**DRIVER DROWSINESS DETECTION**

**A PROJECT REPORT**

***Submitted by***

**GURU ANAND G (953617104014)**

**KAVIN NARESH G R (953617104017)**

**SURYA M (953617104049)**

***In partial fulfillment for the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

***IN***

**COMPUTER SCIENCE AND ENGINEERING**

**RAMCO INSTITUTE OF TECHNOLOGY, RAJAPALAYAM**

**ANNA UNIVERSITY: CHENNAI 600 025**

**MARCH 2021**

**ANNA UNIVERSITY: CHENNAI 600 025**

**BONAFIDE CERTIFICATE**

Certified that this project report titled **“DRIVER DROWSINESS DETECTION”** is the bonafide work of **GURU ANAND G (953617104014), KAVIN NARESH G R (953617104017), SURYA M (953617104049)** who carried out the project work under my supervision.

|  |  |
| --- | --- |
| **SIGNATURE**  **Dr.K.Vijayalakshmi, M.E, Ph.D. HEAD OF THE DEPARTMENT**  Professor & Head  Department of Computer Science and Engineering,  Ramco Institute of Technology, North Venganallur Village,  Rajapalayam-626117. | **SIGNATURE**  **Ms.I.Getzhi Akila Poornima, M.E., SUPERVISOR**  Assistant professor,  Department of Computer Science and Engineering,  Ramco Institute of Technology, North Venganallur Village,  Rajapalayam-626117. |

Submitted for Anna University Project Viva-Voce Examination held at Ramco Institute of Technology, Rajapalayam on\_\_\_\_\_\_\_\_\_\_\_ .

**ACKNOWLEDGEMENT**

We are greatly indebted to our beloved founder Chairman **Shri.P.R RAMASUBRAHMANEYA RAJHA** and our chairman **Shri.P.R.VENKETRAMA RAJA** for providing excellent facilities to pursue our study and to carry out our project work successfully.

We sincerely thank our Principal **Dr.L.GANESAN**, for his continuous support and advice. We wish to express our sincere gratitude to **Dr.K.VIJAYALAKSHMI**, Professor and Head, Department of Computer Science and Engineering for her support and encouragement for the completion of this project work.

We would like to thank our project coordinators, **Dr.M.Kaliappan** Associate Professor and **Mr.R.Venkatesh** Assistant Professor (Senior Grade), Department of Computer Science and Engineering for their support and suggestions.

We are grateful to our project guide **Ms.I.Getzhi Akila Poornima** Assistant Professor**,** Department of Computer Science and Engineering for her insightful comments, valuable suggestions which helped me to complete this project work successfully.

Our sincere thanks go to our revered faculty members and lab technicians for their help during the course of this project work.

Last but not least, we extend our indebtedness towards our beloved parents for their support which made this project a successful one.

**ABSTRACT**

Computer vision based thoughts have been used for the creation of a Driver Drowsiness Detection System. The little camera has been utilized by a framework that concentrates straight towards the essence of the driver and checks the driver's eyes with a particular ultimate objective to perceive weakness. A notice sign is issued to alert the driver, in such circumstances when exhaustion is perceived. The framework oversees using information picked up for the picture to find the facial tourist spots, which gets the area where the eyes of an individual may exist. On the off chance that the eyes of the driver are discovered close for a specific measure of casings, the proposed framework accepts that the driver is falling asleep and an alarm of caution has been issued. The structure can work just when the eyes are found, and works in encompassing lighting conditions too. A countless number of people drive on the highway day and night. Taxi driver, Bus driver, Truck driver and people travelling suffer from lack of sleep. Due to which it becomes very dangerous when he feels sleepy.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER**  **NO.** | **TOPIC** | **PAGE**  **NO.** |
|  | **ABSTRACT** | I |
|  | **LIST OF FIGURES** | VI |
| **1** | **INTRODUCTION** | **11** |
|  | * 1. Objective of the project | 21 |
|  | 1.2 Scope of the project | 22 |
|  | 1.3 Overview of Machine Learning | 22 |
|  | 1.3.1 Opencv machine learning  Algorithm | 33 |
|  | 1.3.2 Neural networks/Multilayer  Perception(MLP) Algorithms | 44 |

|  |  |  |
| --- | --- | --- |
| **2** | **LITERATURE SURVEY** | **15** |
|  | 2.1 Driver drowsiness detection using  ANN image processing | 15 |
|  | 2.2 Enhanced drowsiness detection using deep  Learning | 15 |
|  | 2.3 Driver behavior detection and classification Using deep convolutional networks | 16 |
|  | 2.4 A Machine learning approach for driver  Drowsiness detection based on eye state | 16 |
|  | 2.5 Eye state analysis using Iris detection to Extract driver’s microsleep periods | 17 |
|  | 2.6 Driver drowsiness detection system using Raspberry pi | 17 |
|  | 2.7 Drowsiness detection system using opencv  2.8 Drowsy driver detection and altering  System | 18  88 |
| 8 | 2.9 Efficient driver fatigue detection and  Alerting system  2.10 IOT based driver drowsiness detection and Health monitoring system  2.11 A Survey on drowsiness detection  Techniques  2.12 Driver fatigue detection based on eye state analysis | 19  9  110  110  1  110 |
| **3** | **PROPOSED WORK** | **111** |
|  | 3.1 Methodology | 111 |
|  | |  |  | | --- | --- | | 3.2 The Model Architecture | 20 | | 3.3 Computational Analysis | 20 | | 3.4 Developing image processing solutions with opencv & DLIB  3.5 Algorithm and Implementation  3.6 Working  3.7 Data gathering  3.8 Feature Engineering | 20 |   3.9 Training  3.10 Testing | 1  115  16  18  20  21  26  27  28  28 |

|  |  |  |
| --- | --- | --- |
| **4** | **EXPERIMENTAL RESULTS AND**  **DISCUSSIONS** | **29** |
|  | 4.1 Outcomes | 30 |
| **5** | **CONCLUSION AND FUTURE WORK** | **35** |
|  | 5.1 Conclusion | 35 |
|  | 5.2 Future work | 35 |

|  |  |  |
| --- | --- | --- |
| **APPENDIX I** | **WORKING ENVIRONMENT** | 36 |
| **APPENDIX II** | **CODING** | 37 |
|  | **REFERENCES** | 41 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE**  **NO** | **FIGURE CAPTION** | **PAGE**  **NO** |
| 4.1 | Installing Anaconda | 28 |
| 4.2 | Installing virtualenv package | 28 |
| 4.3 | Activating virtual environment | 29 |
| 4.4 | Installing python libraries | 30 |
| 4.5 | Execution of drowsiness detection | 30 |
| 4.6 | Detection of active face | 31 |
| 4.7 | Detection of drowsy face | 32 |
| 4.8 | Tensorflow graph generation | 32 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# CHAPTER 1

**INTRODUCTION**

The Driver exhaustion is a noteworthy factor in countless mishaps. Late measurements gauge that yearly 1,200 passings and 76,000 wounds can be credited to fatigue related crashes. Driver drowsiness and fatigue is a major factor which results in numerous vehicle accidents. Developing and maintaining technologies which can efficiently detect or prevent drowsiness at the wheel and alert the driver before a mishap is a major challenge in the field of accident prevention systems. Because of the danger that drowsiness can cause on the roads some methods need to be developed for preventing counteracting its effects. With the advent of modern technology and real time scanning systems using cameras we can prevent major mishaps on the road by alerting car driver who is feeling drowsy through a drowsiness detection system. The point of this undertaking is to build up a prototype drowsiness detection system. The spotlight will be put on planning a framework that will precisely monitor the open or shut condition of the driver's eyes continuously. By monitoring the eyes, it is believed that the symptoms of driver fatigue can be detected early enough to avoid a car accident. Detection of fatigue involves the observation of eye movements and blink patterns in a sequence of images of a face.

* 1. **OBJECTIVE OF THE PROJECT**

The majority of accidents happen due to the drowsiness of the driver. So, to prevent these accidents we will build a system using Python, OpenCV, and Keras which will alert the driver when he feels sleepy. Drowsiness detection is a safety technology that can prevent accidents that are caused by drivers who fell asleep while driving. The objective of this project is to build a drowsiness detection system that will detect that a person’s eyes are closed for a few seconds. This system will alert the driver when drowsiness is detected.

* 1. **SCOPE OF THE PROJECT**

Designing a prototype Drowsiness Detection System which will focus on continuously and accurately monitoring the state of the driver’s eyes in real time to check whether they are open or closed for more than a given period of time. The primary goal is to initially plan a framework to distinguish driver's sluggishness by persistently checking the retina of the eye. The framework works disregarding driver wearing displays and in different lighting conditions. To caution the driver on the identification of laziness by utilizing ringer or alert. Speed of the vehicle can be reduced. Traffic management can be maintained by reducing accidents.

**1.3 OVERVIEW OF MACHINE LEARNING**

The objective of AI is to transform information into data. Subsequent to having gained from a social affair of information, we need a machine that can address any question about the information: What is different information that is like given information? Is there a face in the picture? What sort of advertisement will impact the client? There is generally a cost parameter, hence the question arises: Of the numerous items that we can profit from, which one will probably be purchased by the client if a promotion is appeared for it?AI changes over information into data by identifying standards or examples from that information.

**1.3.1OpenCV’s Machine Learning Algorithms**

The ML calculations that are incorporated into OpenCV are given as pursues. Every one of the calculations are available in the ML library separated from Mahalanobis and K-implies, which are available in CVCORE, and the calculation of face recognition, which is available in CV.

