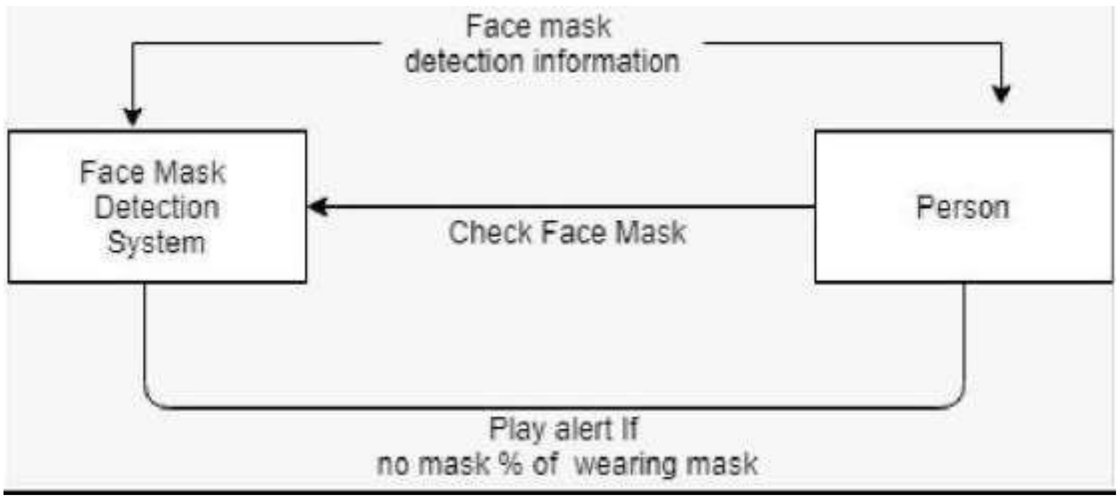
SYSTEM DESIGN

SYSTEM DESIGN

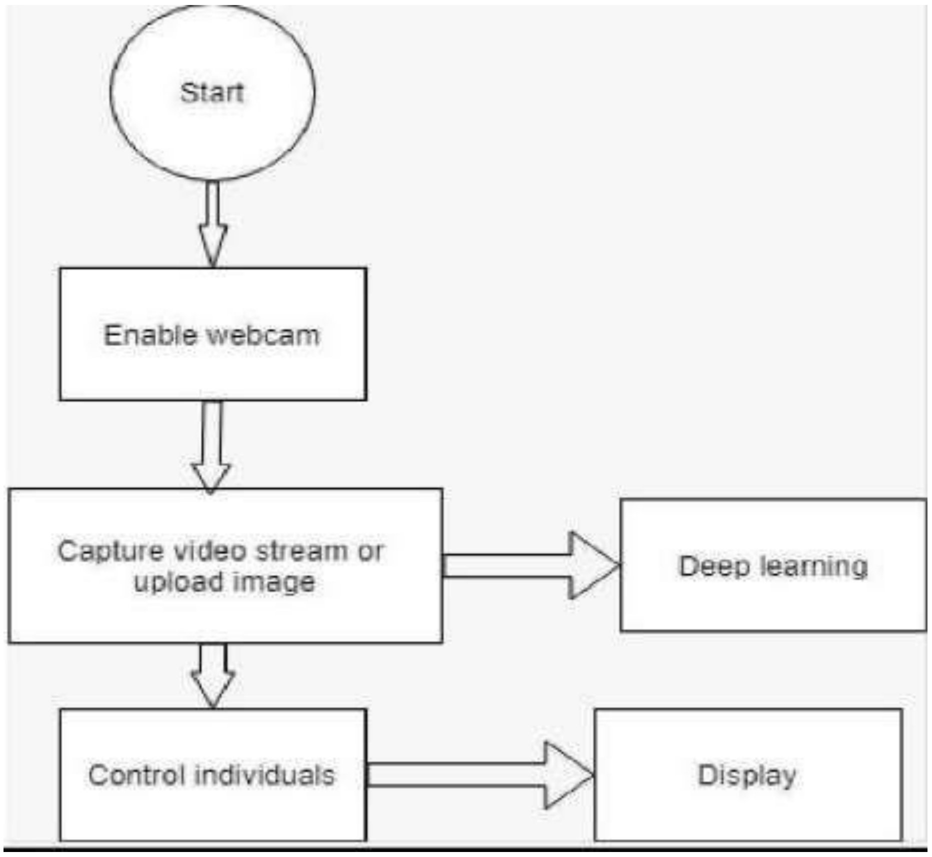
Data Flow Diagram:

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out onthis data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

Level 0


Face mask detection information
Face Mask Detection System
Check Face Mask
Person
Play alert If no mask % of wearing mask

Level 1

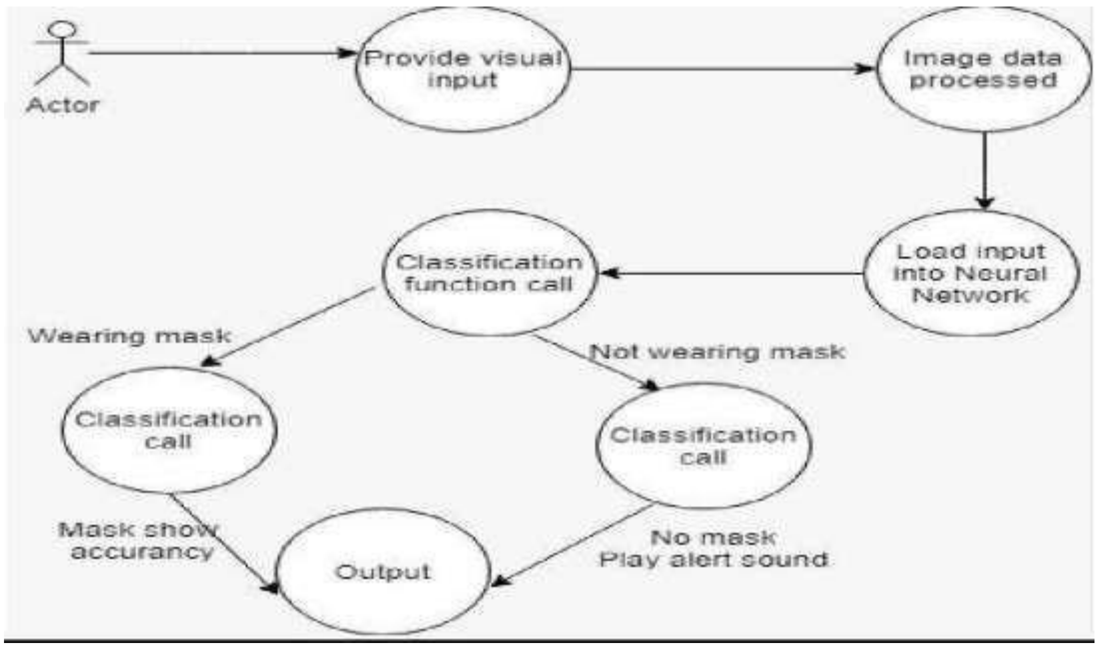


In level 0 when a person face captured through the web camera our trained model will capture the face using head model and extract the features based on cafeemodel and image will resized. Then our model will compare the image into the dataset which we used to train our model and model will display the result that person wearing a mask or not. If a person not wearing mask then it play a alert sound and display how much accurately person wearing mask.

In level 1 the face will be detected and with the help of model prediction will be done that whether a person has worn a mask or not. For camera operations OpenCV library will be used. First the model will be loaded. With the use of VideoStream() the camera will be open. With the use of some methods in OpenCV the face will be detected. And with help of the model file prediction will be done that whether the person has worn a mask or not.

