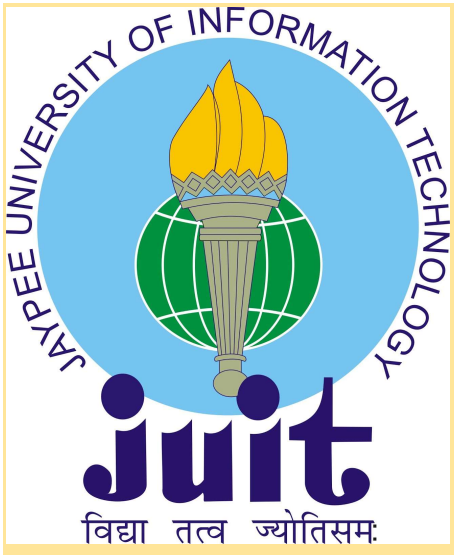
PROJECT REPORT

Drowsiness Detection cum Face Mask Detection System



Submitted to
Dr. Aman Sharma
Dept. of Computer Science and Engineering
JUIT

By
Kartik Ganotra
Adhiraj Gupta

Jaypee University of Information and technology

Certificate

This is to certify that the project titled “Drowsiness Detection cum Face Mask Detection System” has been submitted to the Department of Computer Science and Engineering,Jaypee University Of Technology by following students.

Student Name (With Roll No)

Kartik Ganotra (191254)
Adhiraj Gupta (191233)

Declaration

We, hereby declare that the discussion entitled “Drowsiness Detection cum Face Mask Detection System”
Being submitted by us and have not been submitted by anyone, elsewhere.

DATED: 24th November 2020

Acknowledgement

We like to share our sincere gratitude to all those who helped us in completion of this project. During the project we faced several difficulties due to lack of our knowledge and experience but these people helped us to get over from all the challenges and final compilation of our idea to a shaped sculpture.

We would like to thank Dr Aman Sharma for his governance and guidance, because of which our whole team was able to learn the minute aspects of project work.

In the end we would like to thank Dr Aman Sharma for providing us this opportunity to learn from his experiences.

We are also thankful to our whole class and most of all our parents who have inspired us to face all the challenges and win all the hurdles in life.

Abstract

Driver Drowsiness and Fatigue is one of the major causes of road accidents on expressways and long road trips. On long road trips, especially when driving alone or with people who might be sleeping, the driver may feel drowsiness. Or, on the trips where the driver is already tired or hasn't slept and is restless, the driver might fall asleep while driving and some serious accident or mishap may happen. Not only while driving cars or trucks, one flying a commercial aircraft might also experience fatigue due to over flying, less sleep and jet-lag.
After the outbreak of the global pandemic of COVID-19, masks are a part of everyone's lives. Especially while driving a commercial vehicle like buses, it is essential to wear a mask to reduce the risk of spreading the Corona virus disease. Although, it is a rule to wear a face mask while driving, yet, very few people obey it, or even if they do, they do not wear the mask properly.

This project aims at solving both the problems listed above. Using face detection tools, we identify if a person is feeling drowsy by checking the obvious signs to it, like closing eyes. After some seconds, the driver has yet not opened her/his eyes, the system beeps an alarm to alert the driver.
Face mask detection system in the project also works on face detection. A face mask is detected by camera.
In the initial development of the project,the main focus was only on drowsiness detection systems but later on the face mask detection system was developed and it was merged with the initial project. Using Python and Machine learning we created a very helpful tool.

Content

Drowsiness Detection cum Face Mask Detection System

Jaypee University of Information and technology

Certificate

Declaration

Acknowledgement

Abstract

Introduction

1.1 Purpose of This Project

1.1.1 Driver Drowsiness and Current Statistics

1.1.2 Face Masks and COVID Prevention

1.2 Document Conventions

Literature Survey

2.1 System Review

2.1 Technology Used

2.1.1 Python

2.1.2 Image Processing

2.1.3 Machine Learning

Requirement Analysis

3.1 Software Requirements

3.1.1 Intended Audience

3.1.2 Project Scope

3.1.3 Problem Definition

3.2 Operating System

3.3 Hardware Requirements

3.3.1 Laptop

3.3.2 Webcam

System Design

Development of "Drowsiness cum Face Mask Detection System"

5.1 Scope of Development

5.2 Libraries Used

5.2.1 Keras

5.2.2 Imutils

5.2.3 Dlib
5.2.4 Distance
5.2.5 Scipy
5.2.6 Numpy
5.2.7 face_recognition
5.2.8 cv2

System Training and Testing
6.1 System Training
6.2 System Testing
6.2.1 Face mask detection system
6.2.2 Drowsiness Detection System
6.2.3 Drowsiness cum face mask detection system

Implementation
7.1 Drowsiness Detection System
7.2 Face Mask Detection System

Screenshots of Project
8.1 Drowsy Person without Mask
8.2 Non Drowsy Person with Mask
8.3 Drowsy Person with Mask

Conclusion and Future Scope
9.1 Conclusion
9.2 Future Scope

Bibliography

Chapter 1

Introduction

1.1 Purpose of This Project

1.1.1 Driver Drowsiness and Current Statistics

Driver fatigue and drowsiness is a significant factor in many vehicle and plane crashes and accidents. Many no. of accidents happened in the recent past because of driver or pilot drowsiness or fatigue.