**1.3.2Neural networks / Multilayer perceptron (MLP) algorithm:**

It is a discriminative algorithm which almost always contains hidden units in between the output and the input nodes for better representation of the input signal. It is slow to train; however, it is quite fast to run. It remains the best performer for applications like letter recognition. [4]

**CHAPTER 2**

**LITERATURE SURVEY**

**2.1 Driver drowsiness detection using ANN image processing [1]**

The paper presents a study regarding the possibility to develop a drowsiness detection system for car drivers based on three types of methods: EEG and EOG signal processing and driver image analysis. In previous works the authors have described the research on the first two methods.

In this paper the authors have studied the possibility to detect the drowsy or alert state of the driver based on the images taken during driving and by analyzing the state of the driver's eyes: opened, half-opened and closed.

For this purpose, two kinds of artificial neural networks were employed: a 1 hidden layer network and an auto-encoder network.

Developing these devices, though, experiences several difficulties linked to swift and proper identification of the exhaustion signs of a rider. One of the specialized conceivable outcomes to execute driver tiredness discovery frameworks is to utilize the vision-based approach. Here we are identifying the driver's drowsiness by evaluating the vision framework.

This new system requires no training data at any step or special cameras. The tests performed to evaluate our proposed driver’s drowsiness detection system using real video sequences acquired by a low cost webcam, show that the algorithm provides good results and can work in real-time.

**2.2 Enhanced Drowsiness Detection Using Deep Learning: An fNIRS Study [2]**

In this paper, a deep-learning-based driver-drowsiness detection for brain-computer interface (BCI) using functional near-infrared spectroscopy (fNIRS) is investigated. The passive brain signals from drowsiness were acquired from 13 healthy subjects while driving a car simulator. The brain activities were measured with a continuous-wave fNIRS system, in which the prefrontal and dorsolateral prefrontal cortices were focused. Deep neural networks (DNN) were pursued to classify the drowsy and alert states. For training and testing the models, the convolutional neural networks (CNN) were used on color map images to determine the best suitable channels for brain activity detection in 0~1, 0~3, 0~5, and 0~10 second time windows. The average accuracies (i.e., 82.7, 89.4, 93.7, and 97.2% in the 0~1, 0~3, 0~5, and 0~10 sec time windows, respectively) using DNNs from the right dorsolateral prefrontal cortex were obtained. The CNN architecture resulted in an average accuracy of 99.3%, showing the model to be capable of differentiating the images of drowsy/non-drowsy states. The proposed approach is promising for detecting drowsiness and in accessing the brain location for a passive BCI.

**2.3 Driver behavior detection and classification using deep convolutional neural networks [3]**

Driver behavior monitoring system as Intelligent Transportation Systems (ITS) have been widely exploited to reduce the traffic accidents risk. Most previous methods for monitoring the driver behavior rely on computer vision techniques. Such methods suffer from violation of privacy and the possibility of spoofing.

This paper presents a novel yet efficient deep learning method for analyzing the driver behavior. We have used the driving signals, including acceleration, gravity, throttle, speed, and Revolutions Per Minute (RPM) to recognize five types of driving styles, including normal, aggressive, distracted, drowsy, and drunk driving.

To take the advantages of successful deep neural networks on images, we learn a 2D Convolutional Neural Network (CNN) on images constructed from driving signals based on recurrence plot technique.

Experimental results confirm that the proposed method can efficiently detect the driver behavior.

**2.4 Deep CNN: A Machine Learning Approach for Driver Drowsiness Detection Based on Eye State [4]**

Driver drowsiness is one of the reasons for large number of road accidents these days. With the advancement in Computer Vision technologies, smart/intelligent cameras are developed to identify drowsiness in drivers, thereby alerting drivers which in turn reduce accidents when they are in fatigue. In this work, a new framework is proposed using deep learning to detect driver drowsiness based on Eye state while driving the vehicle. To detect the face and extract the eye region from the face images, Viola-Jones face detection algorithm is used in this work. Stacked deep convolutional neural network is developed to extract features from dynamically identified key frames from camera sequences and used for the learning phase. A SoftMax layer in CNN classifier is used to classify the driver as sleep or non-sleep. This system alerts the driver with an alarm when the driver is in a sleepy mood. The proposed work is evaluated on a collected dataset and shows better accuracy with 96.42% when compared with traditional CNN. The limitation of traditional CNN such as pose accuracy in regression is overcome with the proposed Staked Deep CNN.

**2.5 Eye State Analysis Using Iris Detection to Extract Driver’s Micro Sleep Periods**

Eye state analysis is a critical step for drowsiness detection. In this paper, we propose a robust algorithm for eye state analysis, which we incorporate into a system for driver’s drowsiness detection to extract micro sleep periods. The proposed system begins by face extraction using Support Vector Machine (SVM) face detector then a new approach for eye state analysis based on Circular Hough Transform (CHT) is applied on eyes extracted regions. Finally, we proceed to a drowsy decision.

**2.6 Driver Drowsiness Detection System Using Raspberry Pi**

This proposed system is used for Driver & Road safety system. Based on computer vision techniques, the driver’s face is located from a color video captured in a car. Then, face detection is employed to locate the regions of the driver’s eyes, which are used as the templates for eye tracking in subsequent frames. The tracked eye’s images are used for drowsiness detection in order to generate warning alarms. The proposed approach has three phases: Face, Eye detection and drowsiness detection. The Haar face detection algorithm takes captured frames of image as input and then the detected face as output. It can be concluded this approach is a low cost and effective solution to reduce the number of accidents due to driver's Drowsiness to increase the transportation safety.

**2.7 Prediction Of Drowsy Driver Detection By Using Soft Computing Technique**

This study describes a program based on changes in facial expressions for automatic drowsy drivers and accident prevention. The main reason for road accidents might be due to the number of driving years. Review of face expression will include the driver's somnolence assessment to make the driver cautious. The study therefore outlines the approach to understanding drowsiness in automobiles. We achieve our methodology by taking a face picture of the driver, by searching for the facial features by handling images and using the hybrid technique to analyze the driver's drowsiness level.

**2.8 Drowsiness Detection System Using OpenCV**

More and more professions nowadays require concentration over the long term. Drivers must keep a near eye on the street, so they can respond to sudden occasions quickly. Driver fatigue typically becomes an instantaneous reason behind several traffic accidents. In this manner, there's a ought to create the frameworks that will distinguish and inform a driver of her/his terrible psychophysical condition, which may essentially diminish the number of fatigue-related car mishaps.

**2.9 Drowsy Driver Detection using Representation Learning**

The advancement of computing technology over the years has provided assistance to drivers mainly in the form of intelligent vehicle systems. Driver fatigue is a significant factor in a large number of vehicle accidents.

Thus, driver drowsiness detection has been considered a major potential area so as to prevent a huge number of sleep induced road accidents. This paper proposes a vision based intelligent algorithm to detect driver drowsiness. Previous approaches are generally based on blink rate, eye closure, yawning, eyebrow shape and other hand engineered facial features.

The proposed algorithm makes use of features learnt using convolutional neural networks so as to explicitly capture various latent facial features and the complex non-linear feature interactions.

A softmax layer is used to classify the driver as drowsy or non-drowsy. This system is hence used for warning the driver of drowsiness or in attention to prevent traffic accidents. We present both qualitative and quantitative results to substantiate the claims made in the paper.

**2.10 Efficient Driver Fatigue Detection and Alerting System**

In order to the drowsy driver, this paper contains a new fatigue driving detection algorithm. Experts told that drivers who do not take their usual break, when driving long distance can be feeling sleepy.

Most of the severe road accidents are caused by sleepy drivers rather than drink driving. In order to avoid accidents fatigue detection methods will detect early signs of fatigue in drivers. If driver is falling symptoms of weariness then immediately message is generated that driver is fatigue, then this message will be transferred to the control room in COMMAND navigation system that indicating status of driver.

The role of image processing is to recognize the face of the driver and then extracts the image of the eyes of the driver for detection of drowsiness. The fatigue is detected in the system by an image processing method of comparing the images in video and by using human features we will detect if the driver is fatigued or not.

**2.11 IOT Based Driver Drowsiness Detection and Health Monitoring System.**

This paper introduces driver drowsiness and health parameters causing fatigue. Generally, road accidents caused by fatigue Driver fatigue is a very serious problem causing many thousands of road accidents each year.

It is not possible to calculate the exact number of accidents because of drowsiness but research shows 20% of accidents happens only because of fatigue (rospa). This project provides USB Camera for Eye-Blink Monitoring System and provide buzzer that alerts the driver during the drowsy condition.

Driver’s location can be track using GPS. In the proposed web application design admin will be controlling the parameters of the system and send messages to the colleague. The health monitor of driver is taken care by wearable heart beat sensor, temperature sensor.

Alcohol sensor is provided to detect the alcoholic condition of a rider, and when this condition exists the vehicle speed goes down.

**2.12 A Survey on Drowsiness Detection Techniques**

There are 1.24 million traffic accidents every year with 2.4% caused by drowsy drivers. In that context, several methods for drowsiness detection have been developed. The state of art methods for drowsiness detection was reviewed in this study. Three main approaches were analyzed: behavioral approach, based on the analysis of image and video of drivers; vehicular approach, based on devices inside the vehicle, the majority are sensors embedded on the steering wheel; and the physiological approach, these intrusive methods are devices that a driver has to use on the head, hands, and fingers.

Some authors claim that the physiological methods have the highest accuracy but there is no clear evidence of a comparison between them Moreover, deep learning is used intensively, but SVMs and features vectors are still used. The main disadvantage in this field is related to the dataset.