Use Case Diagram:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

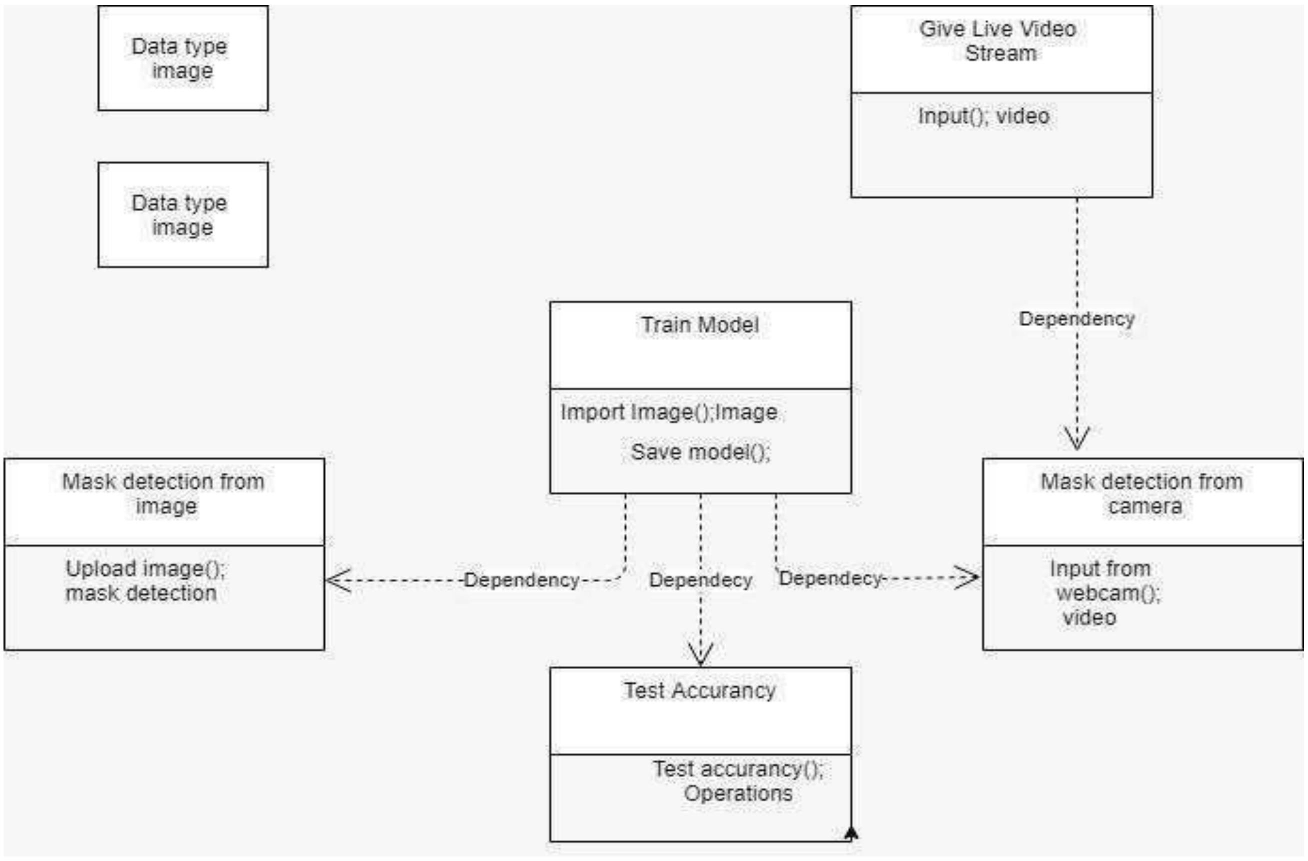


When a image is loaded or video is captured the following process will occur:

1. Image acquisition: to acquire a digital image
2. Image pre-processing: to improve the image in ways that increases the chances for success of the other processes.
3. Image segmentation: to partitions an input image into its constituent parts of objects.
4. Image description: to extract the features that result in some quantitative information of interest of features that are basic for differentiating one class of objects from another.
5. Image recognition: to assign a label to an object based on the information provided by its description.
6. Image segmentation: to convert the input data to a from suitable for computer processing. Finally it detects whether a person is wear

Class Diagram:

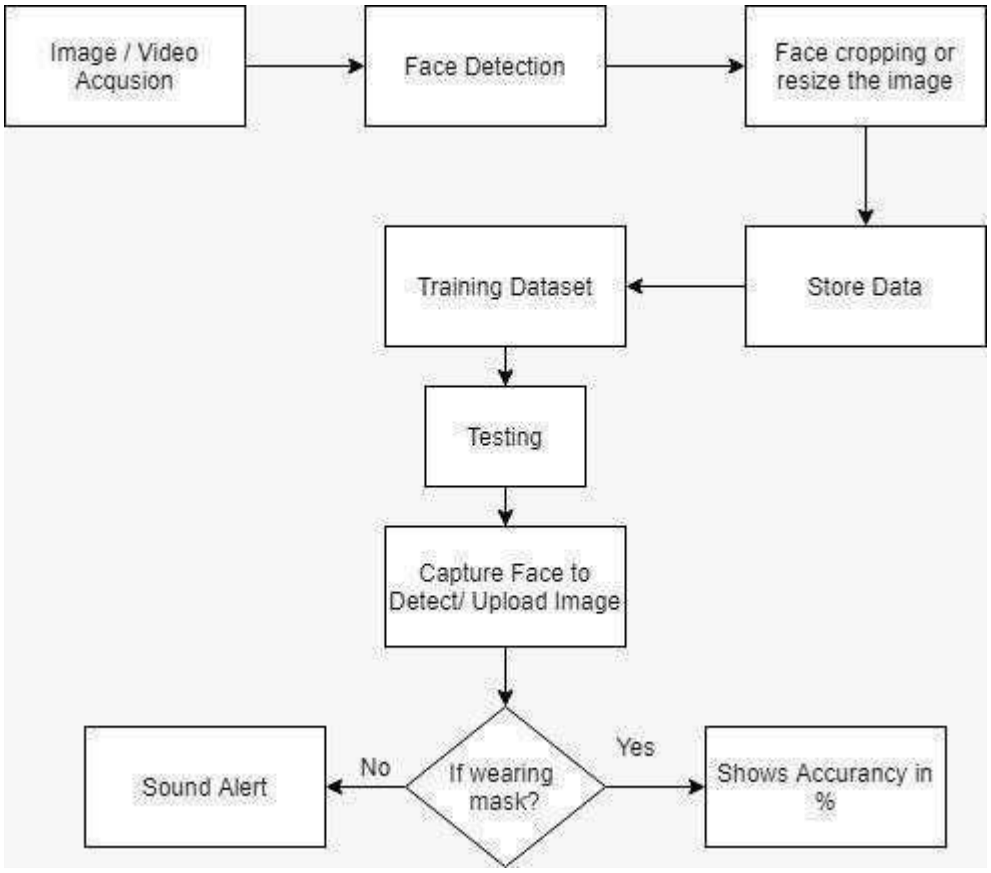
In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



The framework works as a 2-factor authentication in order to allow a person who wants to enter into the organisation. As a first step we are performing the face mask detection, where we are using a web cam to capture the image of the person who wants to enter into the organisation. The captured image is forwarded to the face mask detection algorithm, where the trained dataset is used to compare the face of the person and determine whether the person is wearing a mask or not. Based on the result, if the person is wearing a mask, the face is bounded by a green-colored rectangular box with the accuracy. If the person is not wearing a mask, then the face is bounded by a red- colored rectangular box with accuracy, and also it leads to a Security Alert sound, which indicates there is a person near the gate not wearing a mask.

