DAYANANDA SAGAR COLLEGE OF ENGINEERING

(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)

A Mini-Project Report on

"Face Mask Detection Using Computer Vision"

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

BACHELOR OF ENGINEERING
In
INFORMATION SCIENCE AND ENGINEERING
ACCREDITED BY NBA

Submitted by

Akash S S (1DS18IS008)
Kedar Hegde (1DS18IS035)
Rahul B V (1DS18IS045)
Sandesh S Hegde (1DS18IS057)

Under the Guidance of
Prof. LATHA A P

2020-21

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING
DAYANANDA SAGAR COLLEGE OF ENGINEERING
SHAVIGE MALLESHWARA HILLS, KUMARASWAMY LAYOUT, BANGALORE-78

DAYANANDA SAGAR COLLEGE OF ENGINEERING

Shavige Malleshwara Hills, Kumaraswamy Layout
Bangalore-560078

Department of Information Science and Engineering

ACCREDITED BY NBA & NAAC

2020-21

Certificate

This is to certify that the Project Work entitled - "**FACE MASK DETECTION using Computer Vision Technology**" is a bonafide work carried out by **AKASH S S (1DS18IS008), KEDAR HEGDE (1DS18IS035), RAHUL B V (1DS18IS057), and SANDESH S HEGDE (1DS18IS057)**, in partial fulfillment for the 6th semester of Bachelor of Engineering in Information Science & Engineering of the Visvesvaraya Technological University, Belgaum during the year 2020-2021.

The Mini-Project Report has been approved as it satisfies the academics prescribed for the Bachelor of Engineering degree.

Signature of Guide	Signature of HOD	Signature of Principal
[Prof. Latha A P]	[Dr. K N Rama Mohan Babu]	[Dr. C P S Prakash]

Name of the Examiners: Signature with Date:

1.

2.

ACKNOWLEDGEMENT

It is great pleasure for us to acknowledge the assistance and support of a large number of individuals who have been responsible for the successful completion of this Mini-Project.

We take this opportunity to express our sincere gratitude to **Dayananda Sagar College of Engineering** for having provided us with a great opportunity to pursue our Bachelor's Degree in this institution.

In particular, we would like to thank **Dr. C. P. S Prakash**, Principal, Dayananda Sagar College of Engineering for his constant encouragement and advice.

Special thanks to **Dr. K.N. Rama Mohan Babu**, HOD, Department of Information Science & Engineering, Dayananda Sagar College of Engineering for his motivation and invaluable support well through the development of this Mini-Project.

We are highly indebted to our internal guide **Latha A P, Professor**, Department of Information Science & Engineering, Dayananda Sagar College of Engineering for her constant support and guidance. She has been a great source of inspiration throughout this project.

Finally, we gratefully acknowledge the support of our families during the completion of the project.

Akash S S (1DS18IS008)
Kedar Hegde (1DS18IS035)
Rahul B V (1DS18IS045)
Sandesh S Hegde (1DS18IS057)

ABSTRACT

The new Coronavirus disease (COVID-19) has seriously affected the world. By the end of November 2020, the global number of new coronavirus cases had already exceeded 60 million and the number of deaths 14,10,378 according to information from the World Health Organization (WHO). To limit the spread of the disease, mandatory face-mask rules are now becoming common in public settings around the world. Additionally, many public service providers require customers to wear **face masks** under predefined rules (e.g., covering both mouth and nose) when using public services. These developments inspired research into automatic (computer-vision-based) techniques for face-mask detection to help monitor public behavior and contribute to constraining the COVID-19 pandemic. Although existing research in this area resulted in efficient techniques for face-mask detection, these usually operate under the assumption that modern face detectors provide perfect detection performance (even for masked faces) and that the main goal of the techniques is to detect the presence of face-masks only.

CONTENTS

1. INTRODUCTION

1.1 Overview

Face-mask detection represents both detections as well as a classification problem because it requires first the location of the faces of people in digital images and then the decision of whether they are wearing a mask or not.

Fig. 1: Example images from the MAFA dataset.

All the presented examples are labeled as masked faces, while only the ones marked green have correctly placed masks. In this work, we compile a dataset of masked faces, annotate it manually concerning the mask placement, and then build a computer vision model for the detection of properly worn face-masks.

Fig. 2: People not wearing masks / Masks worn improperly

1.2 Problem Statement

Detect people that pass through a security-like camera and identify their face mask usage.

1.3 Objectives

A project objective describes the desired results of a project, which often includes a tangible item. An objective is specific and measurable and must meet time, budget, and quality constraints**.**

- Check Individuals And Crowds Wearing Masks In Public.
- Use Digital Screens To Remind Visitors To Wear Masks.
- Alert Staff When No Masks Are Detected - Distress Signal deployment.
- Works With Existing USB Or IP Cameras With RTSP Streams.
- Anonymous & Spoof Proof.