The researches are not releasing them. Moreover, and more alarming, the few public datasets, are not in real conditions, they are recorded in offices or are simulated, as opposed to having your own training data from a larger sample of participants, while including new distinct signals of drowsiness (sudden head movement, hand movement, or even tracking eye movements, and others). Nevertheless, despite the huge amount of research, the several devices on markets, and car systems; it is not clear which method is the most appropriate, what sensors are the most useful and least intrusive. This study, shows a systematic literature review of the most recently and relevant methods

**2.13 Driver Fatigue Detection based on Eye State Analysis**

In this paper, we present an effective vision-based driver fatigue detection method. Firstly, the inter frame difference approach binding color information is used to detect face. If it exists, the face area is segmented from the image based on a mixed skin tone model.

After analysis, it is appreciated that SVM is the most widely used classifier, as it provides mostly greater precision and speed, but it does not work well for large data sets, in Table 6 we present the most relevant methods. On the other hand, 6 A Survey on Drowsiness Detection Techniques 7 both CNN and HMM are slow in training and expensive. The methods that have the highest percentage of certainty are those that work with deep learning, and most of them use CNN,since it uses special convolution and pooling operations and performs parameter sharing. This enables CNN models to run on any device, making them universally attractive.

These methods are more reliable and accurate but are intrusive for drivers, we present the publications related to physiological methods. The EEG is the most common device, a EEG have 7 bands representing the state of brain, the bands most used are α, β and θ. Moreover, these methods could be divide in FFT-based spectral analysis, wavelet-based spectral analysis and Higher Order Statistics-based analysis.

As other methods, these are difficult to compare because of the lack public database, each research built its own database in a simulated environment

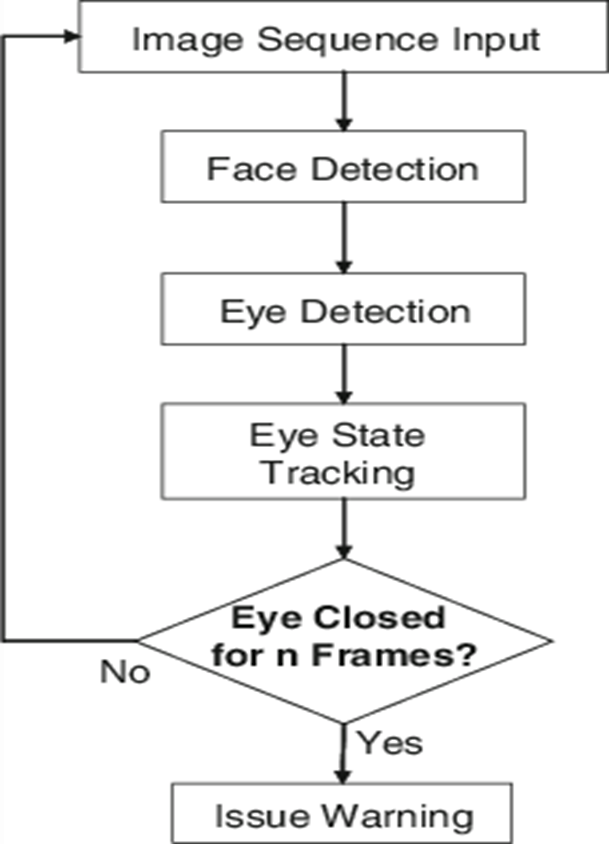
The majority of works carried out tests for the detection of drowsiness with databases in simulated environments, and the lesser number of works that carried out tests in an uncontrolled environment, which required gathering more than one method or combining characteristics to achieve the detection with less margin of error, the work that proved to be most relevant with deep learning.

Then we simulate the process of crystallization to obtain the location of eyes within the face area. Later, eye area, average height of the pupil and width to height ratio are used to analyze the eye’s status. Finally, the driver fatigue is confirmed by analyzing the changes of eye’s states. The experimental results show validity of our proposed method.

**CHAPTER 3**

**PROPOSED WORK**

* 1. **METHODOLOGY**



The flowchart of the proposed system has been shown in the above figure. The camera captures the image and sends it to the processor of the laptop which consists of a 32-bit memory card installed with OpenCV which helps in image processing. If the signal crosses the threshold of a set of continuous frames with EAR less than threshold value, it will automatically make the alarm beep and the speed of the vehicle gets reduced. Otherwise that signal is rejected and the next signal is processed.

In this Python project, we will be using OpenCV for gathering the images from webcam and feed them into a which will classify whether the person’s eyes are ‘Open’ or ‘Closed’. The approach we will be using for this Python project is as follows:

Step 1 – Take image as input from a camera.

Step 2 – Detect the face in the image and create a Region of Interest (ROI).

Step 3 – Detect the eyes from ROI and feed it to the classifier.

Step 4 – Classifier will categorize whether eyes are open or closed.

Step 5 – Calculate score to check whether the person is drowsy.

**Step 1 – Take Image as Input from a Camera**

With a webcam, we will take images as input. So to access the webcam, we made an infinite loop that will capture each frame.

We use the method provided by OpenCV, cv2.VideoCapture(0) to access the camera and set the capture object (cap). cap.read() will read each frame and we store the image in a frame variable.

**Step 2 – Detect Face in the Image and Create a Region of Interest (ROI)**

To detect the face in the image, we need to first convert the image into grayscale as the OpenCV algorithm for object detection takes gray images in the input. We don’t need color information to detect the objects. We will be using a haar cascade classifier to detect faces. This line is used to set our classifier face = cv2.CascadeClassifier(‘path to our haar cascade xml file’).

Then we perform the detection using faces = face.detectMultiScale(gray). It returns an array of detections with x,y coordinates, and height, the width of the boundary box of the object. Now we can iterate over the faces and draw boundary boxes for each face.

for (x,y,w,h) in faces:

cv2.rectangle(frame, (x,y), (x+w, y+h), (100,100,100), 1 )

**Step 3 – Detect the eyes from ROI and feed it to the classifier**

The same procedure to detect faces is used to detect eyes. First, we set the cascade classifier for eyes in Left-eye and Right-eye respectively then detect the eyes using left\_eye = leye.detectMultiScale(gray). Now we need to extract only the eyes data from the full image. This can be achieved by extracting the boundary box of the eye and then we can pull out the eye image from the frame with this code.

l\_eye = frame [ y : y+h, x : x+w ]

l\_eye only contains the image data of the eye. This will be fed into our CNN classifier which will predict if eyes are open or closed. Similarly, we will be extracting the right eye into r\_eye.

**Step 4 – Classifier will Categorize whether Eyes Are Open or Closed**

We are using [CNN](https://en.wikipedia.org/wiki/Convolutional_neural_network) classifier for predicting the eye status. To feed our image into the model, we need to perform certain operations because the model needs the correct dimensions to start with. First, we convert the color image into grayscale using r\_eye = cv2.cvtColor(r\_eye, cv2.COLOR\_BGR2GRAY).

Then, we resize the image to 24\*24 pixels as our model was trained on 24\*24 pixel images cv2.resize(r\_eye, (24,24)). We normalize our data for better convergence r\_eye = r\_eye/255 (All values will be between 0-1). Expand the dimensions to feed into our classifier. We loaded our model using model = load\_model(‘models/cnnCat2.h5’). Now we predict each eye with our model  
lpred = model.predict\_classes(l\_eye). If the value of lpred[0] = 1, it states that eyes are open, if value of lpred[0] = 0 then, it states that eyes are closed.

**Step 5 – Calculate Score to Check whether Person is Drowsy**

The score is basically a value we will use to determine how long the person has closed his eyes. So if both eyes are closed, we will keep on increasing the score and when eyes are open, we decrease the score. We are drawing the result on the screen using cv2.putText() function which will display real time status of the person.cv2.putText(frame, “Open”, (10, height-20), font, 1, (255,255,255), 1, cv2.LINE\_AA) A threshold is defined for example if score becomes greater than 15 that means the person’s eyes are closed for a long period of time. This is when we beep the alarm using sound.play.

**3.2 The Model Architecture**

The model we used is built with Keras using Convolutional Neural Networks (CNN). A convolutional neural network is a special type of deep neural network which performs extremely well for image classification purposes. A CNN basically consists of an input layer, an output layer and a hidden layer which can have multiple numbers of layers. A convolution operation is performed on these layers using a filter that performs 2D matrix multiplication on the layer and filter.

The CNN model architecture consists of the following layers:

Convolutional layer; 32 nodes, kernel size 3

Convolutional layer; 32 nodes, kernel size 3

Convolutional layer; 64 nodes, kernel size 3

Fully connected layer; 128 nodes

The final layer is also a fully connected layer with 2 nodes. In all the layers, a Relu activation function is used except the output layer in which we used Softmax.

**Prerequisites**

The requirement for this Python project is a webcam through which we will capture images. You need to have Python (3.6 version recommended) installed on your system, then using pip, you can install the necessary packages.

OpenCV – pip install opencv-python (face and eye detection).

TensorFlow – pip install tensorflow (keras uses TensorFlow as backend).

Keras – pip install keras (to build our classification model).

Pygame – pip install pygame (to play alarm sound).

The “haar cascade files” folder consists of the xml files that are needed to detect objects from the image. In our case, we are detecting the face and eyes of the person.The models folder contains our model file “cnnCat2.h5” which was trained on convolutional neural networks.We have an audio clip “alarm.wav” which is played when the person is feeling drowsy. “Model.py” file contains the program through which we built our classification model by training on our dataset. You could see the implementation of convolutional neural networks in this file.

“Drowsiness detection.py” is the main file of our project. To start the detection procedure, we have to run this file.Let’s now understand how our algorithm works step by step.