Aviation accidents are still extremely rare, but when they have occurred, figures show that 80% are a result of human error, with pilot fatigue accounting for 15-20% of human error in fatal accidents.

Fatigue leads to slower reaction times and impaired concentration and decision making. There's also the danger of falling asleep.

A survey by the British Airline Pilots' Association (Balpa) of 500 of its members showed 43% had involuntarily fallen asleep in the cockpit, and of those, 31% said that when they woke up the other pilot was also asleep.

In the year 2011, 16 passengers on an Air Canada flight were injured as a result of pilot fatigue. The co-pilot woke disoriented from a nap and, believing that the plane was going to collide with another aircraft, put the jet into a dive, sending passengers sprawling in the cabin.

According to the National Highway Traffic Safety Administration, every year about 100,000 police-reported crashes involve drowsy driving. These crashes result in more than 1,550 fatalities and 71,000 injuries. The real number may be much higher, however, as it is difficult to determine whether a driver was drowsy at the time of a crash.

A study by the AAA Foundation for Traffic Safety estimated that 328,000 drowsy driving crashes occur annually. That's more than three times the police-reported number. The same study found that 109,000 of those drowsy driving crashes resulted in an injury and about 6,400 were fatal. The researchers suggest the prevalence of drowsy driving fatalities is more than 350% greater than reported.

Beyond the human toll is the economic one. NHTSA estimates fatigue-related crashes resulting in injury or death cost society $109 billion annually, not including property damage.

1.1.2 Face Masks and COVID Prevention

Since the outbreak of COVID-19, masks have become an essential part of our lives. Not only while in public places, masks are desirable to be worn in public transport vehicles like buses and traveller vehicles.

Reports from WHO suggest that COVID-19 virus can be contained by wearing masks. Many states have made wearing a mask compulsory and even fines may be imposed on not wearing a mask. To verify the status of a person wearing a mask, a face-mask detection tool is the best alternative for public transport vehicles like buses.
The drowsiness detection cum face mask detection system is a whole package for a public transport vehicle for the prevention of accidents as well as to prevent the spread of COVID-19. This project has a wide scope and has the ability to be used in a bus or an aircraft.

1.2 Document Conventions

Main Heading Font size: 20 (bold fonts)
Sub-headings Font size: 16 (bold fonts)
Normal Content Font size: 14 (normal fonts)
Font Used: Times New Roman

Chapter 2

Literature Survey

2.1 System Review

This survey was done to get a gasp on the need of the general population and the prerequisites which would be required during the project. To do as such, we went through different websites and videos. We reached the decision that there is a need for the developed programs/applications solution and there is a great extent of progress too.

2.1 Technology Used

2.1.1 Python

Python is the best suited language for making this project. With many of its features and pre-built packages, and most importantly code readability, Python was the best choice of programming language to make the project in. Python 3 is the basis of the program that we wrote. It utilizes many of the python libraries which are listed.

2.1.2 Image Processing

Image processing was essential and can be called the backbone of this project. Image capturing and processing are the necessities of this project.

2.1.3 Machine Learning

Machine learning may be termed as a subset of Artificial Intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly told.

Chapter 3

Requirement Analysis

3.1 Software Requirements

3.1.1 Intended Audience

The intended audience for this document is the development team, the project evaluation jury, and other tech-savvy enthusiasts who wish to further work on the project.

3.1.2 Project Scope

This project, “Drowsiness Detection System cum Face Mask Detection System” has an intensive scope in the future. Drowsiness Detection is currently only used in self driving cars and some other expensive cars. Face mask detection systems are currently not being used in any public transportation vehicle in India, or in any other country. This project provides a great scope to be used practically in the future as it offers better results and has a dual work with a single hardware component. As it can detect both, if a person is drowsy while driving or is she/he wearing a face mask or not. This project can be readily installed in self driving cars like Tesla (model S/3/X/Y). It has a scope of usage in public transportation vehicles like buses and small traveller vehicles also.

3.1.3 Problem Definition

Fatigue and drowsiness are two of the most common causes of road accidents, and are such problems, which can not be as easily identified as problems like drinking and driving. A driver drowsiness detection system is much needed to change the driving habits of people and to prevent such accidents.
COVID-19 is one of the most dangerous diseases humanity has ever encountered, a disease which bought the whole world to a halt. To prevent from COVID-19, wearing a mask is very much essential, especially in closed surroundings like cars and aircrafts. A face mask detection system is a great tool to change the habits of drivers to wear a mask while driving a vehicle and also to prevent the spread of the disease.

3.2 Operating System

The Windows 10 OS has been used for developing this application.