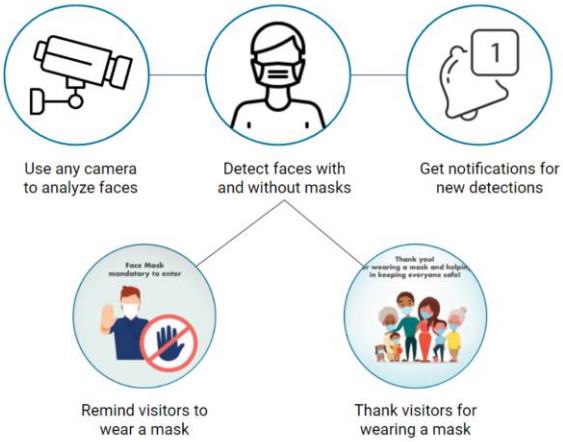

Fig 3: Objectives

1.4 Motivation

Several online sources for face mask detection are currently available on the internet. These systems can detect face masks in videos where people are placed in front of the camera. Since no dataset annotated for face mask detection is currently available, a model cannot be trained to detect faces and do a mask/ no mask classification simultaneously. Thus, most current solutions divide the face mask detection system into two sub-modules; 1- the face detector and 2- the mask/ no mask classifier. The detector detects faces in the video stream, regardless of wearing a mask or not, and then outputs bounding boxes that identify the faces' coordinates. Next, the detected faces are cropped and passed to the classifier. Finally, the classifier decides if the cropped face is wearing a mask or not.

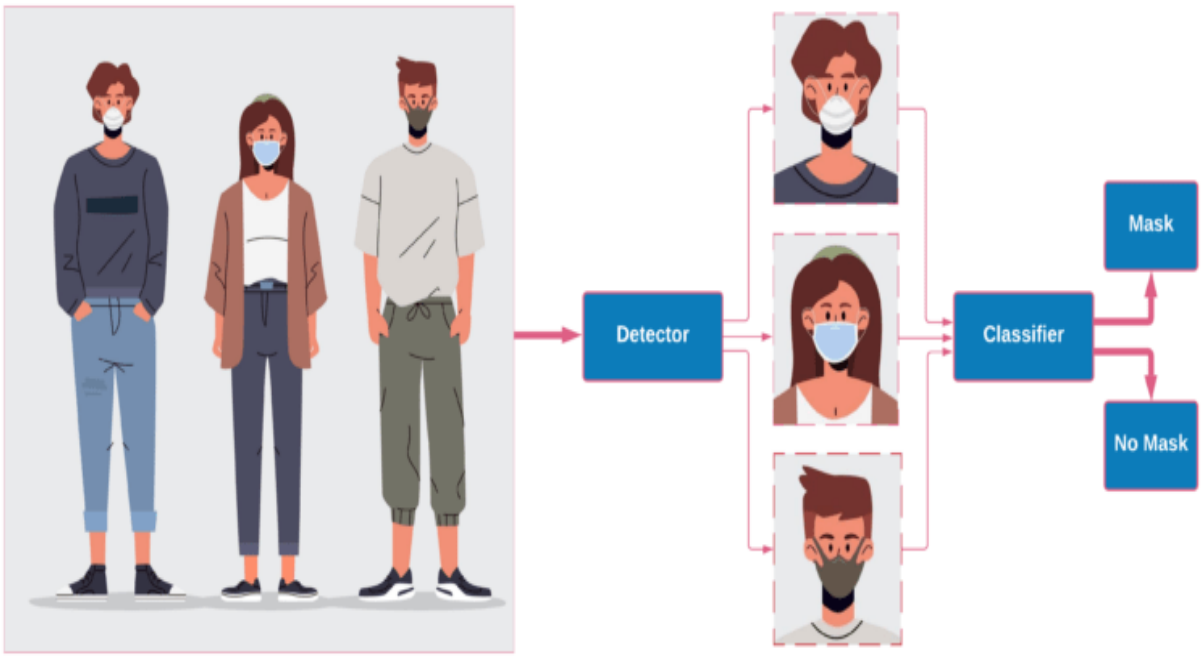

Fig.4: Working Procedure

We were curious about building a face mask detector that can generalize well to real-world data. We tried different datasets with several detectors and classifiers to come up with the right solution. No mask dataset and build a face mask classifier that can work well on real-world data.

2. LITERATURE SURVEY

- **"MAMATA S. KALAS"** proposed a review of various methods and algorithms used for face detection, etc. Three Different algorithms i.e. Haar cascade, AdaBoost, template matching were described Finally it includes some applications of face detection. In this paper, we represent a methodology for face detection robustly in a real-time environment. Here we use a Harr-like classifier and AdaBoost algorithm to track faces on the OpenCV platform which is open source and developed by Intel.

- **"W. ZHAO, et.al"** proposed an up-to-date critical survey of still- and video-based face recognition research. To provide an up-to-date review of the existing literature, and the second is to offer some insights into the studies of machine recognition of faces. To provide a comprehensive survey, we categorize existing recognition techniques and present detailed descriptions of representative methods within each category.