**3.3 COMPUTATIONAL ANALYSIS**

**MATCHING**

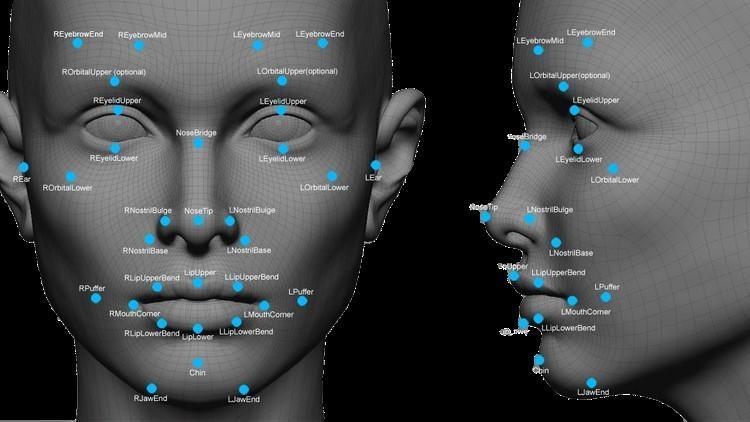
Dashboard mounted cameras are used to monitor the eyes of the driver in real time to detect drowsiness.

**DROWSINESS DETECTION DESIGN**

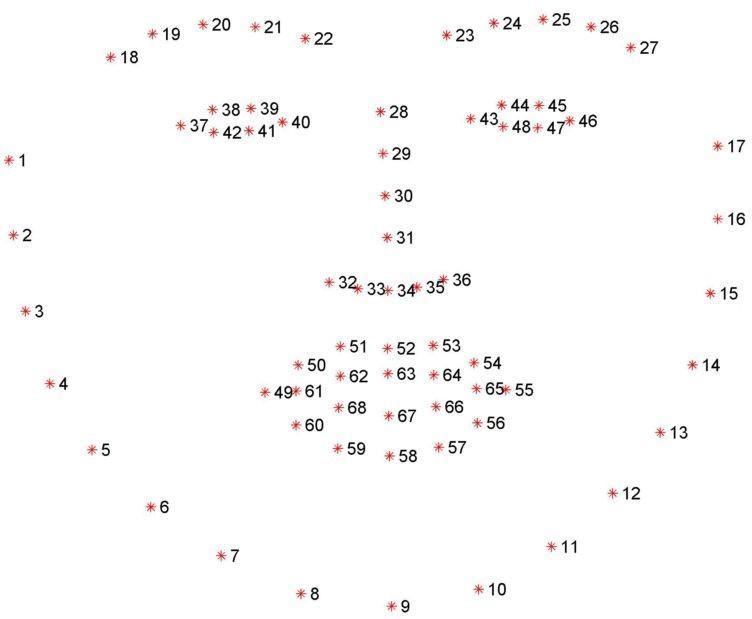
A camera is set up that looks for faces in the input video stream and monitors frames of faces. In the event that a face is identified, facial milestone identification is connected and the eye district is removed from the edges of the video stream.

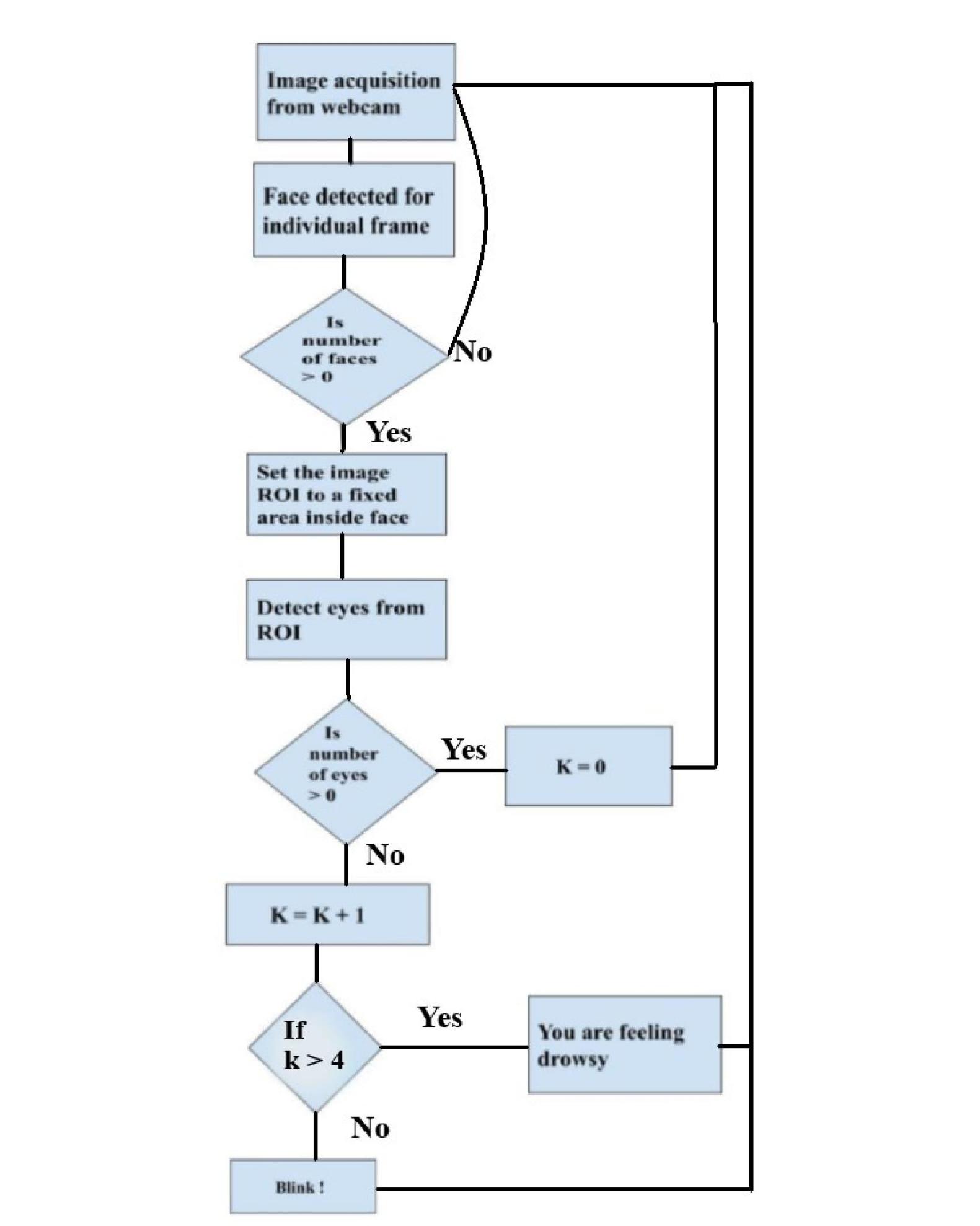
**3.4 DEVELOPING IMAGE PROCESSING SOLUTIONS WITH OPENCV & DLIB**

OpenCV was developed keeping image processing in mind. Every function and data struct of OpenCV concerns itself with an Image Processing library. Comparatively, Matlab is hugely of generic use & slow.Any usefulness can be accomplished by methods for tool kits in OpenCV, it might be money related tool compartments or explicit DNA tool stash.



Also the dlib library comes with a oriented gradients based face detector histogram a facial landmark predictor comes bundled in the library.Facial landmarks generated by dlib is an indexable list as described in below image



* 1. **ALGORITHM & IMPLEMENTATION**

**3.6 WORKING**

Drivers' faces are monitored throughout using a video or web camera. In order to detect the drowsiness, the first step is to detect the face using the set of frames taken by the camera. Then the location of the eyes is detected and the retina of the eye is continuously monitored. The captured image is sent to the processor for image processing. It converts the received image to digital signal using OpenCV.The digital signal is transmitted from the transmitter to the receiver. Both the transmitter and the receiver are paired up. The signal is then passed to the LPC2148, the microcontroller. If the signal crosses the threshold value of EAR for a given number of frames, then the alarm beeps and the speed of the vehicle is automatically reduced.



**CONVOLUTION NEURAL NETWORK**

The convolutional neural network, or CNN for short, is a specialized type of neural network model designed for working with two-dimensional image data, although they can be used with one-dimensional and three-dimensional data. Central to the convolutional neural network is the convolutional layer that gives the network its name. This layer performs an operation called a “*convolution* “. In the context of a convolutional neural network, a convolution is a linear operation that involves the multiplication of a set of weights with the input, much like a traditional neural network. Given that the technique was designed for two-dimensional input, the multiplication is performed between an array of input data and a two-dimensional array of weights, called a filter or a kernel.

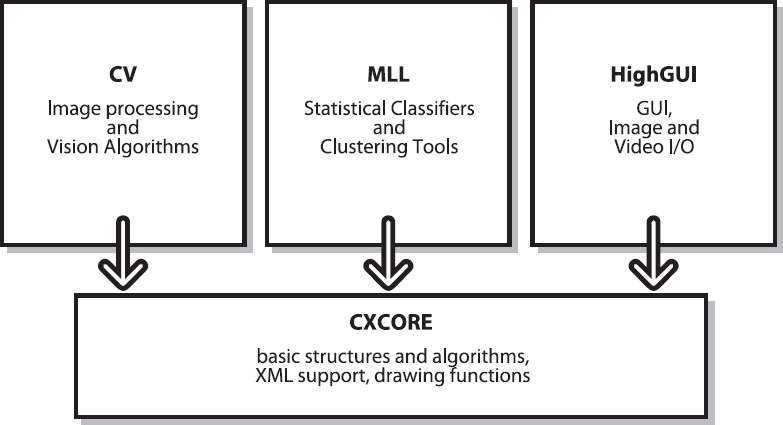
The filter is smaller than the input data and the type of multiplication applied between a filter-sized patch of the input and the filter is a dot product. A dot product is the element-wise multiplication between the filter-sized patch of the input and filter, which is then summed, always resulting in a single value. Because it results in a single value, the operation is often referred to as the “scalar product “. Using a filter smaller than the input is intentional as it allows the same filter (set of weights) to be multiplied by the input array multiple times at different points on the input. Specifically, the filter is applied systematically to each overlapping part or filter-sized patch of the input data, left to right, top to bottom. This systematic application of the same filter across an image is a powerful idea. If the filter is designed to detect a specific type of feature in the input, then the application of that filter systematically across the entire input image allows the filter an opportunity to discover that feature anywhere in the image. This capability is commonly referred to as translation invariance, e.g. the general interest in whether the feature is present rather than where it was present.