Activity Diagram:

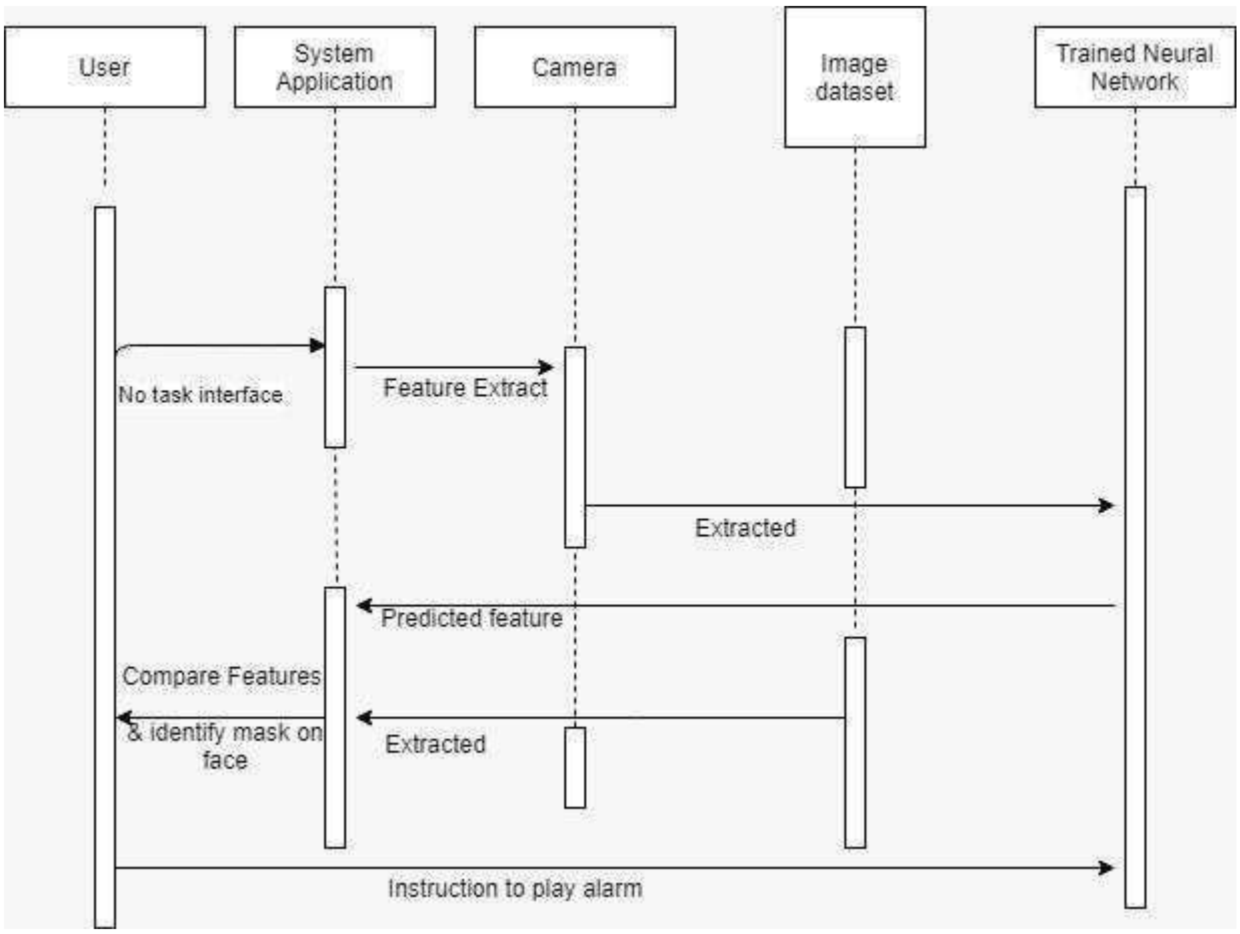
Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



First will upload the image or capture the live video then image will cropped and resized in 224*224 pixel then model will detect the face and identify the pattern of image using the different layers of Keras then compare with trained data this action will perform by using relu and softmax activation functions of cefeemodel. When we test accuracy of this created model we got 97% accuracy. After testing the model will capture video or we can upload image to check whether the person wearing mask or not. If a person not wear a mask then it will play alert sound and shows accuracy of wearing mask.

Interaction Diagram / Sequence Diagram:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagram.



Through the system application camera will capture the person image, our trained model will be extract the features from dataset .then based on training model will predict feature of person and compare the feature with given image then produce result as mask or no mask.

IMPLEMENTATION

IMPLEMENTATION:

The implementation of the proposed approach is presented here and the various steps followed and the data used for implementation is stated. Also, the selection of each and every library, framework, algorithm and the dataset.

Data at Source

Here for out experiment we have downloaded the raw images from the github on Image Search and the task of augmentation on this image is done by the OpenCV. “Mask” and “No Mask” are the tags given to the images at the start itself. The size and resolutions of the taken images are varied because the they are taken from various devices having different configurations. We have used total 1578 images that we classified as with_mask of 787 images and without_mask of 791 images.

Data Preprocessing

Here the preprocessing steps have been stated which are performed for making the images noise free and make them clear for detection, and this preprocessed image could be given to the neural networks model as input. Resizing the input image (400 x 400) .Applying the color filtering (RGB) over the channels (Our model MobileNetV2 supports 2D 3 channel image) .Scaling / Normalizing images using the standard mean of keras build in weights .Center cropping the image with the pixel value of 224x224x3 .Finally Converting them into tensors .

Deep Learning Frameworks

For the implementation of such deep learning network the following options can be seen as per availability.

1. TensorFlow
2. Keras
3. Caffee
4. OpenCV

MobilenetV2:

MobileNetV2 is a state of the art for mobile visual recognition including classification, object detection and semantic segmentation. This classifier uses Depth wise Separable Convolution which is introduced to dramatically reduce the complexity cost and model size of the network, and hence is suitable to Mobile devices, or devices that have low computational power. In MobileNetV2, another best module that is introduced is inverted residual structure. Non-linearity in narrow layers is deleted. Keeping MobileNetV2 as backbone for feature extraction, best performances are achieved for object detection and semantic segmentation.

Tensorflow:

TensorFlow is an open source framework developed by Google researchers to run machine learning, deep learning and other statistical and predictive analytics workloads. Like similar platforms, it's designed to streamline the process of developing and executing advanced analytics applications for users such as data scientists, statisticians and predictive modelers. The TensorFlow software handles data sets that are arrayed as computational nodes in graph form. The edges that connect the nodes in a graph can represent multidimensional vectors or matrices, creating what are known as tensors. Because TensorFlow programs use a data flow architecture that works with generalized intermediate results of the computations, they are especially open to very large-scale parallel processing applications, with neural networks being a common example.

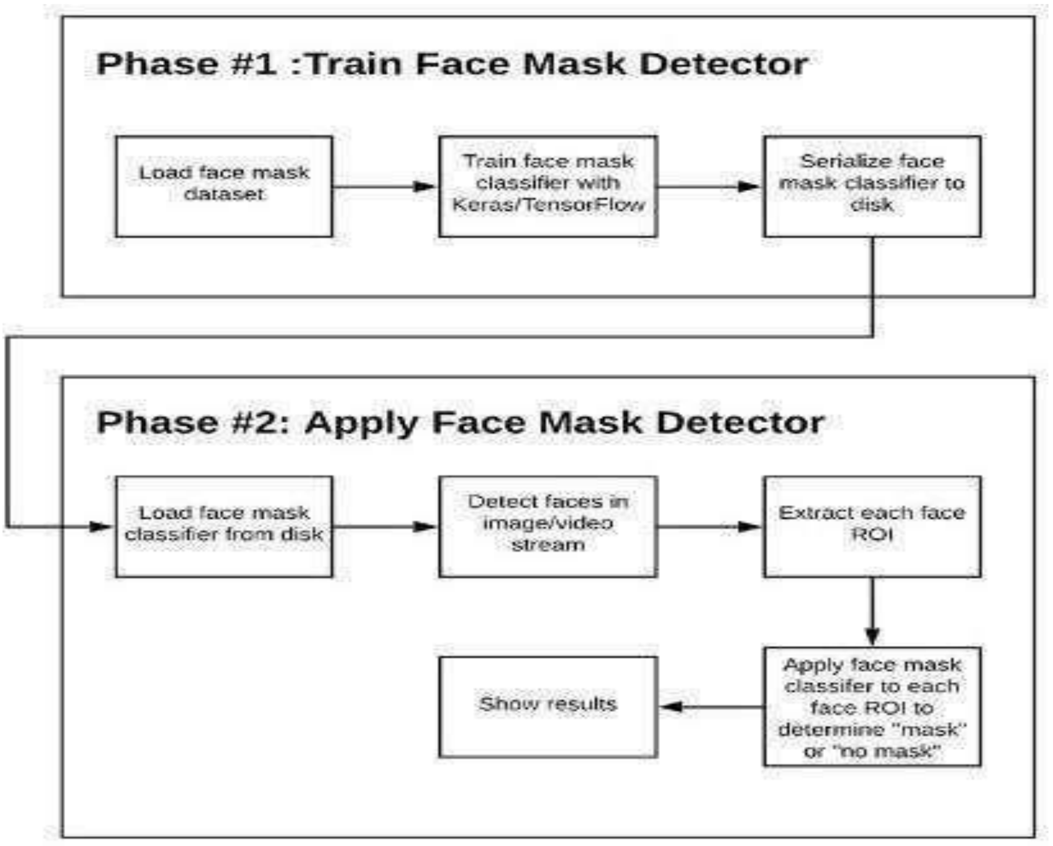
In order to train a custom face mask detector, we need to break our project into two distinct phases, each with its own