- **"MOHAMMAD MARUFUR RAHMAN, et.al"** proposed a system that restricts the growth of COVID-19 by finding out people who are not wearing any facial masks in a smart city network where all the public places are monitored with Closed-Circuit Television (CCTV) cameras. While a person without a mask is detected, the corresponding authority is informed through the city network. A deep learning architecture is trained on a dataset that consists of images of people with and without masks collected from various sources. The trained architecture achieved 98.7% accuracy on distinguishing people with and without a facial mask for previously unseen test data. It is hoped that our study would be a useful tool to reduce the spread of this communicable disease in many countries in the world.
- **"SAMUEL ADY SANJAYA, et.al"** introduced face mask detection that can be used by the authorities to make mitigation, evaluation, prevention, and action planning against COVID-19. The face mask recognition in this study is developed with a machine learning algorithm through the image classification method: MobileNetV2. The steps for building the model are collecting the data, pre-processing, split the data, testing the model, and implement the model. The built model can detect people who are wearing a face mask and not wearing it at an accuracy of 96,85 percent. After the model was implemented in 25 cities from various sources of the image, the percentage of people wearing a face mask in the cities has a strong correlation to the vigilance index of COVID-19 which is 0.62.

- **"PREETI NAGRATH, et.al"** proposed approach in this paper uses deep learning, TensorFlow, Keras, and OpenCV to detect face masks. This model can be used for safety purposes since it is very resource-efficient to deploy. The SSDMNV2 approach uses Single Shot Multibox Detector as a face detector and MobilenetV2 architecture as a framework for the classifier, which is very lightweight and can even be used in embedded devices (like NVIDIA Jetson Nano, Raspberry pi) to perform real-time mask detection. The technique deployed in this paper gives us an accuracy score of 0.9264 and an F1 score of 0.93. The dataset provided in this papwas collected from various sources can be used by other researchers for further advanced models such as face recognition, facial landmarks, and the facial part detection process.

3. REQUIREMENTS

3.1 Functional Requirements

Functional requirements define the basic system behavior. Essentially, they are what the system does or must not do, and can be thought of in terms of how the system responds to inputs. Functional requirements usually define if/then behaviors and include calculations, data input, and business processes.

- The system must have an unbiased 'with_mask' dataset.
- The dataset must have over 1500+ images in both 'with_mask' and 'without_mask' classes.
- The dataset must not re-use the same images in the training and testing phases.
- There must not be any object between the system and the face of the user for successful face detection and hence the face mask detection.
- The end position of the face must be fit inside the webcam frame and must be closer to the camera.
- Correctly able to detect masks in 'png', 'jpg', 'jpeg', and 'gif' format images. The system must be able to detect face masks on human faces on every frame in a live video.
- The system must be able to detect face masks on human faces on every frame in a live video by showing the probability along with the output of 'Mask' or 'NoMask'.

3.2 Non Functional Requirements

In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. They are contrasted with functional requirements that define specific behavior or functions.

- The face should be localized by detecting the facial landmarks and the background must be ignored.
- The system will be implemented in Python script with an accuracy of the model of over 90%.
- The user must not move his/her face out of the camera's sight to get correct results.
- The background must not be too bright or too dark while detecting the face mask.
- The system must be portable and can be applied to embedded devices with limited computational capacity (ex., Raspberry Pi, Google Coral, NVIDIA Jetson Nano, etc.).
- The output response operation must be fast and under 5 seconds per person.
- The system must be able to correctly detect more than one face if present, and hence the presence of a mask in the frame. The system should be easy for usability and self-descriptive for maintenance purposes.

3.3 Hardware Requirements

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware, A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in the case of operating systems.

- Desktop or Laptop with Windows/Linux/macOS having specs:
  - 4 GB RAM (or higher).
  - Minimum of 2 GB of memory (HDD or SSD).
  - High-speed Internet Connection.
  - Integrated or a separate Webcam.
  - Graphics Card 512 MB (or higher).
- Facemasks for testing.

3.4 SOFTWARE REQUIREMENTS

The software requirements are a description of the features and functionalities of the target system. Requirements convey the expectations of users from the software product. The requirements can be obvious or hidden, known or unknown, expected or unexpected from the client's point of view.

- Python IDE (Spyder or PyCharm).
- pip installer to install modules.
- Modules like TensorFlow, Keras, NumPy, CV2, etc.
- Haar Cascade Classifier and SVM algorithm.

4. SYSTEM ANALYSIS

4.1 Existing System

The dataset has images having masked and unmasked human faces. An existing XML file helps us to detect faces from the image. Imported OpenCV reads an image and it returns an object of NumPy array by default and using img.shape we are checking the height and width of the image and it also returns 3 which is the color channel of the image. Divide the dataset into ‘test’ and ‘train’. With the use of the SVM algorithm, train a model. The captured image is compared with the trained model which in turn gives the required output.

4.2 Proposed System

Face Mask Detection System uses existing IP cameras and CCTV cameras combined with Computer Vision to detect people without masks.
An existing XML file helps us to detect faces from the video.
A continuous video stream in which faces are detected using an XML classifier is compared with the trained model to verify whether the person is masked or not.