3.3 Hardware Requirements

3.3.1 Laptop

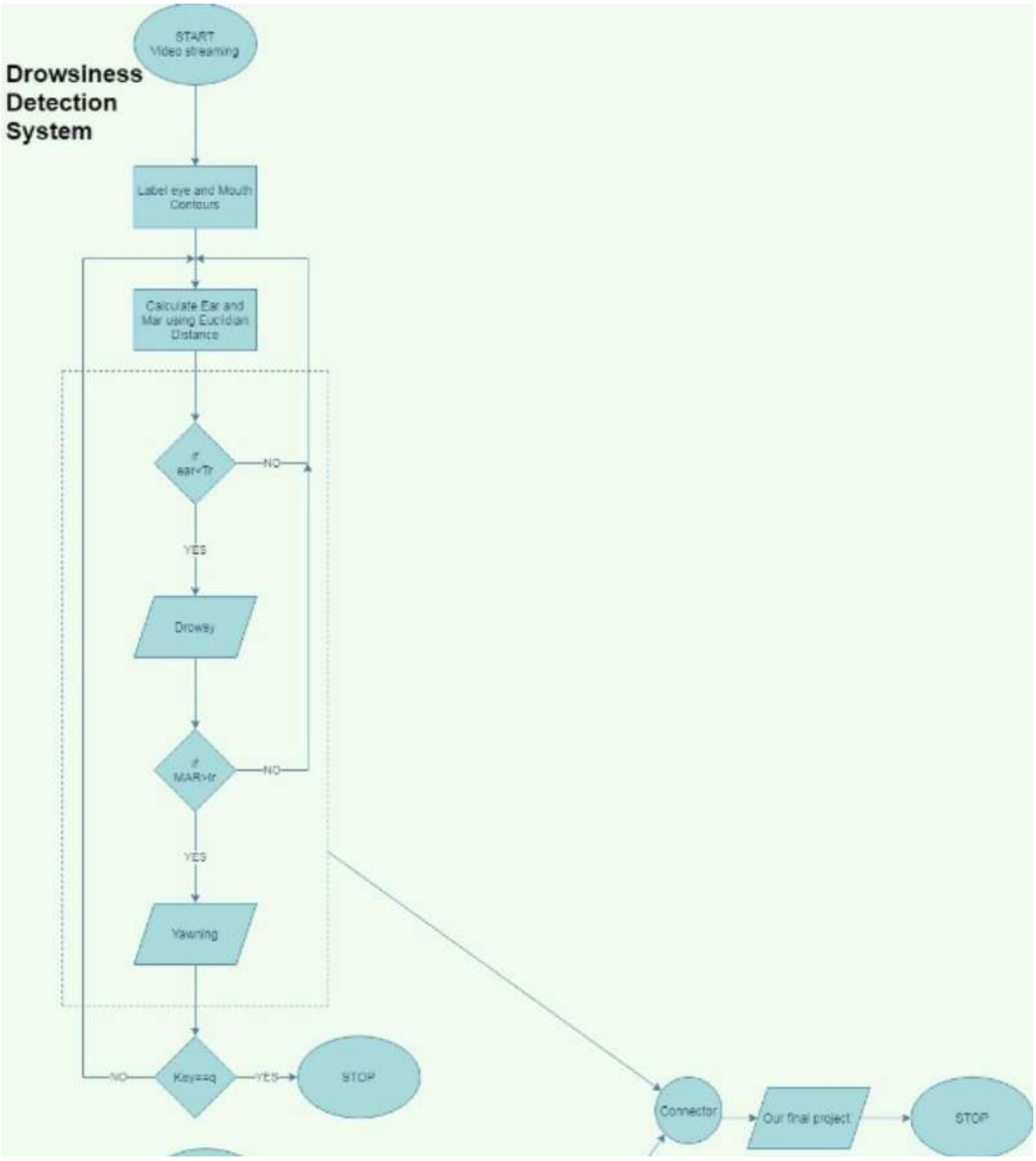
The Laptop as H/w tool used for requirement analysis, understanding, research, development, testing and executing the codes of the program.