**OpenCV**

OpenCV is an open source computer vision library accessible in python coding language to code for visionary capabilities of our smart pc. OpenCV was expected for computational capability and having a high focus on ongoing picture location and distinguishing proof. OpenCV is coded with streamlined C and can take work with multicore processors. If we need progressively programmed improvement utilizing Intel models [Intel], you can purchase Intel's Integrated Performance Primitives (IPP) libraries [IPP]. These comprise low-level schedules in different algorithmic regions which are streamlined. OpenCV consequently utilizes the IPP library, at runtime if that library is introduced.

**OpenCV Structure and Content**

OpenCV left an Intel Research movement planned to drive CPU-raised applications. Toward this end, Intel pushed various endeavors that included continuous beam following and moreover 3D show dividers. One of the product engineers working for Intel at the time was visiting schools. He saw that two or three top school social events, like the MIT Media Lab, used to have well-made similarly as inside open PC vision foundations



**Getting to use Machine Learning for Computer Vision**

Typically, most calculations take an information vector having numerous highlights as info. Here, the quantity of highlights may number in the thousands. On the off chance that our undertaking is perceiving a particular sort of article— take for instance, an individual's face. The main issue that we experience is getting and marking the preparation information which falls into positive (for example there is a face in the window) and negative (for example no face) cases. We before long understand that countenances can show up at different scales: for example, their picture may consist of just a couple of pixels, or we may take a gander at an ear which is filling the entire screen.

More terrible still, faces are typically impeded. We need to characterize what we really mean when we state that a face is in the window. Subsequent to having marked the information that was acquired from different sources, we ought to choose which highlights we have to remove from these items. Likewise, we should recognize what objects we are after.

On the off chance that the countenances dependably seem upstanding, there is no purpose behind utilizing revolution invariant highlights and furthermore no explanation behind endeavoring to turn the articles before handling. All in all, we should attempt to discover highlights that express a little invariance in the articles.

These can be scale-tolerant histograms of angles or hues or even the prominent SIFT highlights. When we have imperative foundation window data, we would first be able to evacuate it so as to enable different articles to emerge. At that point we play out our picture handling. This may consist of normalizing the picture and after that registering the different highlights. The subsequent information vectors are altogether given the name that is related with the article, activity, or window. [4] When the information is gotten and changed over into highlight vectors, we separate the information into preparing sets, approval sets and test sets. It is fitting to do our learning, approval, and testing utilizing a cross-approval structure.

Here, the information is part of K subsets and we run different preparations (possibly approval) just as test sessions. Every session comprises different arrangements of information that assume the jobs of preparing (approval) and test. The test outcomes acquired from these different sessions are utilized for averaging to get the last execution result. An increasingly precise picture of how the classifier performs when sent in a task can be given by cross-approval. Since our information is prepared, we should pick a classifier.

Generally, the decision of the classifier is controlled by computational, information, and memory necessities. For specific applications, as online client inclination displays, we have to prepare the classifier rapidly.

In such a case, closest neighbors, ordinary Bayes, or choice trees ought to be a decent decision. At the point when memory is the essential thought, choice trees or neural systems are utilized for their space productivity.

When we have sufficient energy to prepare our classifier yet it needs to run rapidly, neural systems can be a decent decision, similarly as with ordinary Bayes classifiers and bolster vector machines.

When we have room schedule-wise to prepare yet require high exactness, at that point boosting and irregular trees are great decisions.

When we simply need a simple and reasonable check climate our highlights are picked well or not, at that point choice trees or closest neighbors ought to be utilized. For a decent out of the container characterization execution, boosting or irregular trees are attempted.

**3.7 DATA GATHERING**

The data is gathered from twitter and from Kaggle data as described in chapter 3

**LABELLING**

Data labelling is the manual curation of data by humans on machine learning and AI applications.

Labelling data is essential because computers have innumerable limitations, and some of them cannot be simply solved without human intervention. Actually, a computer can be programmed to perform hard calculations and automate activities that would be too burdensome to be done manually by us, however the same is unable to tell the difference between a car from a dog in a picture without proper training.

Hence it is important to label the data images during data set gathering.

**3.8 FEATURE ENGINEERING**

Feature engineering is the process of using the domain knowledge of the data to create features that makes machine learning algorithms work properly.

If feature engineering is performed properly, it helps to improve the power of prediction of machine learning algorithms by creating the features using the raw data that facilitate the machine learning process. Features in machine learning are very important, being building a block of datasets, the quality of the features in your dataset has a major impact on the quality of the insights you will get while using the dataset for machine learning.

However, depending on the different business problems in different industries it is not necessary the features should be the same features, so here you need to strongly understand the business goal of your data science project.

Where on the other hand, using the “feature selection” and “feature engineering” process you can improve the quality of your dataset’s features, which is a very tedious and difficult process. In this project many new features are created from the dataset

The dataset used for this model is created by us. To create the dataset, we wrote a script that captures eyes from a camera and stores in our local disk. We separated them into their respective labels ‘Open’ or ‘Closed’. The data was manually cleaned by removing the unwanted images which were not necessary for building the model. The data comprises around 7000 images of people’s eyes under different lighting conditions. After training the model on our dataset, we have attached the final weights and model architecture file “models/cnnCat2.h5”. Now, you can use this model to classify if a person’s eye is open or closed.

**3.9 TRAINING**

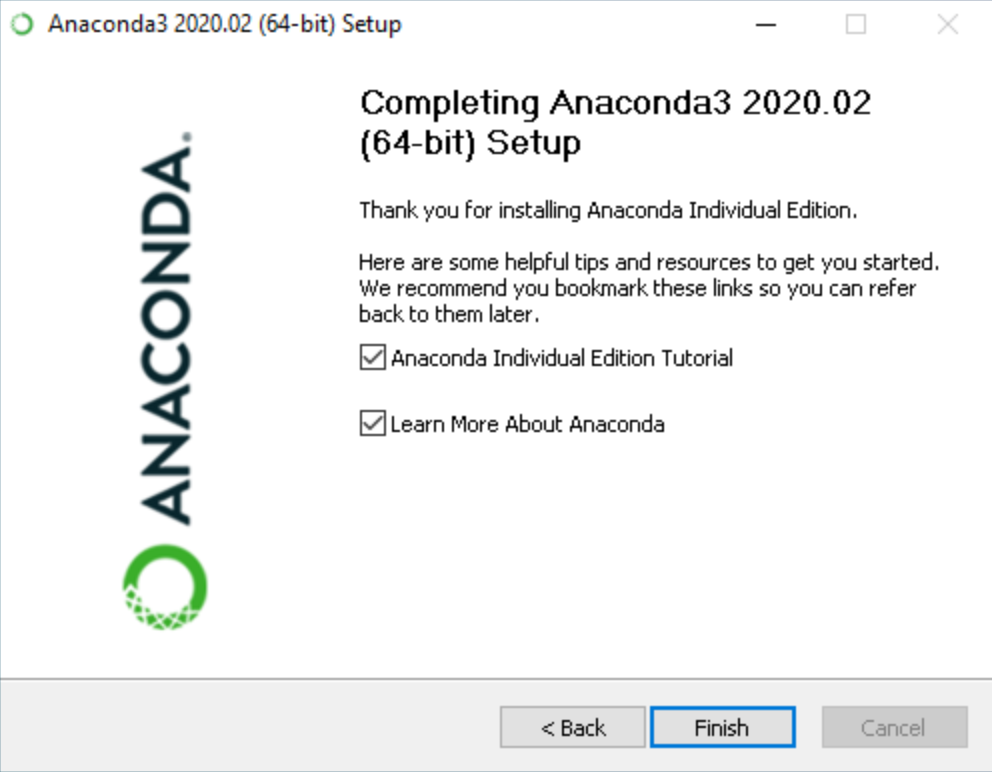
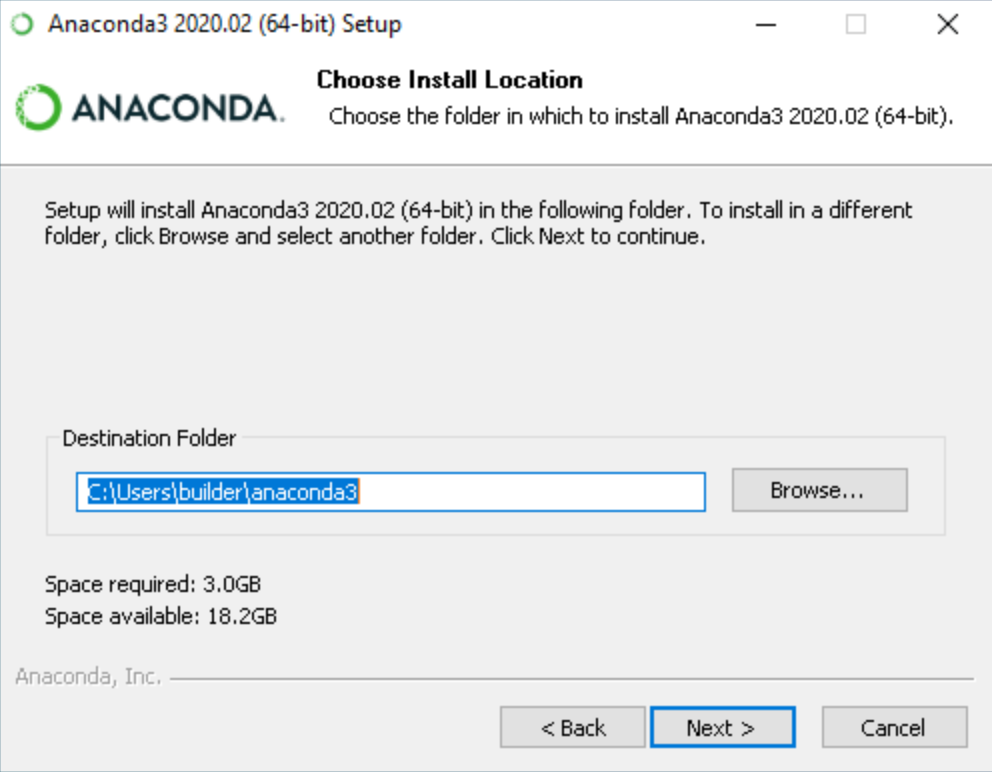
Training a model simply means learning (determining) good values for all the weights and the bias from labelled examples. In supervised learning, a machine learning algorithm builds a model by examining many examples and attempting to find a model that minimizes loss; this process is called empirical risk minimization**.** Loss is the penalty for a bad prediction. That is, loss is a number indicating how bad the model's prediction was on a single example. If the model's prediction is perfect, the loss is zero; otherwise, the loss is greater.