5. SYSTEM DESIGN

5.1 Introduction

The framework configuration prepare develops a general structure building outline. The programming diagram incorporates addressing the item system works in a shape that might be changed into at least one anticipates. The essential demonstrated by the end customer must be placed systematically. A diagram is a creative system; an extraordinary design is the best approach to reasonable structure. The structure "Layout" is portrayed as "The methodology of applying distinctive frameworks and guidelines with the ultimate objective of describing a strategy or a system in sufficient purpose important to permit its physical affirmation". Diverse design segments are taken after to add to the system. The design detail depicts the segments of the system, the sections or segments of the structure, and their appearance to end customers.

5.2 Design Consideration

The explanation behind the plan is to orchestrate the course of action of the issue dictated by the necessities report. This stage is the underlying stage in moving from the issue to the game plan space. All things considered, start with what is obliged; the diagram takes us to work towards how to satisfy those necessities. The design of the system is perhaps the most essential segment affecting the way of the item and noteworthily affects the later stages, particularly testing and upkeep. The system diagram delineates all the huge data structure, report game plan, yield, and genuine modules in the system and their Specification is picked.

5.3 Architecture Diagram

The architectural configuration procedure is concerned with building up a fundamental basic system for a framework. It includes recognizing the real parts of the framework and interchanges between these segments. The beginning configuration procedure of recognizing these subsystems and building up a structure for subsystem control and correspondence is called construction

modeling outline and the yield of this outlined procedure is a portrayal of the product structural planning. The proposed architecture for this system is given below.

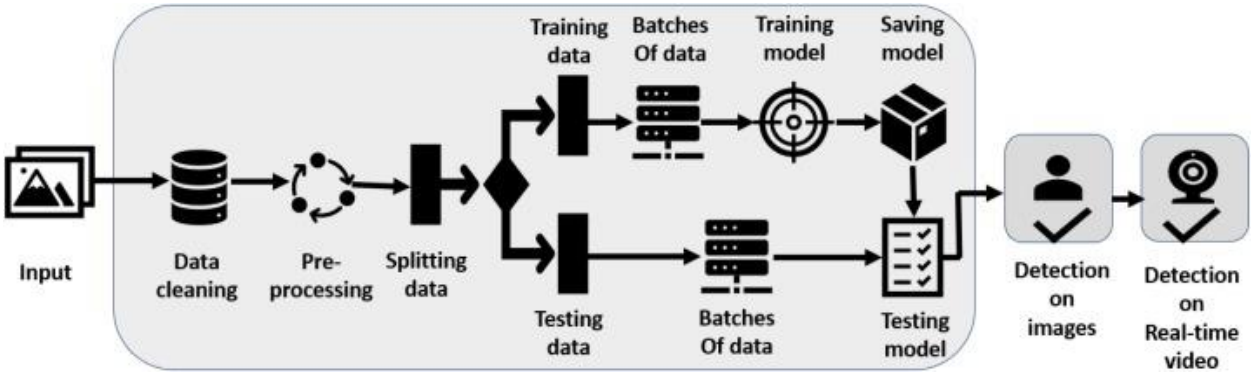

Fig. 5: Architecture Diagram

5.4 Use Case Diagram

A use case chart is a kind of behavioral graph made from a Use-case examination. Its object is to present a graphical diagram of the usefulness gave by a framework regarding performers, their objectives (spoke to as utilization cases), and any conditions between those utilization cases. Use case chart gives us the data about how that clients and utilization cases are connected with the framework. Use cases are used amid prerequisites elicitation and examination to speak to the usefulness of the framework. Use cases concentrate on the conduct of the framework from an outside perspective.

A use case depicts a capacity gave by a framework that yields an obvious result for a performer. A performing artist portrays any element that collaborates with the system. The performers are outside the limit of the framework, while the use cases are inside the limit of the framework. On-screen characters are spoken to with stick figures, use cases with ovals, and the limit of the framework with a container encasing the use cases.