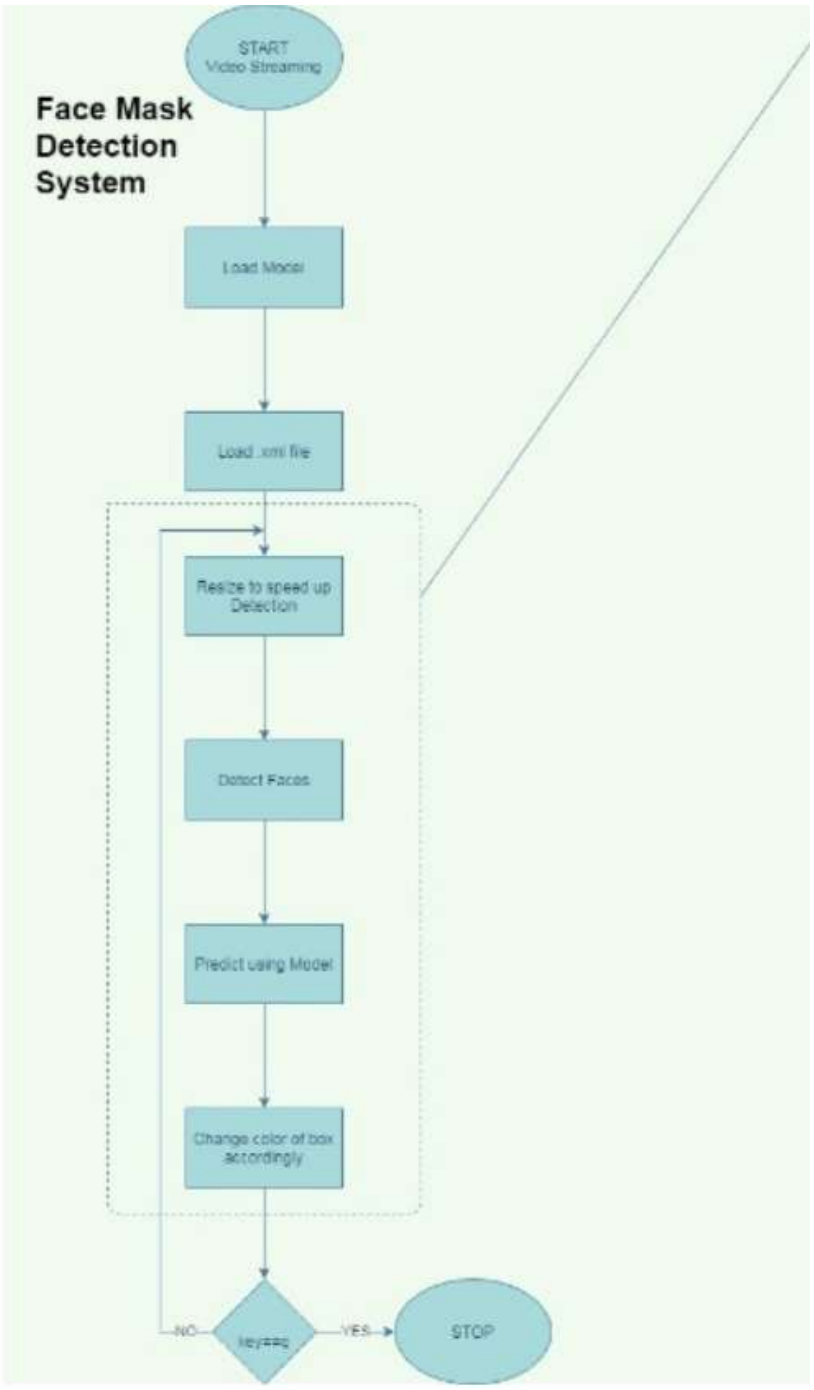
3.3.2 Webcam

An inbuilt Webcam on the Laptop used for getting the video feed of the person.

Chapter 4

System Design




Face Mask Detection System
START Video Streaming
Load Model
Load .xml file
Resize to speed up Detection
Detect Faces
Predict using Model
Change color of box accordingly
NO
YES
STOP

Chapter 5

Development of "Drowsiness cum Face Mask Detection System"

5.1 Scope of Development

This project, "Drowsiness Detection cum Face Mask Detection System" has an intensive scope in the future. Drowsiness Detection is currently only used in self driving cars and some other expensive cars. Face mask detection system currently not being used in any public transportation vehicle in India, or in any other country throughout the world. This project provides a great scope to be used practically in the future as it offers better results and has a dual work with a single hardware component. As it can detect both, if a person is feeling drowsy while driving or is she/he wearing a face mask or not. This project can be readily installed in self driving cars like Tesla (model S/3/X/Y). It has a scope of usage in public transportation vehicles like buses and small traveller vehicles also.

5.2 Libraries Used

The desired solution for the Drowsiness Detection cum Face Mask Detection System has been developed by taking into consideration the Scope of Development (as mentioned in above section) using various Python specific libraries packages and modules, the same are described below.

5.2.1 Keras

Keras is a deep learning API written in Python, running on top of the machine learning platform TensorFlow. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result as fast as possible is key to doing good research.

5.2.2 Imutils

A series of convenience functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, and displaying Matplotlib images easier with OpenCV, OpenCV can be a big, hard to navigate library, especially if you are just getting started learning computer vision and image processing. The find_function method allows you to quickly search function names across modules.

5.2.3 Dlib

Dlib is a modern C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real world problems.

5.2.4 Distance

The package distance provides helpers for computing similarities between arbitrary sequences. Included metrics are Levenshtein, Hamming, Jaccard, and Sorensen, Euclidean (used in our project) distance, plus some bonuses. All distance computations are implemented in pure Python, and most of them are also implemented in C.