**3.10 TESTING**

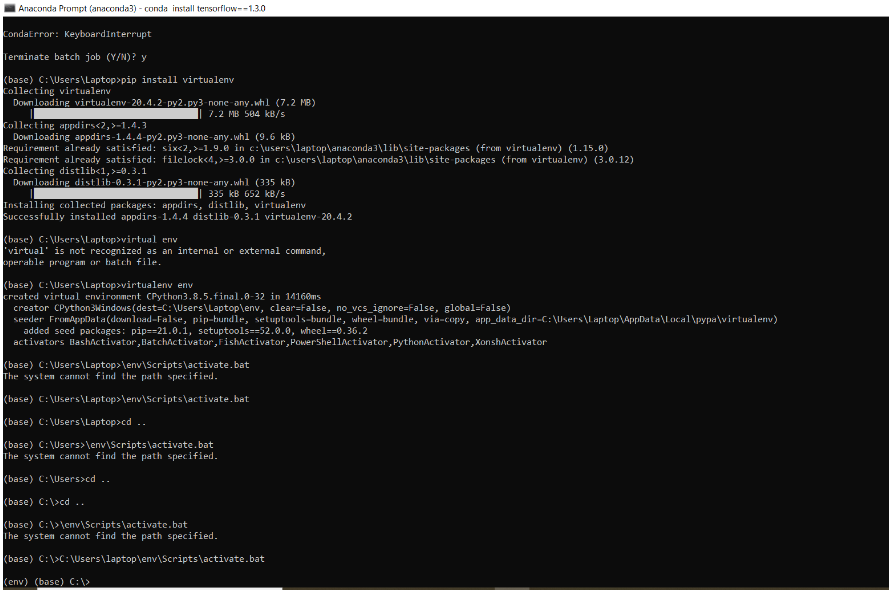
This corresponds to the final evaluation that the model goes through after the training phase has been completed. This step is critical to test the generalizability of the model. By using this set, we can get the working accuracy of our model. Training a model involves looking at training examples and learnin*g* from how off the model is by frequently evaluating it on the validation set. However, the last and most valuable pointer on the accuracy of a model is a result of running the model on the testing set when the training is complete.

**CHAPTER 4**

**EXPERIMENTAL RESULTS AND CONCLUSIONS**

****

**Figure 4.1 Installing Anaconda**

****

**Figure 4.2 Installing virtualenv package**

In this figure 4.1 we are working on the virtual environment. It is unique because, in this virtual environment we have dependencies and python libraries which are only required to execute our project.

Here we have “virtualenv” package installed on conda packages for creating various virtual environments for different projects, without any version mismatch errors.  
To install , simply type in the following command in the conda base environment prompt:

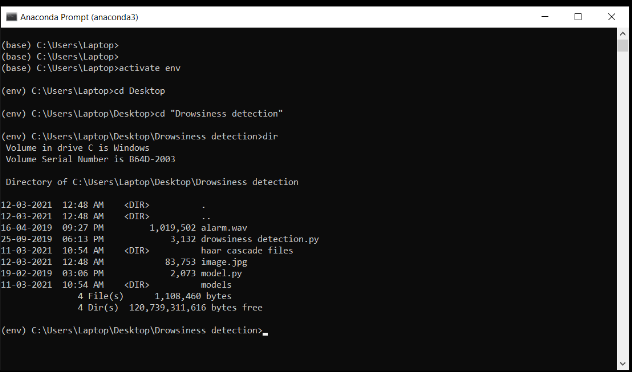
pip install virtualenv

To create a virtual environment, use the following command in conda prompt in base environment:

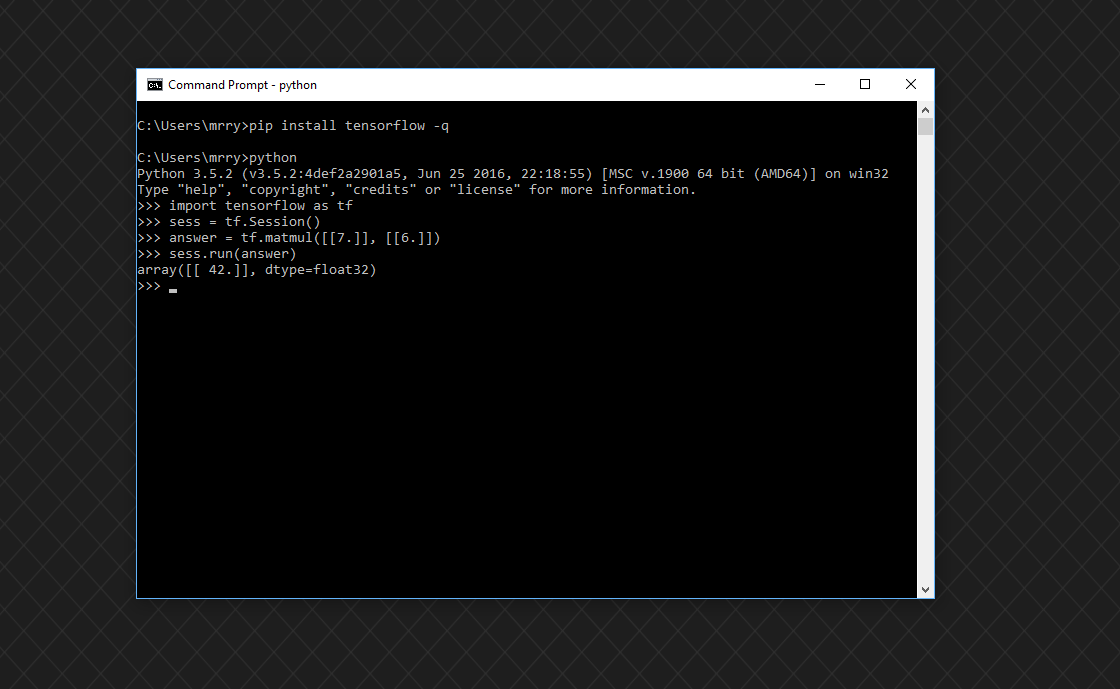
python3 -m venv env

To activate, use the following command:

activate env

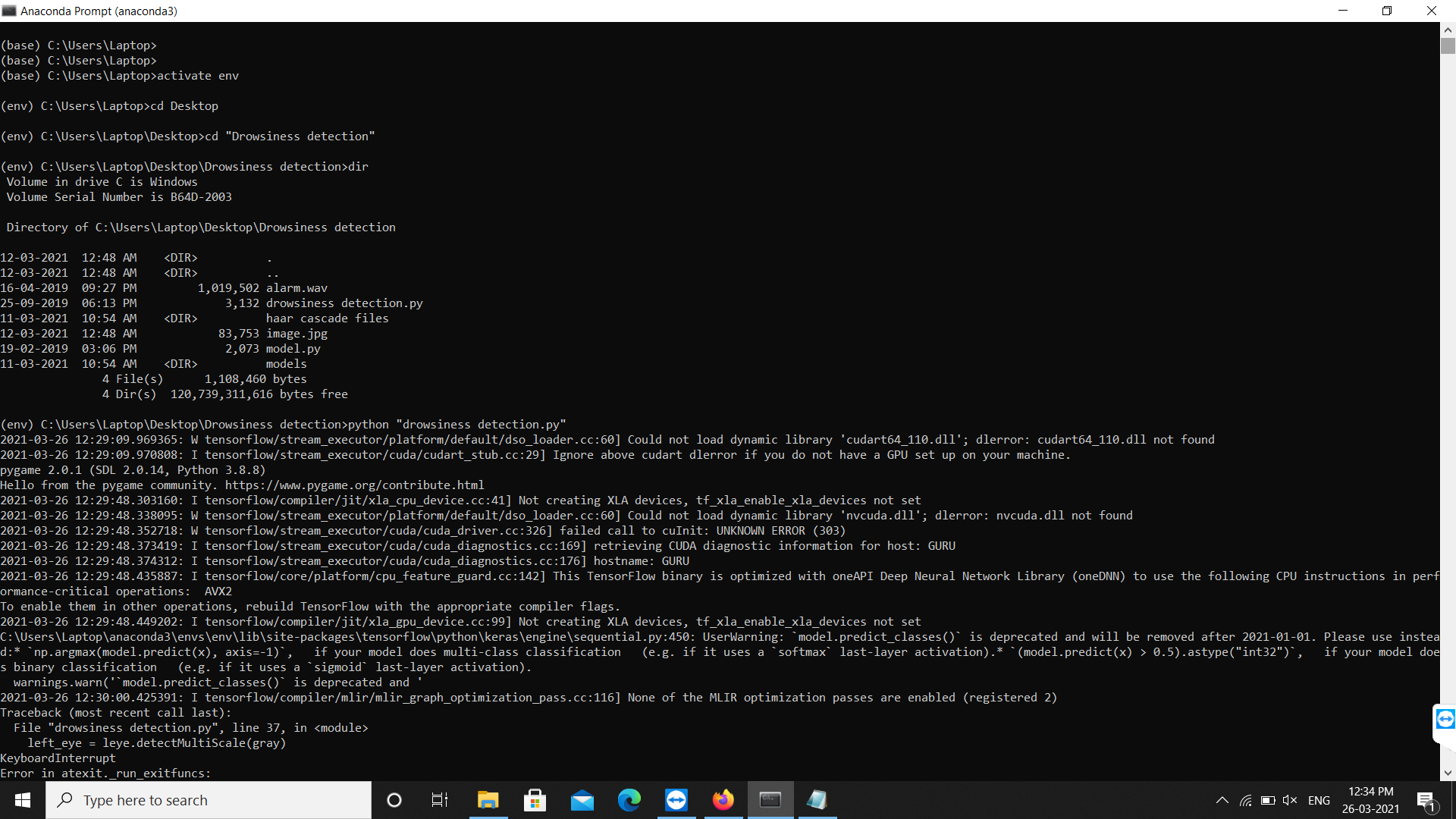
****

**Figure 4.3 Activating Vitual environment**



**Figure 4.4 Installing python libraries**

And then in the above command prompts, we installed packages such as Tensorflow, OpenCV, Keras, Pygame for its specific uses in this project.

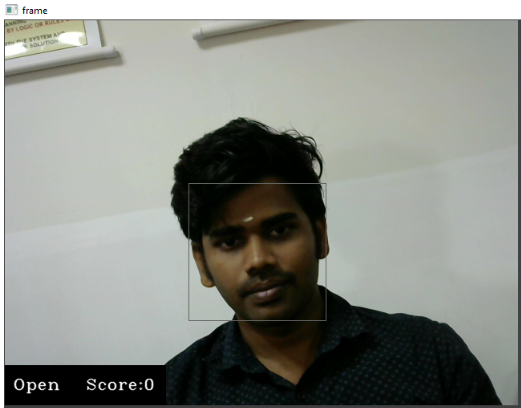


**Figure 4.6 Execution of drowsiness detection**

After installing all the required packages to run this project, simply type in the command prompt as:

python drowsiness detection.py

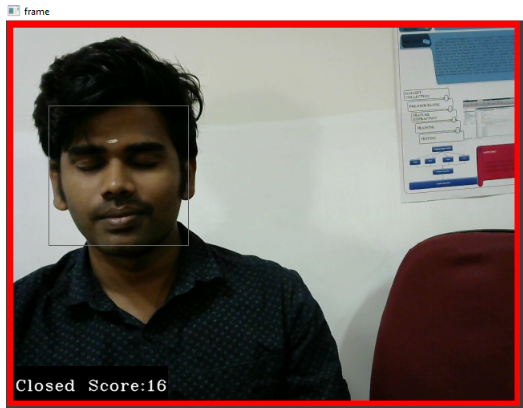
**4.10 RESULTS**

****

**Figure 4.7 Detection of active face**

In a while, a frame pop-ups and the drowsiness detectionstarts with recognition for faces in the input from the camera. Based on the pose of eyes, the score for drowsiness is calculated.

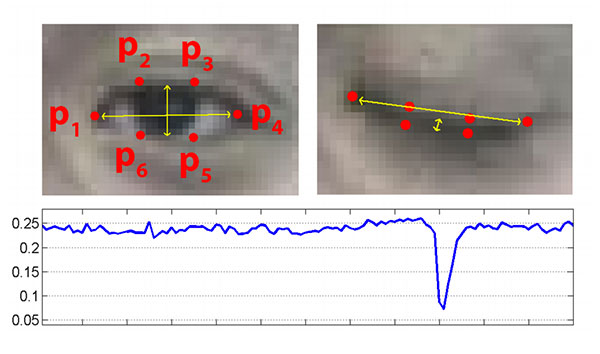
Here in this figure 4.6, the person blinks his eyes properly. So there is no such increase in the drowsiness score. Then, the alarm will not be triggered.

****

**Figure 4.8 Detectoin of drowsy face**

Here in this figure 4.7, the person does not blink his eyes properly. So there is such an increase in the drowsiness score. Then, the alarm will be triggered with a red frame that starts appearing as a drowsiness detection.

**Figure 4.9 Tensorflow graph generation**



**CHAPTER 5**

**CONCLUSION AND FUTURE WORK**

**5.1 CONCLUSION**

In this work, a novel framework for drowsiness detection system based on multilayers perceptron classifiers. It is specifically designed for embedded systems such as Android mobile. The role of the system is to detect facial landmarks from images and deliver the obtained data to the trained model to identify the driver's state. The purpose of the method is to reduce the model's size considering that current applications cannot be used in embedded systems due to their limited calculation and storage capacity. According to the experimental results, the size of the used model is small while having the accuracy rate of 92%. Hence, it can be integrated into advanced driver-assistance systems, the Driver drowsiness detection system, and mobile applications. However, there is still space for performance improvement. The further work will focus on detecting the distraction and yawning of the driver.

**5.2 FUTURE WORK**

The visualization results showed that the proposed method is capable of determining the drowsiness detection of people which can be further developed for use in other environments such as office, restaurant, and school. Furthermore, the work can be further improved by optimizing the facial detection algorithm, integrating other detection algorithms such as mask detection and human body detection, improving the computing power of the hardware, and calibrating the camera perspective view.

# APPENDIX I

# 

# WORKING ENVIRONMENT

**HARDWARE SPECIFICATION**

|  |  |
| --- | --- |
| **Processor** | Intel® Core™ i5 |
| **RAM** | 8 GB |
| **Display** | 20” wide monitor |
| **Keyboard** | Dell USB Keyboard |
| **Mouse** | Optical Mouse |

# 

# SOFTWARE SPECIFICATION

# 

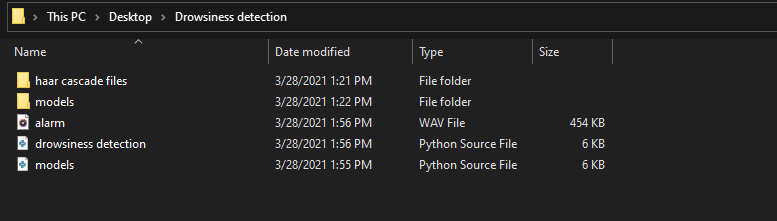
|  |  |
| --- | --- |
| **Operating System** | Windows 10 |
| **Professional Tool** | Anaconda |
| **Languages used** | Python |

# 

# APPENDIX II

# CODING

The contents of the directory are:



**The Source Code of our main file looks like this:**

import cv2

import os

from keras.models import load\_model

import numpy as np

from pygame import mixer

import time

mixer.init()

sound = mixer.Sound('alarm.wav')

face = cv2.CascadeClassifier('haar cascade files\haarcascade\_frontalface\_alt.xml')

leye = cv2.CascadeClassifier('haar cascade files\haarcascade\_lefteye\_2splits.xml')

reye = cv2.CascadeClassifier('haar cascade files\haarcascade\_righteye\_2splits.xml')

lbl=['Close','Open']

model = load\_model('models/cnncat2.h5')

path = os.getcwd()

cap = cv2.VideoCapture(0)

font = cv2.FONT\_HERSHEY\_COMPLEX\_SMALL

count=0

score=0

thicc=2

rpred=[99]

lpred=[99]

**while**(**True**):

ret, frame = cap.read()

height,width = frame.shape[:2]

gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

faces = face.detectMultiScale(gray,minNeighbors=5,scaleFactor=1.1,minSize=(25,25))

left\_eye = leye.detectMultiScale(gray)

right\_eye = reye.detectMultiScale(gray)

cv2.rectangle(frame, (0,height-50) , (200,height) , (0,0,0) , thickness=cv2.FILLED )

**for** (x,y,w,h) **in** faces:

cv2.rectangle(frame, (x,y) , (x+w,y+h) , (100,100,100) , 1 )

**for** (x,y,w,h) **in** right\_eye:

r\_eye=frame[y:y+h,x:x+w]

count=count+1

r\_eye = cv2.cvtColor(r\_eye,cv2.COLOR\_BGR2GRAY)

r\_eye = cv2.resize(r\_eye,(24,24))

r\_eye= r\_eye/255

r\_eye= r\_eye.reshape(24,24,-1)

r\_eye = np.expand\_dims(r\_eye,axis=0)

rpred = model.predict\_classes(r\_eye)

**if**(rpred[0]==1):

lbl='Open'

**if**(rpred[0]==0):

lbl='Closed'

break

**for** (x,y,w,h) **in** left\_eye:

l\_eye=frame[y:y+h,x:x+w]

count=count+1

l\_eye = cv2.cvtColor(l\_eye,cv2.COLOR\_BGR2GRAY)

l\_eye = cv2.resize(l\_eye,(24,24))

l\_eye= l\_eye/255

l\_eye=l\_eye.reshape(24,24,-1)

l\_eye = np.expand\_dims(l\_eye,axis=0)

lpred = model.predict\_classes(l\_eye)

**if**(lpred[0]==1):

lbl='Open'

**if**(lpred[0]==0):

lbl='Closed'

break

**if**(rpred[0]==0 and lpred[0]==0):

score=score+1

cv2.putText(frame,"Closed",(10,height-20), font, 1,(255,255,255),1,cv2.LINE\_AA)

# if(rpred[0]==1 or lpred[0]==1):

**else**:

score=score-1

cv2.putText(frame,"Open",(10,height-20), font, 1,(255,255,255),1,cv2.LINE\_AA)

**if**(score<0):

score=0

cv2.putText(frame,'Score:'+str(score),(100,height-20), font, 1,(255,255,255),1,cv2.LINE\_AA)

**if**(score>15):

#person is feeling sleepy so we beep the alarm

cv2.imwrite(os.path.join(path,'image.jpg'),frame)

**try**:

sound.play()

**except**: # isplaying = False

pass

**if**(thicc<16):

thicc= thicc+2

**else**:

thicc=thicc-2

**if**(thicc<2):

thicc=2

cv2.rectangle(frame,(0,0),(width,height),(0,0,255),thicc)

cv2.imshow('frame',frame)

**if** cv2.waitKey(1) & 0xFF == ord('q'):

break

cap.release()

cv2.destroyAllWindows()

Let’s start our project and see the working of our project. To start the project, you need to open a command prompt, go to the directory where our main file “drowsiness detection.py” exists. Run the script with this command.

python “drowsiness detection.py”

It may take a few seconds to open the webcam and start detection.