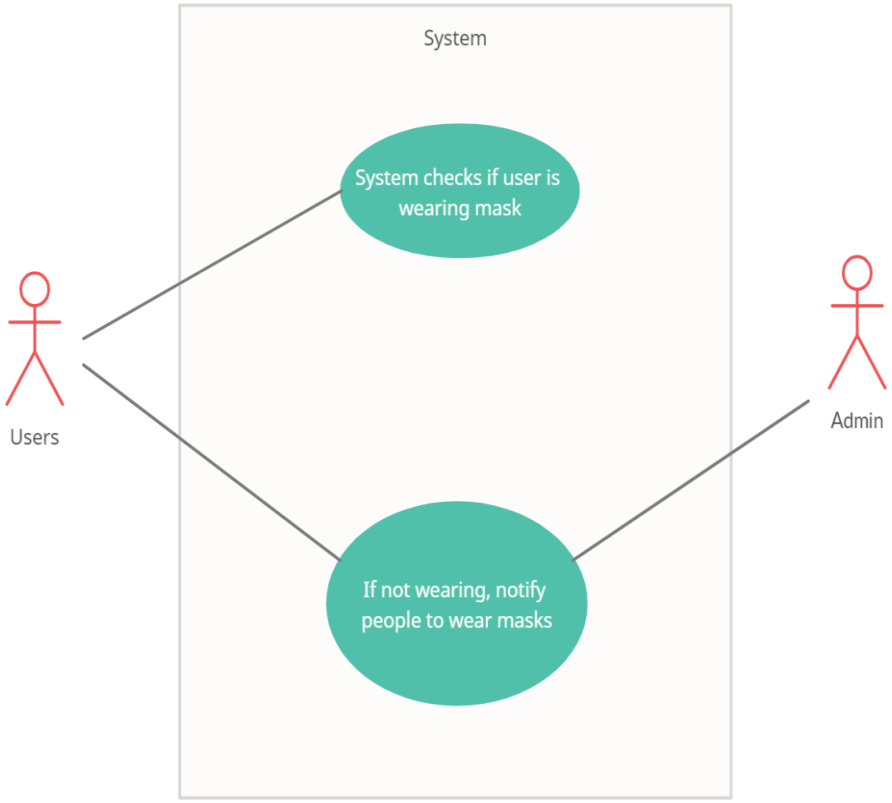

Fig. 6: Use Case Diagram

5.5 Flow Chart Diagram

A flowchart is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task.

The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analyzing, designing, documenting, or managing a process or program in various fields.

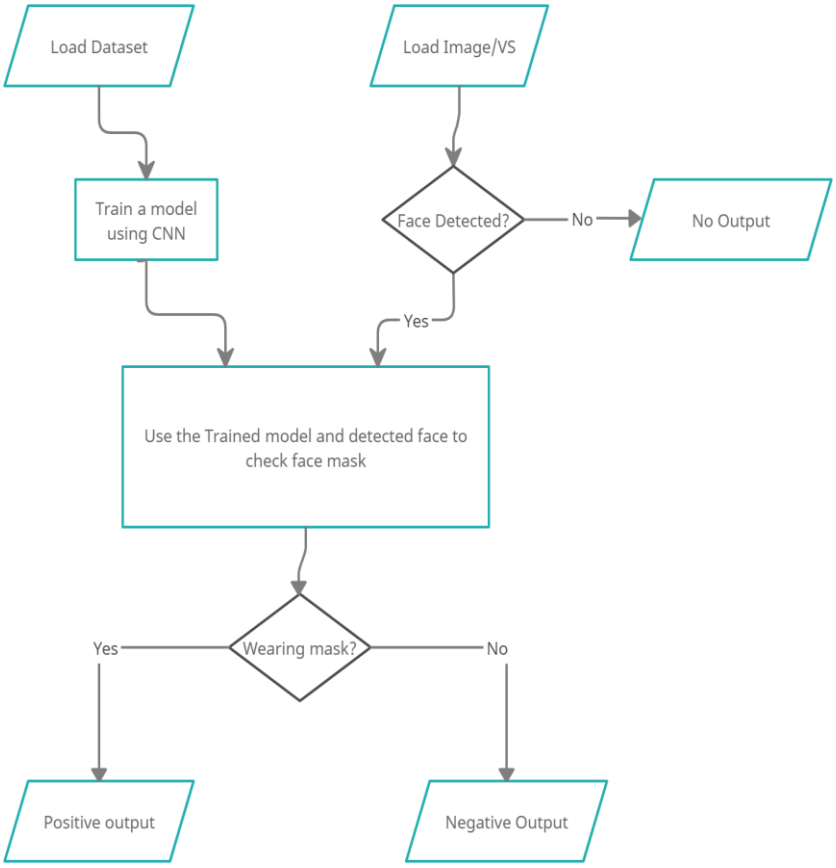

Fig. 7: Flowchart Diagram

5.6 Sequence Diagram

A sequence diagram is a system is an interaction diagram that shows how a process operates with one and another and in what order. It's a construct of a message sequence chart. A sequence diagram shows object interactions arranged in a time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are sometimes called event diagrams or event scenarios.