5.2.5 Scipy

SciPy is a free and open-source Python library used for scientific computing and technical computing. SciPy contains modules for optimization, linear algebra, integration, interpolation, special functions, image processing, etc. scipy.spatial uses euclidean distance

5.2.6 Numpy

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.arrays have use in ml projects also

5.2.7 face_recognition

The face_recognition recognizes and manipulates faces from python or from the command line with the world's simplest face recognition library. It is built using

dlib‘s state-of-the-art face recognition. built with deep learning. The model has an accuracy of 99.38% on the Labeled Faces in the Wild benchmark. This also provides a simple face_recognition command line tool that lets you do face recognition on a folder of images from the command line.

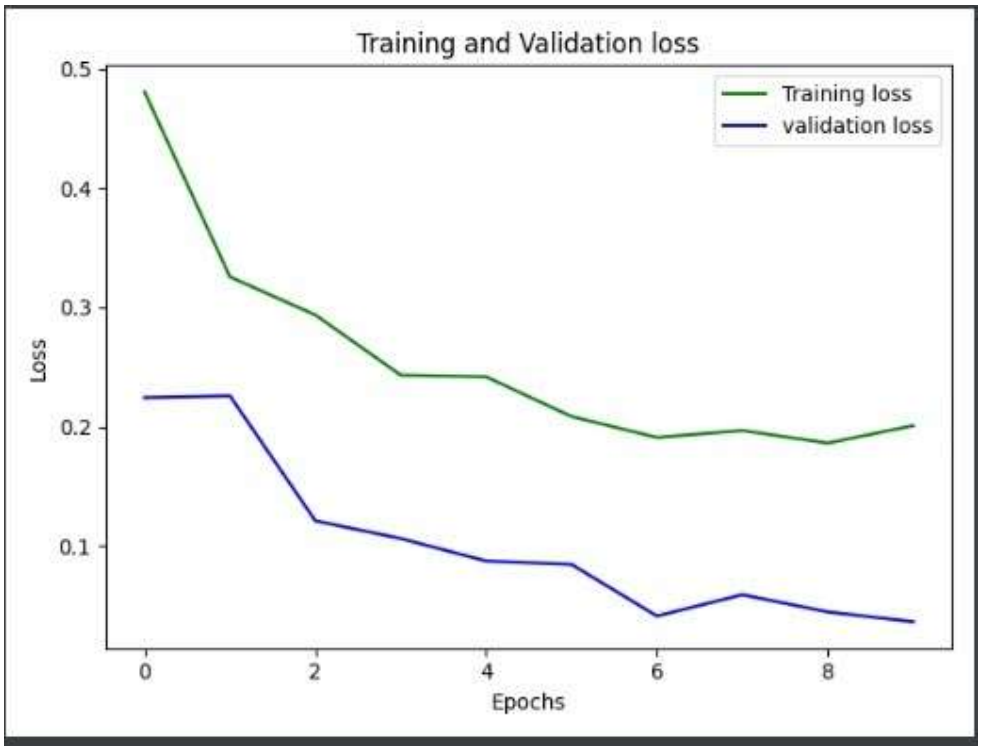
5.2.8 cv2

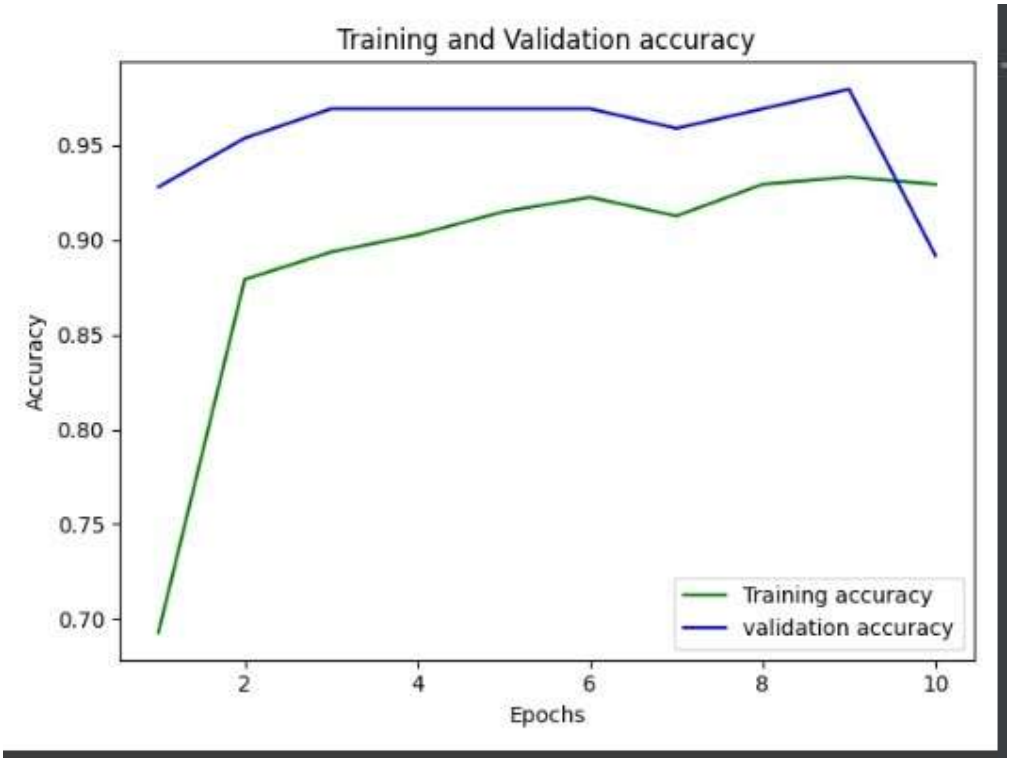
The OpenCV is a cross-platform library using which we can develop real-time computer vision applications. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection.