Training: Here we’ll focus on loading our face mask detection dataset from disk, training a model (using Keras/TensorFlow) on this dataset, and then serializing the face mask detector to

Disk Deployment: Once the face mask detector is trained, we can then move on to loading the mask detector, performing face detection, and then classifying each face as with_mask or without_mask.

There should be three Python scripts in this Project:

- train_mask_detector.py: Accepts our input dataset and fine-tunes MobileNetV2 upon it to create our mask_detector.model. A training history plot.png containing accuracy/loss curves is also produced
- detect_mask_image.py: Performs face mask detection in static images
- detect_mask_video.py: Using your webcam, this script applies face mask detection to every frame in the stream


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

Data Augmentation:

Loading the MobilNetV2 classifier (we will fine-tune this model with pre-trained ImageNet weights)Building a new fully-connected (FC) head

Pre-Processing:

Loading image data

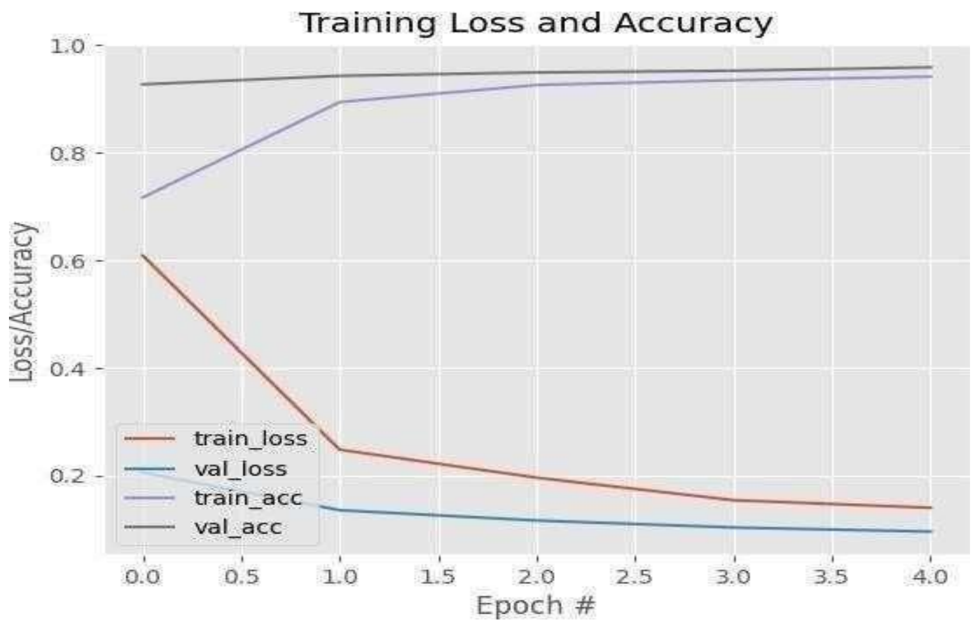
We'll use scikit-learn (sklearn) for binarizing class labels, segmenting our dataset, and printing a classification report.

My imutils paths implementation will help us to find and list images in our dataset. And we'll use matplotlib to plot our training curves.

Our command line arguments include:

Dataset : The path to the input dataset of faces and and faces with masks.

Plot : The path to your output training history plot, which will be generated using matplotlib. Model : The path to the resulting serialized face mask classification model.



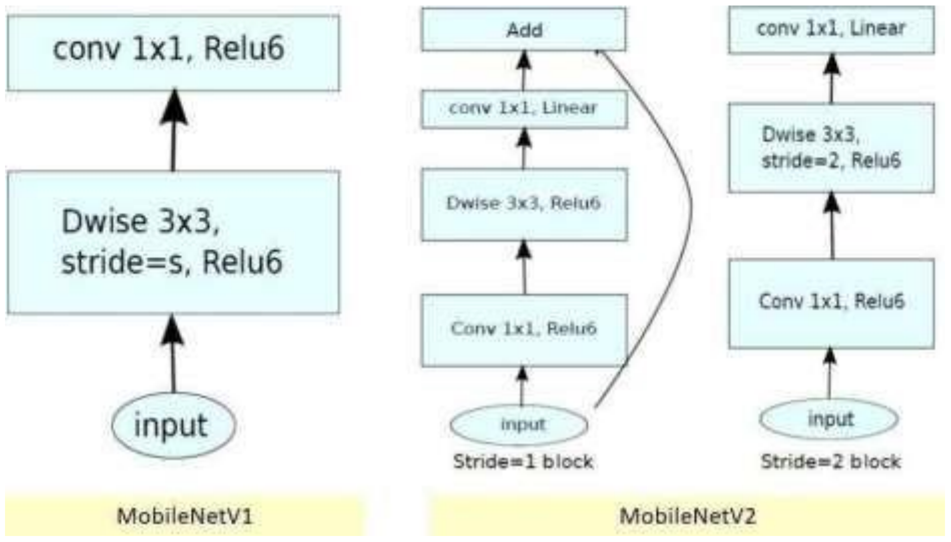
1) Training the COVID-19 face mask detector with Keras/TensorFlow
2) We are now ready to train our face mask detector using Keras, TensorFlow, and Deep Learning.
3) Implementing our COVID-19 face mask detector for images with OpenCV
4) Apply our face mask detector to classify the face as either with_mask or without_mask.
5) Implementing our COVID-19 face mask detector in real-time video streams with OpenCV

This function detects faces and then applies our face mask classifier to each face ROI. Such a function consolidates our code — it could even be moved to a separate Python file if you so choose. Our detect_and_predict_mask function accepts three parameters:

frame: A frame from our stream faceNet: The model used to detect where in the image faces are maskNet: Our COVID-19 face mask classifier model Inside, we construct a blob, detect faces, and initialize lists, two of which the function is set to return. These lists include our faces(i.e., ROIs), locs(the face locations), and preds(the list of mask/no mask predictions)

Module 1: Mask Detection From Live Video Stream

In proposed model MobileNetV2 classifier has been used. It is a convolutional neural network architecture that seeks to perform well on mobile devices. Where MobileNet is a type of convolutional neural network which was designed for mobile and embedded vision applications. They are based on a streamlined architecture that uses depthwise separable convolutions to build lightweight deep neural networks that can have low latency for mobile and embedded devices.The figure 3 below show the basic architecture of MobileNetV2. Where a stride is a part of cnn(convolutional neural network)used for tuning the compression of images and video data.