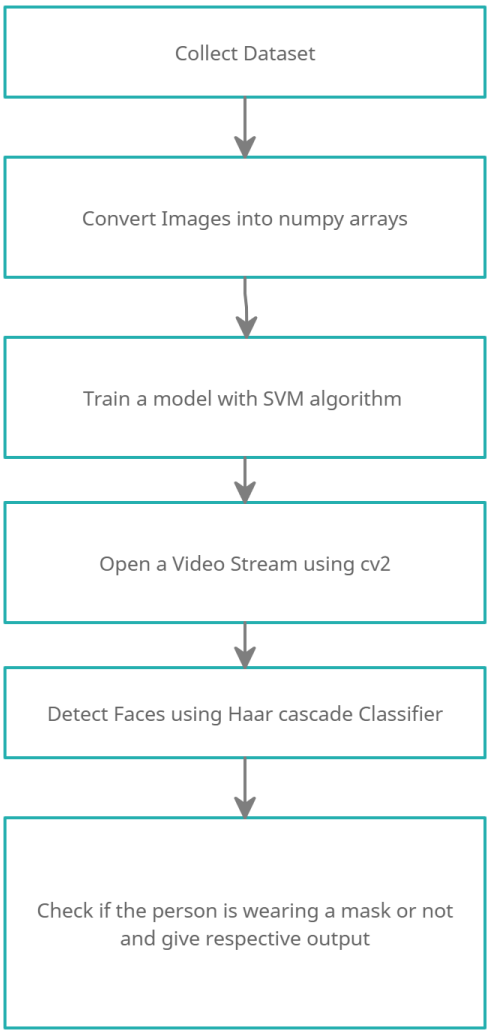

Fig. 8: Sequence Diagram

6. MODULES

6.1 Modules

- CV2
- OS
- NumPy
- TensorFlow
- Keras
- Matplotlib

6.2 Module description

- **OpenCV** - Python is a library of Python bindings designed to solve computer vision problems. OpenCV-Python makes use of Numpy, which is a highly optimized library for numerical operations with a MATLAB-style syntax. All the OpenCV array structures are converted to and from Numpy arrays. This also makes it easier to integrate with other libraries that use Numpy such as SciPy and Matplotlib.

- **OS** - This module provides a portable way of using operating system-dependent functionality. If you just want to read or write a file see open(), if you want to manipulate paths, see the os.path module, and if you want to read all the lines in all the files on the command line see the fileinput module. For creating temporary files and directories see the tempfile module, and for high-level files and directory, handling sees the shutil module.

- **NumPy** - NumPy is a module for Python. The name is an acronym for "Numeric Python" or "Numerical Python". It is an extension module for Python, mostly written in C. This makes sure that the precompiled mathematical and numerical functions and functionalities of Numpy guarantee great execution speed.

- **TensorFlow** - TensorFlow is a Python library for fast numerical computing created and released by Google. It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensorFlow.

- **Keras** - Keras is a minimalist Python library for deep learning that can run on top of Theano or TensorFlow. It was developed to make implementing deep learning models as fast and easy as possible for research and development.

- **Matplotlib** - Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK.

7. PSEUDOCODE

Image dataset to NumPy array

```
for category in categories:
    folder_path=os.path.join(data_path,category)
    img_names=os.listdir(folder_path)

    for img_name in img_names:
        img_path=os.path.join(folder_path,img_name)
        img=cv2.imread(img_path)

        try:
            gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)

            resized=cv2.resize(gray,(img_size,img_size))

            data.append(resized)
            target.append(label_dict[category])

        except Exception as e:
            print('Exception:',e)
```

Training model

```
model=Sequential()

model.add(Conv2D(200,(3,3),input_shape=data.shape[1:]))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
#The first CNN layer followed by Relu and MaxPooling layers

model.add(Conv2D(100,(3,3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
#The second convolution layer followed by Relu and MaxPooling layers

model.add(Flatten())
model.add(Dropout(0.5))
#Flatten layer to stack the output convolutions from second convolution layer
model.add(Dense(50,activation='relu'))
#Dense layer of 64 neurons
model.add(Dense(2,activation='softmax'))
#The Final layer with two outputs for two categories

model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])


from sklearn.model_selection import train_test_split

train_data,test_data,train_target,test_target=train_test_split(data,target,test_size=0.1)


checkpoint = ModelCheckpoint('model-
{epoch:03d}.model',monitor='val_loss',verbose=0,save_best_only=True,mode='auto')
history=model.fit(train_data,train_target,epochs=20,callbacks=[checkpoint],validation_split=0.2)
```

Live stream

```
while(True):

    ret,img=source.read()
    gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    faces=face_clsfr.detectMultiScale(gray,1.3,5)

    for x,y,w,h in faces:

        face_img=gray[y:y+w,x:x+w]
        resized=cv2.resize(face_img,(100,100))
        normalized=resized/255.0
        reshaped=np.reshape(normalized,(1,100,100,1))
        result=model.predict(reshaped)

        label=np.argmax(result,axis=1)[0]

        cv2.rectangle(img,(x,y),(x+w,y+h),color_dict[label],2)
        cv2.rectangle(img,(x,y-40),(x+w,y),color_dict[label],-1)
        cv2.putText(img, labels_dict[label], (x, y-
10),cv2.FONT_HERSHEY_SIMPLEX,0.8,(255,255,255),2)


    cv2.imshow('LIVE',img)
    key=cv2.waitKey(1)

    if(key==27):
        break
```

8. TESTING

Testing of any product comprises giving the product an arrangement of test information and watching if the product carries on not surprisingly. If the product neglects to carry on obviously, then the conditions under which disappointment happens are noted for investigating and amendment. At last, the framework generally tries to guarantee that blunders in past countenances are revealed and the venture acts as determined.

8.1 Basics of software testing

8.1.1 Black Box testing

Black box testing is done to find the following

- Incorrect or missing functions.
- Interface errors.
- Errors on external database access.
- Performance error.
- Initialization and termination error

8.1.2 White Box Testing

This allows the tests to

- Check whether all independent paths within a module have been exercised at least once.
- Exercise all logical decisions on their false sides.
- Execute all loops and their boundaries and within their boundaries.
- Exercise the internal data structure to ensure their validity.
- Ensure whether all possible validity checks and validity lookups have been provided to validate data entry.

8.2 Types of Testing

- Following are the different types of testing.
- Unit Testing.
- Integration Testing.
- System Testing.
- Performance Testing.
- Validation Testing.
- Acceptance Testing.