Chapter 6

System Training and Testing

6.1 System Training






```
from keras.optimizers import RMSprop
from keras.preprocessing.image import ImageDataGenerator
import cv2
from keras.models import Sequential
from keras.layers import Conv2D, Input, ZeroPadding2D, BatchNormalization, Activation, MaxPooling2D, Flatten, Dense, \
Dropout
from keras.models import Model, load_model
from keras.callbacks import TensorBoard, ModelCheckpoint
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score
from sklearn.utils import shuffle
import imutils
import numpy as np
import matplotlib.pyplot as plt
model = Sequential([Conv2D(100, (3, 3), activation='relu', input_shape=(150, 150, 3)),MaxPooling2D(2, 2),Conv2D(100, (3,3), activation='relu'),MaxPooling2D(2, 2),Flatten(),Dropout(0.5),Dense(50, activation='relu'),
    Dense(2, activation='softmax')])
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['acc'])
TRAINING_DIR = r"C:\Users\KeshavG\PycharmProjects\pythonProject3\Dataset\train"
train_datagen = ImageDataGenerator(rescale=1.0 / 255,rotation_range=40,width_shift_range=0.2,height_shift_range=0.2,shear_range=0.2,zoom_range=0.2,horizontal_flip=True, fill_mode='nearest')
train_generator = train_datagen.flow_from_directory(TRAINING_DIR, batch_size=10,target_size=(150, 150))
VALIDATION_DIR = r"C:\Users\KeshavG\PycharmProjects\pythonProject3\Dataset\test"
validation_datagen = ImageDataGenerator(rescale=1.0 / 255)
validation_generator = validation_datagen.flow_from_directory(VALIDATION_DIR_, batch_size=10,target_size=(150, 150))
checkpoint = ModelCheckpoint('model2-{epoch:03d}.model', monitor='val_loss', verbose=0, save_best_only=True,mode='auto')
history = model.fit_generator(train_generator,epochs=10,validation_data=validation_generator,callbacks=[checkpoint])
loss_train = history.history['loss']
loss_val = history.history['val_loss']
epochs = range(0,10)
plt.plot(epochs, loss_train, 'g', label='Training loss')
plt.plot(epochs, loss_val, 'b', label='validation loss')
plt.title('Training and Validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()

acc_train = history.history['acc']
acc_val = history.history['val_acc']
epochs = range(1,11)
plt.plot(epochs, acc_train, 'g', label='Training accuracy')
plt.plot(epochs, acc_val, 'b', label='validation accuracy')
plt.title('Training and Validation accuracy')
```

```
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

6.2 System Testing

6.2.1 Face mask detection system

```
import cv2
import numpy as np
from keras.models import load_model
model=load_model("./model2-006.model")

labels_dict={0:'without mask',1:'mask'}
color_dict={0:(0,0,255),1:(0,255,0)}

size = 4
webcam = cv2.VideoCapture(0) #Use camera 0

# We load the xml file
classifier = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')

while True:
    (rval, im) = webcam.read()
    im=cv2.flip(im,1,1) #Flip to act as a mirror

    # Resize the image to speed up detection
    mini = cv2.resize(im, (im.shape[1] // size, im.shape[0] // size))

    # detect MultiScale / faces
    faces = classifier.detectMultiScale(mini)

    # Draw rectangles around each face
    for f in faces:
        (x, y, w, h) = [v * size for v in f] #Scale the shapesize backup
        #Save just the rectangle faces in SubRecFaces
        face_img = im[y:y+h, x:x+w]
        resized=cv2.resize(face_img,(150,150))
        normalized=resized/255.0
        reshaped=np.reshape(normalized,(1,150,150,3))
        reshaped = np.vstack([reshaped])
        result=model.predict(reshaped)
        #print(result)

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

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

    # Show the image
    cv2.imshow('LIVE',   im)
    key = cv2.waitKey(10)
    # if Esc key is press then break out of the loop
```

```
    if key == 27: #The Esc key
        break
# Stop video
webcam.release()