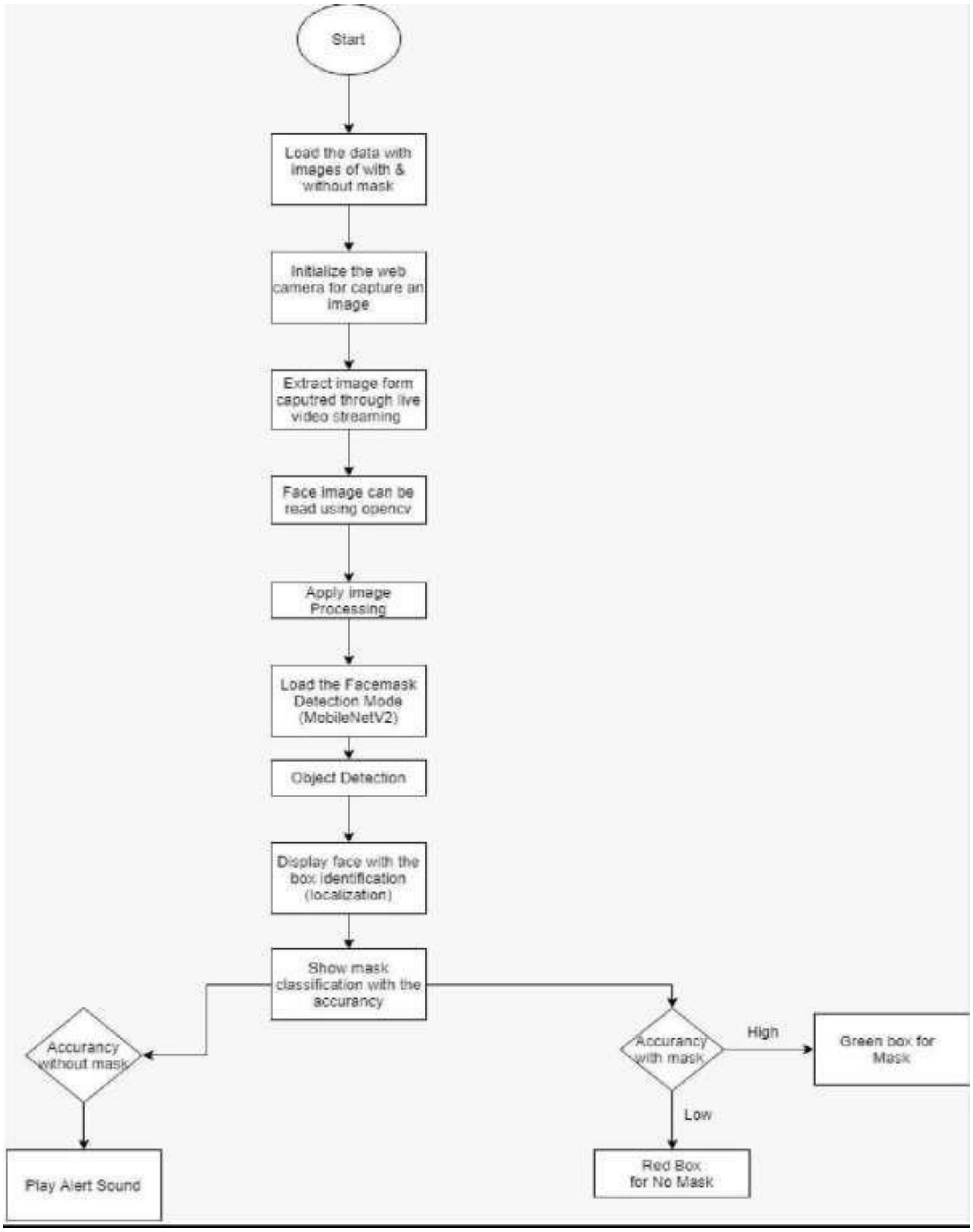
MobileNetV2 has 3 layer’s for both types of blocks where the first layer is 1×1 convolution with ReLU6.the second layer is the depthwise convolution layer and the third layer is another 1×1 convolution but without any non-linearity. Architecture of MobileNetV2 has been built by taking ideas from MobileNetV1,as depth wise separable convolution as an important building block. But the only difference is that version 2 of MobileNet i.e. MobileNetv2 has two new features and they are linear bottlenecks between the layers and shortcut connections between the bottlenecks. The figure 4 below shows the basic architectureof MobileNetV2. opencv, keras, tensorflow and mobilenetv2.We first do a data collection and arrange it in the following manner: we have the data which is divide into two parts i.e. train and test which are further sub divide into two parts known as with mask and without masks. Further we use MobileNetV2 classifier along with opencv to further implement the facemask detection system.

ALGORITHM:

Require: detector of faces and face features.
Ensure: Web camera enabled , frontally facing the web camera.

```
1: Procedure videostream
2: // basis principle
3: for (each frame ∈videostream) do
4:         if (detected face) then
5:               if (¬ detected nose) then
6:               display(“ mask with green color ”)
7:         else
8:         display(“no mask with red color”, play alert sound)
9:               end if
10:        end if
11: end for
12: end procedure
```

Flow Diagram:


Start
Load the data with images of with & without mask
Initialize the web camera for capture an image
Extract image form caputred through live video streaming
Face image can be read using opencv
Apply image Processing
Load the Facemask Detection Mode (MobileNetV2)
Object Detection
Display face with the box identification (localization)
Show mask classification with the accurancy
Accurancy without mask
Play Alert Sound
Accurancy with mask
High
Green box for Mask
Low
Red Box for No Mask

Module 2: Mask Detection For Uploaded Image

Most of the pictures were increased by OpenCV. The arrangement of pictures was at that point named "mask" and "no mask". The pictures that were available were of various sizes and resolutions, presumably extricated from various sources or from machines (cameras) of various resolutions. Pre-processing ventures as referenced underneath was applied to all the raw input images to change over them into clean forms, which could be taken care of to a neural network model. Altering the size of an inserted image (256 x 256). Put in the color Computing (RGB) over the transmission. Scaling/Normalizing pictures utilizing the standard mean of Python work in loads. Centre trimming the picture with the pixel estimation of 224x224x3. Finally transform them into tensors. To recognize the individual on picture transfer wearing a face mask with the assistance of PC vision and profound learning calculation by utilizing the Python library

ALGORITHM:

Step 1: input face mask dataset

Step 2: Train the dataset with python libraries

Step 3: rearrange the dataset to disk

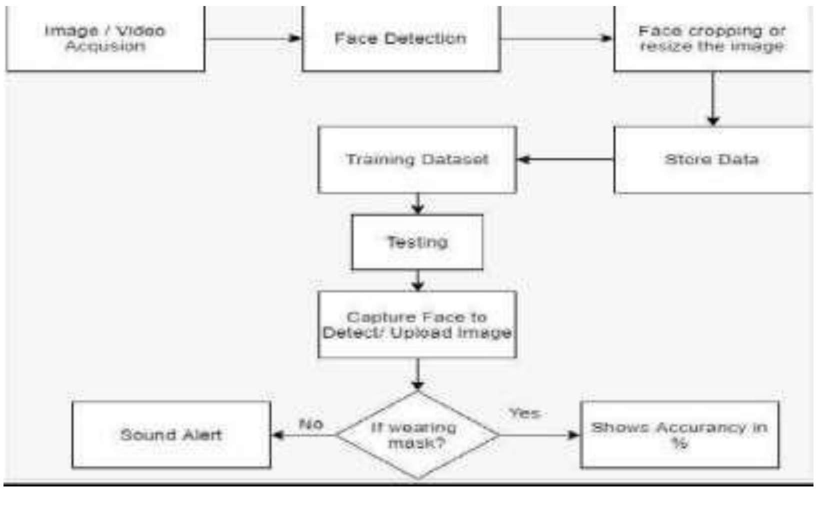
Step 4: load dataset from disk

Step 5: Detect faces from image/video stream

Step6: determine with or without mask

Step 7: output the result

Dataflow diagram:



CAFFEMODEL

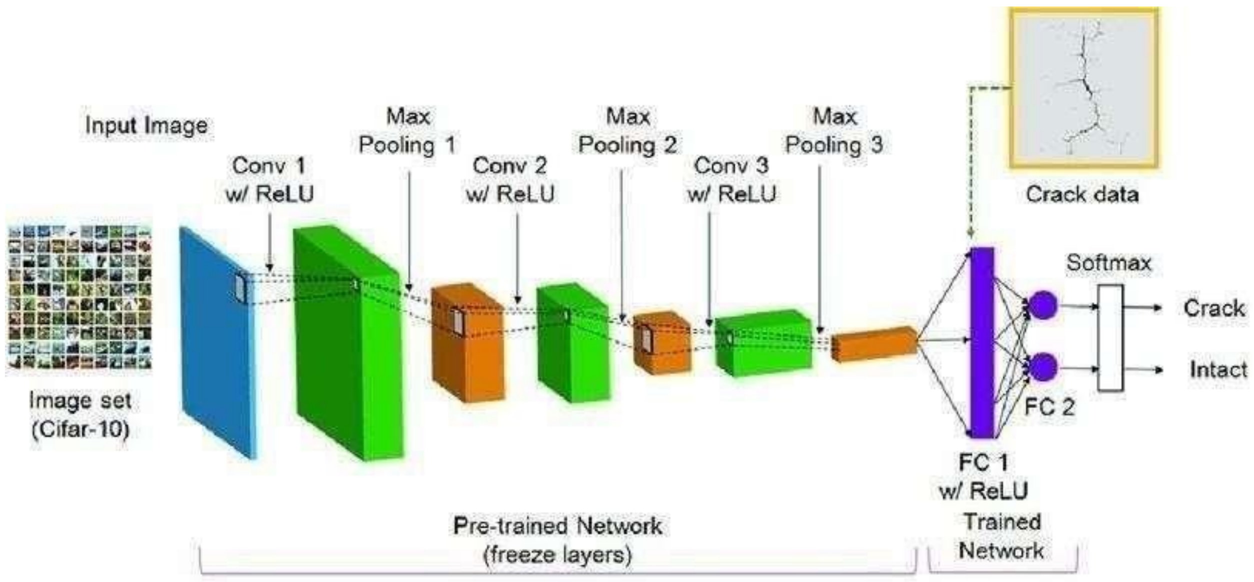
A CAFFEMODEL file is a **machine learning model** created by Caffe. It contains an image classification or image segmentation model that has been trained using Caffe. CAFFEMODEL files are created from .**PROTOTEXT** files.

Caffe features:

Loss functions: Classification Softmax Hinge loss, Linearregression Euclidean loss Attributes/multiclassification Sigmoid cross entropy loss

Available layer types: Convolution Pooling Normalization

Activation functions: ReLU Sigmoid Tanh and more Standard, compact model format caffe train produces a binary .caffemodel file Easily integrate trained models into data pipelines Deploy against new data using command line, Python or MATLAB interfaces Deploy models across HW and OS environments. caffemodel files transfer to any other Caffe installation (including DIGITS).



TESTING AND CODING

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. Is provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product it is the process of exercising ware with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner There are various types of test. Each type addresses a specific testing requirement

TYPES OF TESTS

Unit Testing

that testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid output Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration Testing

Integration tests are designed to test integrated software components to determine if they actually unas cine program Integration testing is specifically aimed at exposing the problems that arise from the combination of components

Functional Test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items

Valid Input	:	identified classes of valid input must be accepted.
Invalid Input	:	identified classes of invalid input must be rejected.
Functions	:	identified functions must be exercised.
Output	:	identified classes of application outputs must be exercised.

Systems Procedures interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows, data fields, predefined processes, and successive processes must be considered for testing.

System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document

Unit Testing

Unit testing is usually conducted as part of a combined code and uns sest phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted two distinct phases

Test Strategy and Approach

Field testing will be performed manually and functional tests will be written in detail

Testing Objectives

- All field entries must work properly.
- Pages must be activated from the idemfled link
- The entry screen, messages and responses must not be delayed.

Features to be Tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed.
- All links should take the user to the correct page

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects

The task of the integration test is to check that components or software applications, eg components in a software system or-one step up-software applications at the company level interact without error

The following are the types of Integration Testing

1) **Top-Down Integration :**

   This method is an incremental approach to the construction of program structure Modules are integrated by moving downward through the control hierarchy, beginning with the main program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner. In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards

2) **Bottom-up Integration :**

   This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

- The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.
- A driver (i.e.) the control program for testing is written to coordinate test case input and output.

- The cluster is tested.
- Drivers are removed and clusters are combined moving upward in the program structure

The bottom up approaches tests each module individually and then each module is modale is integrated with a main module and tested for functionality

Test Results: All the test cases mentioned above passed successfully. No defects encountered

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Test Cases

Test Case Number	**Testing Scenario**	**Expected result**	**Result**
TC – 01	Face detection	Mark a shape around the detected face	Pass
TC – 02	Multiple face detection	Mark a shape around the detected face	Pass
TC – 03	Identify the plain mask	Display the mask found for any color of the mask	Pass
TC -- 04	Identify the pattern mask	Display the mask found for any pattern of the mask	Pass
TC – 05	Identify the mask while wearing sunglasses	Display the mask found for any color of the mask	Pass
TC – 06	Identify the mask of a person having beard	Display the mask found for any color of the mask	Pass
TC -- 07	Detect mask less	Display as not mask if mask is not found	Pass
TC -- 08	Detect facial occlusions	Display as not mask if mask is not found	Pass

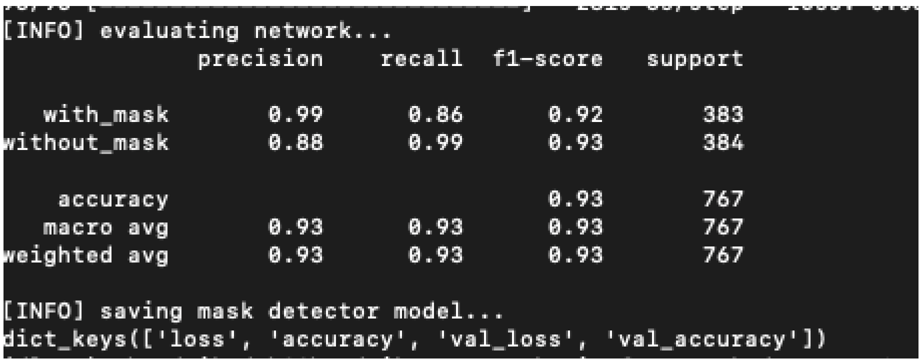
CODING

Trained Module

```
import the necessary packages

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 = 20

BS = 32

DIRECTORY = r"C:\Mask Detection\CODE\Face-Mask-Detection-master\dataset"

CATEGORIES = ["with_mask", "without_mask"]

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

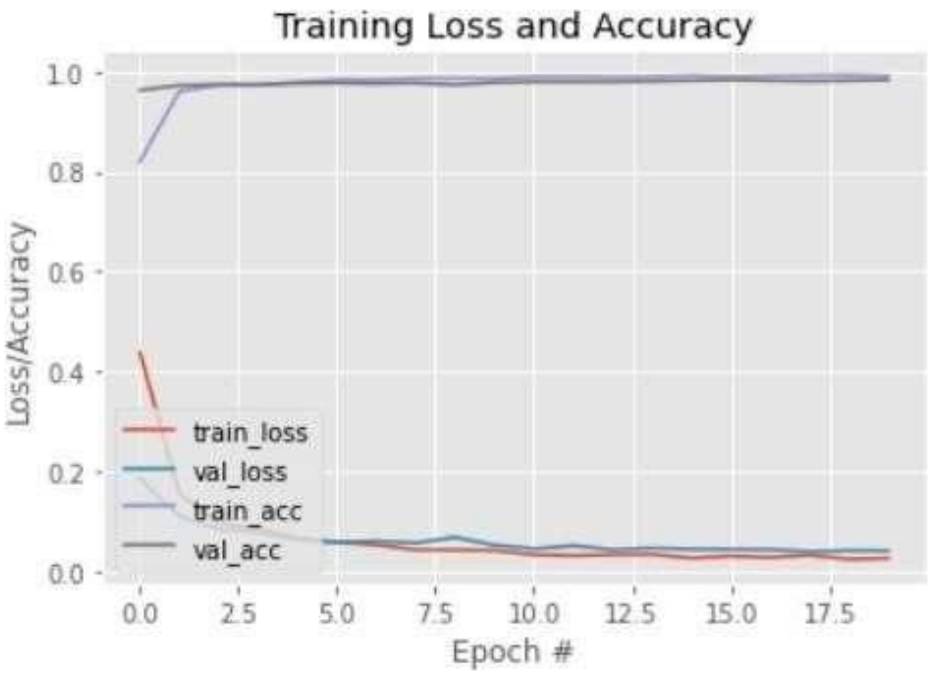
```
  data = [] labels = []
  for category in CATEGORIES:

  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_acc")
  plt.plot(np.arange(0, N), H.history["val_accuracy"], label="val_acc")
  plt.title("Training Loss and Accuracy")

  plt.xlabel("Epoch #")
  plt.ylabel("Loss/Accuracy")
  plt.legend(loc="lower left")
  plt.savefig("plot.png")
```

Our model gave 98% accuracy for Face Mask Detection after training via tensorflow



```
[INFO] evaluating network...
              precision    recall  f1-score   support

   with_mask       0.99      0.86      0.92       383
ithout_mask       0.88      0.99      0.93       384

    accuracy                           0.93       767
   macro avg       0.93      0.93      0.93       767
eighted avg       0.93      0.93      0.93       767

[INFO] saving mask detector model...
dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```

We got the accuracy/ loss training curve plot



Code for Mask Detection Through Live Video Stream

```
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
import numpy as np
import argparse
import cv2
import os
def mask_image():
  ap = argparse.ArgumentParser()
  ap.add_argument("-i", "--image", required=True,
           help="path to input image")
  ap.add_argument("-f", "--face", type=str,
           default="face_detector",
           help="path to face detector model directory")
  ap.add_argument("-m", "--model", type=str,
           default="mask_detector.model",
           help="path to trained face mask detector model")
  ap.add_argument("-c", "--confidence", type=float, default=0.5,
           help="minimum probability to filter weak detections")
  args = vars(ap.parse_args())
  print("[INFO] loading face detector model...")
  prototxtPath = os.path.sep.join([args["face"], "deploy.prototxt"])
  weightsPath = os.path.sep.join([args["face"],
           "res10_300x300_ssd_iter_140000.caffemodel"])

net = cv2.dnn.readNet(prototxtPath, weightsPath)
  print("[INFO] loading face mask detector model...")
  model = load_model(args["model"])
  image = cv2.imread(args["image"]
  orig = image.copy()
```

```
    (h, w) =image.shape[:2]
    blob = cv2.dnn.blobFromImage(image, 1.0, (300, 300),
            (104.0, 177.0, 123.0))
    print("[INFO] computing face detections...")
    net.setInput(blob)
    detections = net.forward()
    for i in range(0, detections.shape[2]):
            confidence = detections[0, 0, i, 2]
            if confidence > args["confidence"]:
                    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 = image[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)
                    face = np.expand_dims(face, axis=0)
                    (mask, withoutMask) = model.predict(face)[0]
                    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(image, label, (startX, startY - 10),
                            cv2.FONT_HERSHEY_SIMPLEX, 0.45, color, 2)
                    cv2.rectangle(image, (startX, startY), (endX, endY), color, 2)

    cv2.imshow("Output", image)
    cv2.waitKey(0)
  if _name_ == "_main_":
    mask_image()
```

RESULTS

Output:

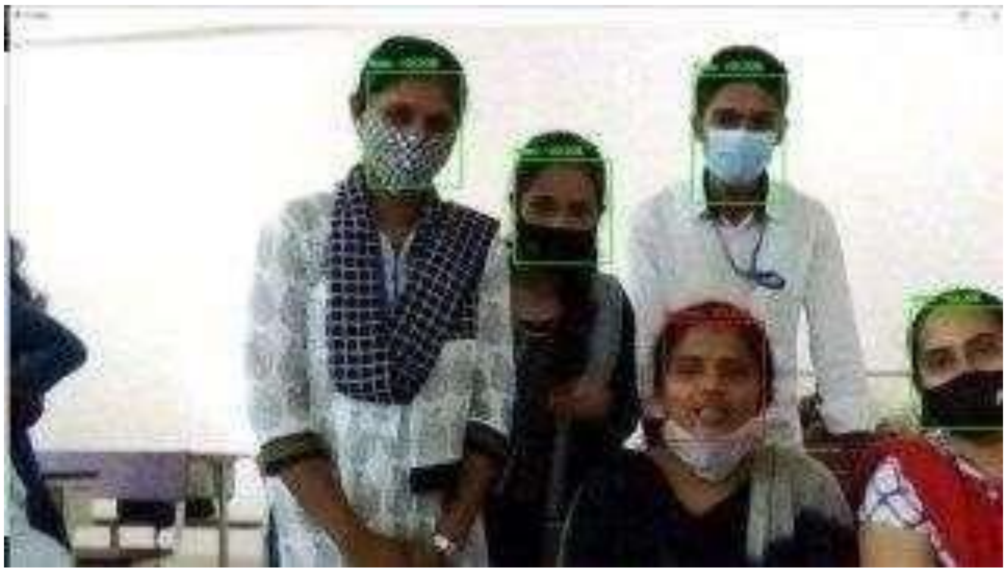


Fig 1. Face mask detected image

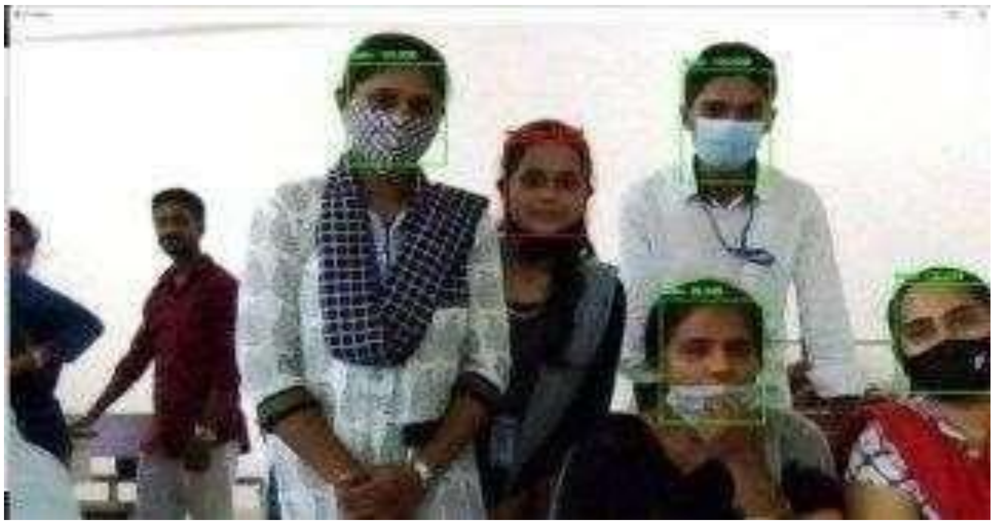


Fig 2. Face mask detected image

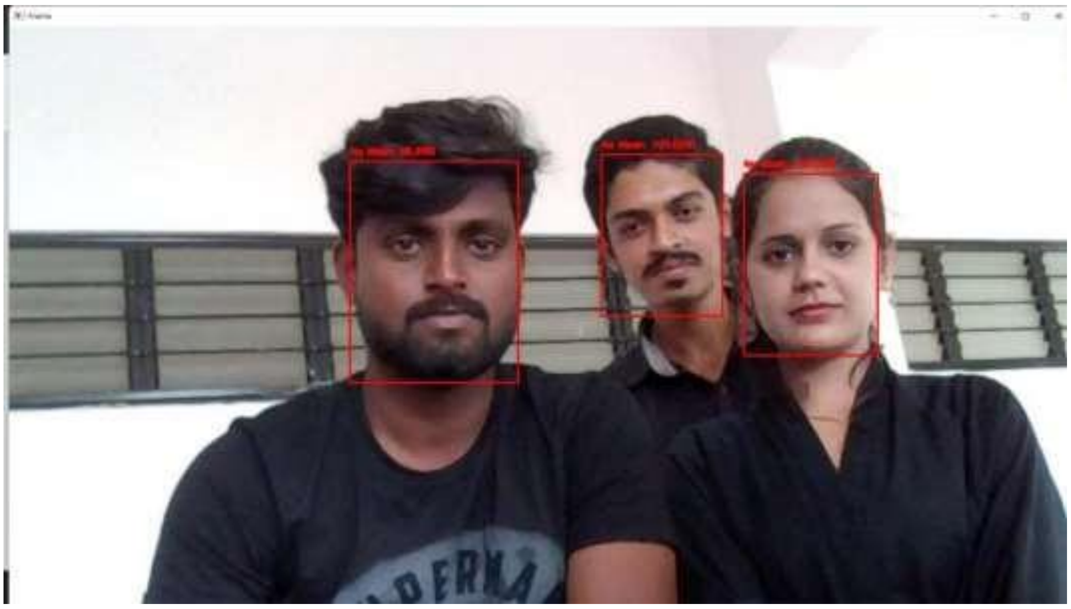


Fig 3. without mask

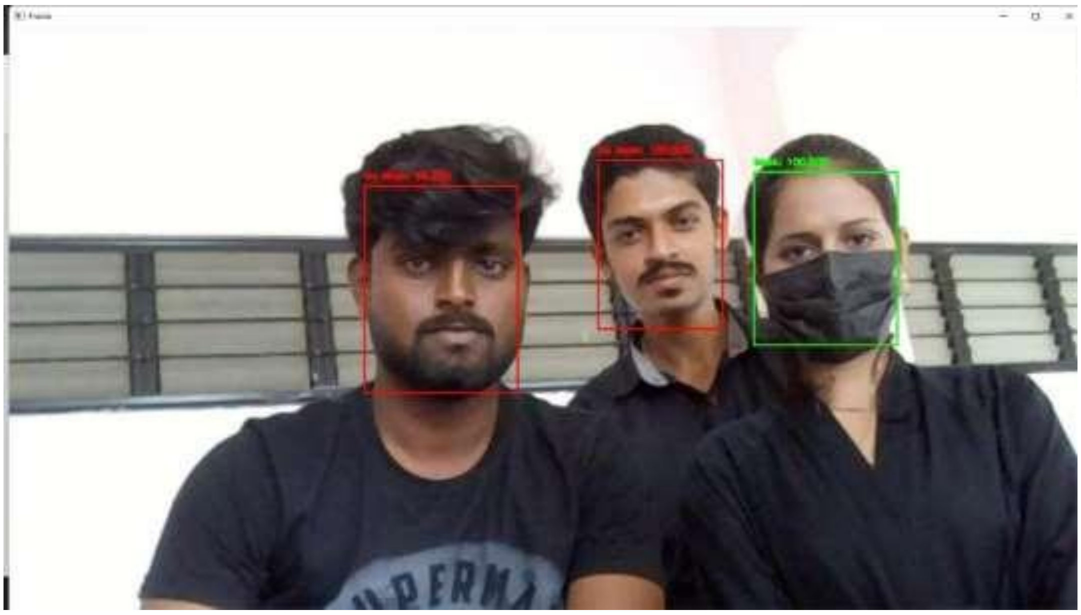


Fig 4. With mask

CONCLUSION

CONCLUSION :

Due to the urgency of controlling COVID-19, the application value and importance of real- time mask and social distancing detection are increasing. This work reviewed, firstly, many research works that seek to surround COVID-19 outbreak. Then, it clarified the basic concepts of deep CNN models. After that, this paper reproduced the training and testing of the most used deep pretrained- based CNN models (DenseNet, InceptionV3, MobileNet, MobileNetV2, ResNet-50, VGG-16, and VGG-19) on the face mask dataset. Finally and after evaluated the numerical results, best models are tested on an embedded vision system consisted of Raspberry Pi board and webcam where efficient real-time deep learning-based techniques are implemented with a social distancing task to automate the process of detecting masked faces and violated or maintained distance between peoples.

REFERENCES

REFERENCES

[1] Face Mask Detection System for COVID_19 Pandemic Precautions using Deep Learning Method, , © 2020 JETIR October 2020, Volume 7, Issue 10, Sneha Sen, Dr. Harish Patidar

[2] DETECTION OF FACE MASKTHROUGH MACHINE LEARNING, Volume:02/Issue:11/November -2020, Kinjal Goswami , A.M Sowjanya

[3] DETECTING OF FACE MASK, Volume:02/Issue:12/December -2020, Karthik Srivathsa D S ,Dr. A. Rengarajan, Nikhil Kumar H

[4] COVID-19 FACEMASK DETECTION WITH DEEP LEARNING AND COMPUTER VISION, Volume: 07 Issue: 08 | Aug 2020 , p-ISSN: 2395-0072, e- ISSN: 2395-0056,Vinitha., Velantina.V2

[5] Covid-19 Face Mask Detection Using TensorFlow, Keras and OpenCV, Arjya Das,Mohammad Wasif Ansari ,Rohini Basak

[6] COVID-19 FACEMASK DETECTION WITH DEEP LEARNING AND COMPUTER VISION, e-ISSN: 2395-0056, p-ISSN: 2395-0072, Volume: 07 Issue: 08 | Aug 2020 , Vinitha, Velantina.

[7] implementation of computer vision frame work for tracking and visualizing face mask in urban environments, Gabriel T S, Peng Sun, PhD, Jerome P.Lynch,PhD

[8] A Deep Learning Based Assistive System to Classify COVID-19 Face Mask for Human Safety with YOLOv3, IEEE – 49239, Md. Rafiuzzaman Bhuiyan ,Sharun Akter Khushbu, Md. Sanzidul Islam

[9] An Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network, Mohammad Marufur Rahman, Saifuddin Mahmud, Jong-Hoon Kim,Md Milon Islam,Md.

[10] Deep Learning Framework to Detect Face Masks from Video Footage, Aniruddha Srinivas Joshi, Shreyas Srinivas Joshi, Goutham Kanahasabai, Rudraksh Kapil, Savyasachi Gupta.

[11] Performance Evaluation of Intelligent Face Mask Detection System with various Deep Learning Classifiers, Vol. 29, No. 11s, (2020), pp. 3074-3082, C.Jagadeeswari, M.Uday Theja.

[12] Ecient Masked Face Recognition Method during the COVID-19 Pandemic, Walid Hariri

[13] Face Mask Detection System using Deep Learning, Published on 05-December- 2020, ISSN: 2455-3778 online, Pinki, Prof. Sachin Garg

[14] Face Mask Detection, Prof. Dr. ChristophLippert, Benjamin Bergner, RazaAli-80514,SaniyaAdeel- 805144,Akhyar Ahmed-804864,Md Shohel Mojumder,Md Hasan Shariar-804693

[15] Real-time face mask identification using Facemasknet deep learning network, Madhura Inamdar · Ninad Mehendaleg

[16] Validating the Correct Wearing of Protection Mask by Taking a Selfie: Design of a Mobile Application

[17] A real time DNN-based face mask detection system using single shot multibox detector and MobileNetV2,Preeti Nagrath, Rachna Jain, and Jude Hemanth

[18] Face Mask Detector using Deep Learning (PyTorch) and Computer Vision (OpenCV),Ramachandran K

[19] Face Mask Recognition using Machine Learning, ISSN: 2349-6002,Tejal Nerpagar, Sakshi Junnare, Janhavi Raut, Aarti Shah, Prof. P.C. Patil

[20] Face detection techniques: a review. Artificial Intelligence Review,52(2),927–948. 10.1007/s10462-018-9650-2, Kumar, A., Kaur, A., Kumar,