Let us consider each testing and discuss it in detail. Firstly we move to the first testing and give its detailed description.

8.2.1 Unit Testing

Singular parts are tried to guarantee that they work accurately. Every part is tried freely, without another framework segment. This framework was tried with the arrangement of legitimate test information for every module and the outcomes were checked with the normal yield. Unit testing centers around confirmation exertion on the littlest unit of the product outline module. This is otherwise called MODULE TESTING. This testing is done amid stages, every module is observed to work agreeably concerning the normal yield from the module.

8.2.2 Integration Testing

Mix testing is another part of testing that is for the most part done keeping in mind the end goal to reveal mistakes related to the stream of information crosswise over interfaces. The unit-tried modules are assembled and tried in little sections, which makes it less demanding to seclude and revise mistakes. This approach proceeds with the unit I have coordinated all modules to frame the framework all in all.

8.2.3 System Testing

Framework testing is a progression of various tests whose basic role is to completely practice the PC-based framework. Framework testing guarantees that the whole incorporated programming framework meets prerequisites. It tests a design to guarantee known and unsurprising outcomes. A case of framework testing is the setup arranged framework mix testing. Framework testing depends on process depiction and streams, underscoring pre-driver process and incorporation focuses.

8.2.4 Performance Testing

The execution testing guarantee that the yield is being delivered inside as far as possible and time is taken for the framework aggregating, offering reaction to the clients and demand being sent to the framework to recover the outcomes.

8.2.5 Validation Testing

The approval testing can be characterized from multiple points of view, however, a straightforward definition is that. Approval succeeds when the product capacities in a way that can be sensibly expected by the end client.

NOTE: Manual testing will suffice for testing the project.

9. RESULTS

Accuracy: Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations. One may think that, if we have high accuracy then our model is best. Yes, accuracy is a great measure but only when you have symmetric datasets where values of false positive and false negatives are almost the same. Therefore, you have to look at other parameters to evaluate the performance of your model. For our model, we have got 0.803 which means our model is A

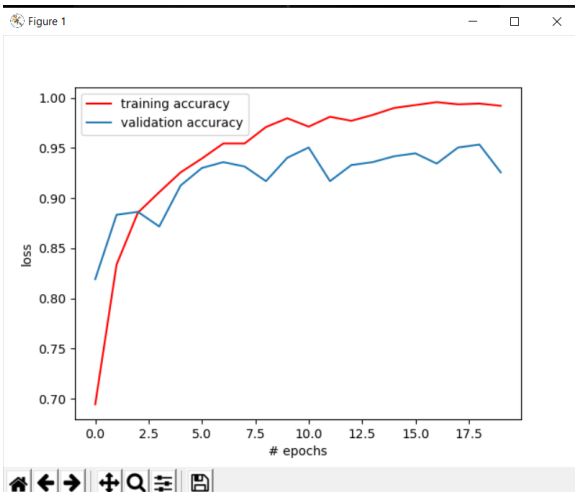

Fig. 9: Accuracy

Loss: Loss is the result of a bad prediction. A loss is a number indicating how bad the model's prediction was on a single example.