# Close all started windows
cv2.destroyAllWindows()
```

6.2.2 Drowsiness Detection System

```
import numpy as np
import cv2
import dlib
import face_recognition
import imutils
from imutils import face_utils
import distance
from playsound import playsound
from scipy.spatial import distance
def eye_aspect_ratio(eye):
    a=distance.euclidean(eye[1],eye[5])
    b=distance.euclidean(eye[2],eye[4])
    c=distance.euclidean(eye[0],eye[3])
    EAR = (a+b)/(2*c)
    return EAR
def mouth_aspect_ratio(mouth):
    a=distance.euclidean(mouth[5],mouth[8])
    b=distance.euclidean(mouth[1],mouth[10])
    c=distance.euclidean(mouth[0],mouth[6])
    MAR=(a+b)/(2*c)
    return MAR
cap=cv2.VideoCapture(0)
detector= dlib.get_frontal_face_detector()
predictor=dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")
(left_eye_start , left_eye_end)=face_utils.FACIAL_LANDMARKS_IDXS['left_eye']
(right_eye_start , right_eye_end)=face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
```

```
(mouth_start, mouth_end) = face_utils.FACIAL_LANDMARKS_IDXS["mouth"]
EAR_threshold=0.22
MAR_threshold = 0.8
eye_consec_frames=25
yawn_status=False
yawn_cnt=0
eye_cnt=0
while True:
    ret,frame = cap.read()
    frame = imutils.resize(frame, width=680)
    pic=cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    prev_yawn_status = yawn_status
    var=detector(pic,0)
    for i in var:
        shape=predictor(pic,i)
        shape=face_utils.shape_to_np(shape)
        left_eye=shape[left_eye_start:left_eye_end]
        right_eye = shape[right_eye_start:right_eye_end]
        mouth=shape[mouth_start:mouth_end]
        leftear = eye_aspect_ratio(left_eye)
        rightear=eye_aspect_ratio(right_eye)
        ear=(leftear+rightear)/2
        mouear=mouth_aspect_ratio(mouth)
        left_hull=cv2.convexHull(left_eye)
        right_hull=cv2.convexHull(right_eye)
        mouth_hull=cv2.convexHull(mouth)
        cv2.drawContours(frame,[left_hull],-1,(0,255,255),1)
```

```
        cv2.drawContours(frame,[right_hull],-1,(0,255,255),1)
        cv2.drawContours(frame,[mouth_hull],-1,(0,255,0),1)
        if ear < EAR_threshold:
            #eye_cnt+=1
            cv2.putText(frame, "Eyes blinking / Feeling drowsy", (10, 100), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 328), 2)
            eye_cnt+=1
            if eye_cnt >=eye_consec_frames:
                cv2.putText(frame, "DROWSINESS ALERT!", (10, 50),cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
                playsound("scary.mp3")

        else:
            eye_cnt=0

        cv2.putText(frame, "EAR: {:.2f}".format(ear), (480, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
        if mouear > MAR_threshold:
            cv2.putText(frame, "Yawning ", (10, 70), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
            yawnStatus = True

        else:
            yawn_status=False

        if prev_yawn_status==True and yawn_status==False:
            yawn_cnt+=1
        cv2.putText(frame, "MAR: {:.2f}".format(mouear), (480, 60),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
    cv2.imshow("output",frame)
    button = cv2.waitKey(1) & 0xFF
    if button == ord("q"):
        break



cv2.destroyAllWindows()
cap.release()
```

6.2.3 Drowsiness cum face mask detection system

```
from keras.models import load_model
import numpy as np
import cv2
import dlib
import face_recognition
import imutils
from imutils import face_utils
import distance
from playsound import playsound
from scipy.spatial import distance
def eye_aspect_ratio(eye):
    a=distance.euclidean(eye[1],eye[5])
    b=distance.euclidean(eye[2],eye[4])
    c=distance.euclidean(eye[0],eye[3])
    EAR = (a+b)/(2*c)
    return EAR
def mouth_aspect_ratio(mouth):
    a=distance.euclidean(mouth[5],mouth[8])
    b=distance.euclidean(mouth[1],mouth[10])
    c=distance.euclidean(mouth[0],mouth[6]);
    MAR=(a+b)/(2*c)
    return MAR
cap=cv2.VideoCapture(0)
detector= dlib.get_frontal_face_detector()
predictor=dlib.shape_predictor(r"C:\Users\KeshavG\PycharmProjects\pythonProject2\shape_predictor_68_face_landmarks.dat")
(left_eye_start , left_eye_end)=face_utils.FACIAL_LANDMARKS_IDXS['left_eye']
(right_eye_start , right_eye_end)=face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
(mouth_start , mouth_end) = face_utils.FACIAL_LANDMARKS_IDXS["mouth"]
EAR_threshold=0.22
MAR_threshold = 0.8
eye_consec_frames=25
yawn_status=False
yawn_cnt=0
eye_cnt=0
model = load_model("./model2-006.model")

labels_dict = {0: 'without mask', 1: 'mask'}
color_dict = {0: (0, 0, 255), 1: (0, 255, 0)}
size = 4
classifier = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')

while True:
    ret,frame = cap.read()
    frame = imutils.resize(frame, width=680)
    pic=cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    prev_yawn_status = yawn_status
```