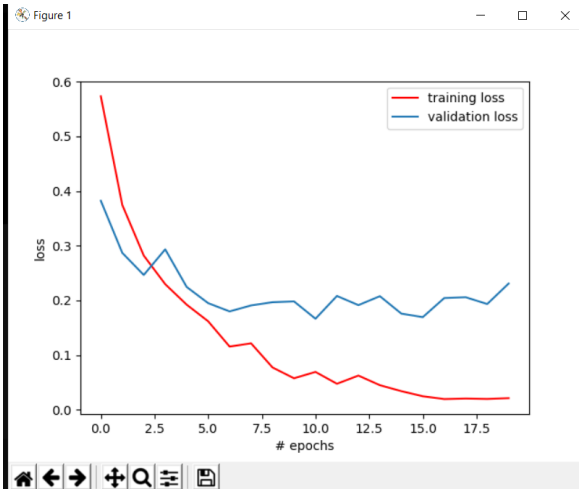

Fig. 10: Loss

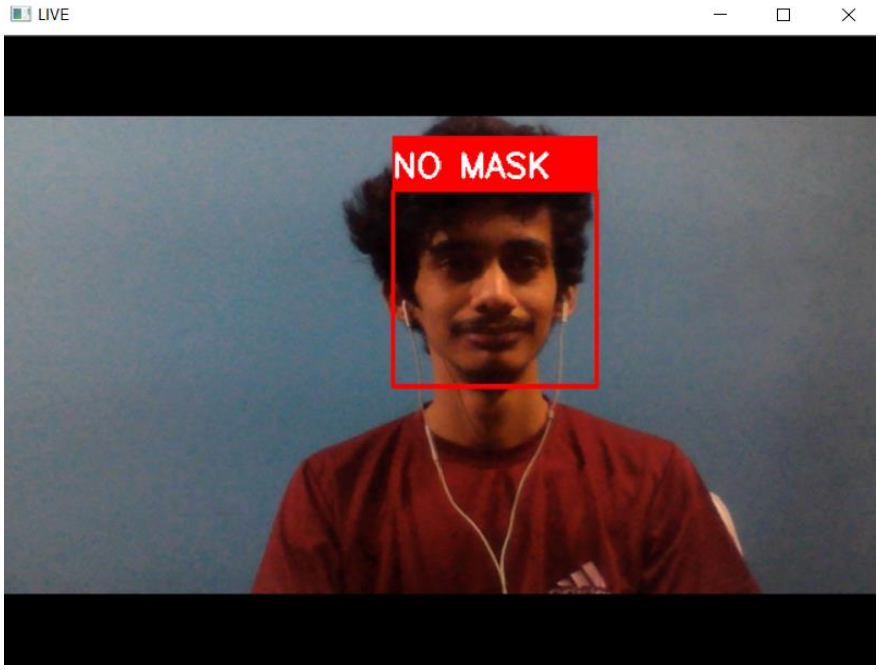

Fig. 11: Without Mask

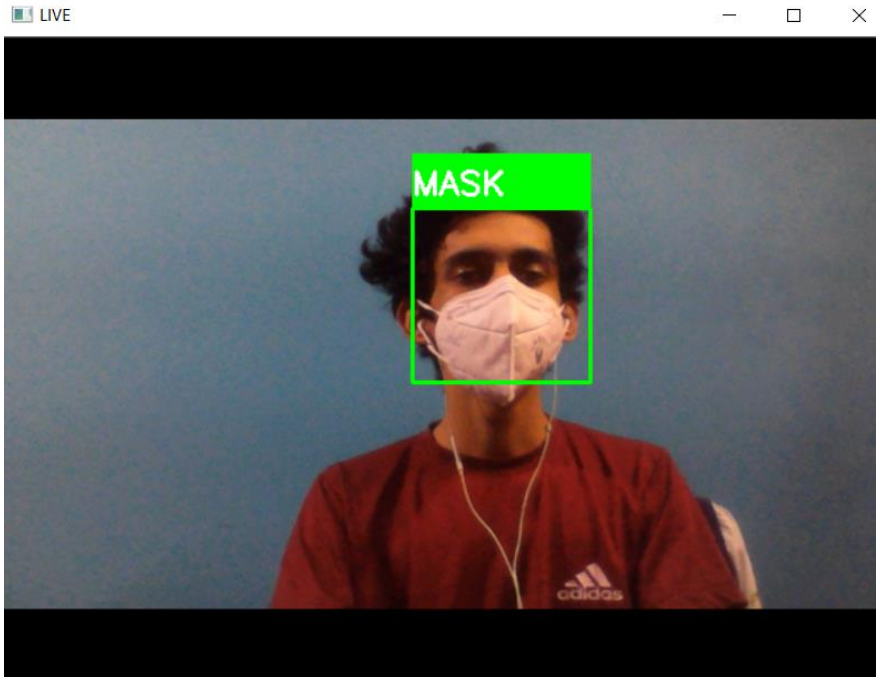

Fig. 12: With Mask

10. CONCLUSION AND FUTURE SCOPE

Since the COVID-19 pandemic has created a lot of chaos, wearing masks and preventing the spread of the virus should be one's top priority. In social gatherings, people usually tend to loosen their masks for their comfort. This increases the spread of the virus.

It requires a lot of manpower to control the crowd and check if they're wearing masks properly or not. Instead, we can use this project to alert the staff so that they can make people wear masks properly or keep a penalty so that people wear them properly. This reduces the spread of the virus which in turn brings an end to this pandemic.

Future Enhancements

- Alert Staff When No Masks Are Detected - Distress Signal deployment.
- Works With Existing USB Or IP Cameras With RTSP Streams.
- Anonymous & Spoof Proof.

11. REFERENCES

1. Real-time Face Mask Detection and Tracking using OpenCV, International Journal of Soft Computing and Artificial Intelligence, ISSN: 2321-404X, Volume-2, Issue-1, May- 2014 by "MAMATA S. KALAS"
2. Face Recognition: A Literature Survey by "W. ZHAO - Sarnoff Corporation, R. CHELLAPPA - University of Maryland, P. J. PHILLIPS - National Institute of Standards and Technology and A. ROSENFELD - University of Maryland", 2015.
3. An Automated System to limit COVID-19 using Facial Mask Detection in the smart city network by "MOHAMMAD MARUFUR RAHMAN, JONG-HOON KIM, SAIFUDDIN MAHMUD, Md. MOTALEB HASSEN MANIK", 2020.
4. Face Mask Detection in the Era of COVID-19 Pandemic by "SAMUEL ADY SANJAYA, SURYO ADI RAKHMAWAN", 2020.
5. SSDMNV2: A real-time DNN-based face mask detection system using a single shot multi-box detector by "PREETI NAGRATH, RACHNA JAIN, AGAM MADAN".
6. World Health Organization. Coronavirus Disease (COVID-19): Advice for the public.
7. TensorFlow Documentation.
8. NumPy Documentation.
9. OpenCV Modules.
10. Scikit-learn.