```
var=detector(pic,0)
#frame=cv2.flip(frame,1,1)
mini = cv2.resize(frame, (frame.shape[1] // size, frame.shape[0] // size))
faces = classifier.detectMultiScale(mini)
for f in faces:
    (x, y, w, h) = [v * size for v in f]  # Scale the shapesize backup
    # Save just the rectangle faces in SubRecFaces
    face_img = frame[y:y + h, x:x + w]
    resized = cv2.resize(face_img, (150, 150))
    normalized = resized / 255.0
    reshaped = np.reshape(normalized, (1, 150, 150, 3))
    reshaped = np.vstack([reshaped])
    result = model.predict(reshaped)
    # print(result)

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

    cv2.rectangle(frame, (x, y), (x + w, y + h), color_dict[label], 2)
    cv2.rectangle(frame, (x, y - 40), (x + w, y), color_dict[label], -1)
    cv2.putText(frame, labels_dict[label], (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (255, 255, 255), 2)
for i in var :
    shape=predictor(pic,i)
    shape=face_utils.shape_to_np(shape)
    left_eye=shape[left_eye_start:left_eye_end]
    right_eye = shape[right_eye_start:right_eye_end]
    mouth=shape[mouth_start:mouth_end]
    leftear = eye_aspect_ratio(left_eye)
    rightear=eye_aspect_ratio(right_eye)
    ear=(leftear+rightear)/2
    mouear=mouth_aspect_ratio(mouth)
    left_hull=cv2.convexHull(left_eye)
    right_hull=cv2.convexHull(right_eye)
    mouth_hull=cv2.convexHull(mouth)
    cv2.drawContours(frame,[left_hull],-1,(0,255,255),1)
    cv2.drawContours(frame,[right_hull],-1,(0,255,255),1)
    cv2.drawContours(frame,[mouth_hull],-1,(0,255,0),1)
    if ear < EAR_threshold:
        #eye_cnt+=1
        cv2.putText(frame, "Eyes blinking / Feeling drowsy", (10, 100), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 328), 2)
        eye_cnt+=1
        if eye_cnt >=eye_consec_frames:
            cv2.putText(frame, "DROWSINESS ALERT!", (10, 50),cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
            #playsound("scary.mp3")
```

```
        else:
            eye_cnt=0

        cv2.putText(frame, "EAR: {:.2f}".format(ear), (480, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
        if mouear > MAR_threshold:
            cv2.putText(frame, "Yawning ", (10, 70), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
            yawnStatus = True

        else:
            yawn_status=False

        if prev_yawn_status==True and yawn_status==False:
            yawn_cnt+=1
        cv2.putText(frame, "MAR: {:.2f}".format(mouear), (480, 60),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
    cv2.imshow("output",frame)
    button = cv2.waitKey(1) & 0xFF
    if button == ord("q"):
        break
```

Chapter 7

Implementation

7.1 Drowsiness Detection System

- Start Video Streaming, capture video of the person from webcam;
- Label eye and mouth contours and label coordinates of eye and mouth.
- Calculate ear and Mar using Euclidean Distance
- Set a threshold value for EAR and MAR
- Take a check if ear < threshold and mar > threshold
- Then print that person is drowsy or yawning accordingly
- After some time of this process show the message DROWSINESS ALERT!!!

7.2 Face Mask Detection System

- Start Video Streaming, capture video of the person from webcam;
- Load the Model
- Load the .xml file for face detection
- Resize to speed up detection
- Detect Faces using xml file uploaded above .
- Predict results of the model you trained in comparison with xml file
- Change the color of the box accordingly green for mask and red for not wearing mask .

Chapter 8

Screenshots of Project

8.1 Drowsy Person without Mask

8.2 Non Drowsy Person with Mask

8.3 Drowsy Person with Mask

Chapter 9

Conclusion and Future Scope

9.1 Conclusion

The project completely meets its objectives and requirements. The system can now easily identify and detect whether a person is wearing a mask as well as at the same time can detect whether a person is feeling drowsy. The framework has achieved an unfaltering state where all the bugs have been disposed of.

9.2 Future Scope

The face mask detection model can be improved incrementally by using other parameters such as determining the type of mask, tightness of mask clip(to prevent fogging on glasses/spectacles).
The drowsiness detection system can be improved by considering other factors such as blink rate, state of car, recording the no. of kilometers the car is driven.

Same model can be used in various other platforms like the screens attached on the seats in aeroplanes/trains/buses to determine whether the person watching that is feeling drowsy or not(if the person is detected to be sleeping, the video/screen may automatically be paused and turned off to save power) and to record her/his status of wearing a face mask as well(if the person is not wearing a face mask, she/he may be alerted).

Bibliography

https://github.com/akshaybahadur21/Drowsiness_Detection/blob/master/Drowsiness_Detection.py
https://data-flair.training/blogs/face-mask-detection-with-python/
https://www.pyimagesearch.com/2017/05/08/drowsiness-detection-opencv/