**REFERENCES**

**[Comprehensive Drowsiness Level Detection Model Combining Multimodal Information](https://ieeexplore.ieee.org/document/8933427/)**

[Mika Sunagawa](https://ieeexplore.ieee.org/author/37088234557);[Shin-ichi Shikii](https://ieeexplore.ieee.org/author/37088235549);[Wataru Nakai](https://ieeexplore.ieee.org/author/37088234133);[Makoto Mochizuki](https://ieeexplore.ieee.org/author/37088235222);[Koichi Kusukame](https://ieeexplore.ieee.org/author/37086492796);[Hiroki Kitajima](https://ieeexplore.ieee.org/author/37088235264)

[IEEE Sensors Journal](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=7361)

Year: 2020 | Volume: 20, [Issue: 7](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=9025730) | Journal Article | Publisher: IEEE

**[A Real-time Driving Drowsiness Detection Algorithm With Individual Differences Consideration](https://ieeexplore.ieee.org/document/8930504/)**

[Feng You](https://ieeexplore.ieee.org/author/37086070817);[Xiaolong Li](https://ieeexplore.ieee.org/author/37086695935);[Yunbo Gong](https://ieeexplore.ieee.org/author/37087130488);[Hailwei Wang](https://ieeexplore.ieee.org/author/37086197027);[Hongyi Li](https://ieeexplore.ieee.org/author/37087130308)

[IEEE Access](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6287639)

Year: 2019 | [Volume: 7](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8600701) | Journal Article | Publisher: IEEE

**[A Smartphone-Based Drowsiness Detection and Warning System for Automotive Drivers](https://ieeexplore.ieee.org/document/8595428/)**

[Anirban Dasgupta](https://ieeexplore.ieee.org/author/38237357000);[Daleef Rahman](https://ieeexplore.ieee.org/author/37086373635);[Aurobinda Routray](https://ieeexplore.ieee.org/author/38571794700)

[IEEE Transactions on Intelligent Transportation Systems](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6979)

Year: 2019 | Volume: 20, [Issue: 11](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8892698) | Journal Article | Publisher: IEEE

**[EDDD: Event-Based Drowsiness Driving Detection Through Facial Motion Analysis With Neuromorphic Vision Sensor](https://ieeexplore.ieee.org/document/8990081/)**

[Guang Chen](https://ieeexplore.ieee.org/author/38251904000);[Lin Hong](https://ieeexplore.ieee.org/author/37088396907);[Jinhu Dong](https://ieeexplore.ieee.org/author/37087244876);[Peigen Liu](https://ieeexplore.ieee.org/author/37088396059);[Jörg Conradt](https://ieeexplore.ieee.org/author/37326140900);[Alois Knoll](https://ieeexplore.ieee.org/author/37276234100)

[IEEE Sensors Journal](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=7361)

Year: 2020 | Volume: 20, [Issue: 11](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=9086908) | Journal Article | Publisher: IEEE

**[Driver Drowsiness Detection Using Condition-Adaptive Representation Learning Framework](https://ieeexplore.ieee.org/document/8580568/)**

[Jongmin Yu](https://ieeexplore.ieee.org/author/37085574898);[Sangwoo Park](https://ieeexplore.ieee.org/author/37086250308);[Sangwook Lee](https://ieeexplore.ieee.org/author/37678069100);[Moongu Jeon](https://ieeexplore.ieee.org/author/37666359200)

[IEEE Transactions on Intelligent Transportation Systems](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6979)

Year: 2019 | Volume: 20, [Issue: 11](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8892698) | Journal Article | Publisher: IEEE

**[A Survey on State-of-the-Art Drowsiness Detection Techniques](https://ieeexplore.ieee.org/document/8704263/)**

[Muhammad Ramzan](https://ieeexplore.ieee.org/author/37073110500);[Hikmat Ullah Khan](https://ieeexplore.ieee.org/author/37086842388);[Shahid Mahmood Awan](https://ieeexplore.ieee.org/author/37086842449);[Amina Ismail](https://ieeexplore.ieee.org/author/37086842720);[Mahwish Ilyas](https://ieeexplore.ieee.org/author/37086841504);[Ahsan Mahmood](https://ieeexplore.ieee.org/author/37086841044)

[IEEE Access](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6287639)

Year: 2019 | [Volume: 7](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8600701) | Journal Article | Publisher: IEEE

**[An Effective Hybrid Model for EEG-Based Drowsiness Detection](https://ieeexplore.ieee.org/document/8718312/)**

[Umit Budak](https://ieeexplore.ieee.org/author/38468148300);[Varun Bajaj](https://ieeexplore.ieee.org/author/38526003500);[Yaman Akbulut](https://ieeexplore.ieee.org/author/37086238971);[Orhan Atila](https://ieeexplore.ieee.org/author/37086934120);[Abdulkadir Sengur](https://ieeexplore.ieee.org/author/38228506200)

[IEEE Sensors Journal](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=7361)

Year: 2019 | Volume: 19, [Issue: 17](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8789581) | Journal Article | Publisher: IEEE

**[Heart Rate Variability-Based Driver Drowsiness Detection and Its Validation With EEG](https://ieeexplore.ieee.org/document/8520803/)**

[Koichi Fujiwara](https://ieeexplore.ieee.org/author/37286130300);[Erika Abe](https://ieeexplore.ieee.org/author/37085413644);[Keisuke Kamata](https://ieeexplore.ieee.org/author/37085635316);[Chikao Nakayama](https://ieeexplore.ieee.org/author/37085735911);[Yoko Suzuki](https://ieeexplore.ieee.org/author/37085424004);[Toshitaka Yamakawa](https://ieeexplore.ieee.org/author/37085425937);[Toshihiro Hiraoka](https://ieeexplore.ieee.org/author/37446852100);[Manabu Kano](https://ieeexplore.ieee.org/author/37300929200);[Yukiyoshi Sumi](https://ieeexplore.ieee.org/author/37086838814);[Fumi Masuda](https://ieeexplore.ieee.org/author/37087334710);[Masahiro Matsuo](https://ieeexplore.ieee.org/author/37085726309);[Hiroshi Kadotani](https://ieeexplore.ieee.org/author/37085737072)

[IEEE Transactions on Biomedical Engineering](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=10)

Year: 2019 | Volume: 66, [Issue: 6](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8718429) | Journal Article | Publisher: IEE

**[Combined EEG-Gyroscope-tDCS Brain Machine Interface System for Early Management of Driver Drowsiness](https://ieeexplore.ieee.org/document/8080143/)**

[Gang Li](https://ieeexplore.ieee.org/author/37085727914);[Wan-Young Chung](https://ieeexplore.ieee.org/author/37085490656)

[IEEE Transactions on Human-Machine Systems](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6221037)

Year: 2018 | Volume: 48, [Issue: 1](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8259354) | Journal Article | Publisher: IEEE

**[Driver Drowsiness Detection Based on Respiratory Signal Analysis](https://ieeexplore.ieee.org/document/8744224/)**

[FedericoGuede-Fernández](https://ieeexplore.ieee.org/author/37085637983);[MireyaFernández-Chimeno](https://ieeexplore.ieee.org/author/38273205200);[Juan Ramos-Castro](https://ieeexplore.ieee.org/author/38317866600);[Miguel A. García-González](https://ieeexplore.ieee.org/author/38317462900)

[IEEE Access](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6287639)

Year: 2019 | [Volume: 7](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8600701) | Journal Article | Publisher: IEEE

**[Enhanced Drowsiness Detection Using Deep Learning: An fNIRS Study](https://ieeexplore.ieee.org/document/8846024/)**

[MAsjidTanveer](https://ieeexplore.ieee.org/author/37085740992);[M.Jawad Khan](https://ieeexplore.ieee.org/author/37085392591);[M. Jahangir Qureshi](https://ieeexplore.ieee.org/author/37087032462);[Noman Naseer](https://ieeexplore.ieee.org/author/37085392509);[Keum-Shik Hong](https://ieeexplore.ieee.org/author/37086688413)

[IEEE Access](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6287639)

Year: 2019 | [Volume: 7](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8600701) | Journal Article | Publisher: IEEE

**[Driver Drowsiness Recognition via 3D Conditional GAN and Two-Level Attention Bi-LSTM](https://ieeexplore.ieee.org/document/8926344/)**

[Yaocong Hu](https://ieeexplore.ieee.org/author/37086563330);[Mingqi Lu](https://ieeexplore.ieee.org/author/37086559239);[Chao Xie](https://ieeexplore.ieee.org/author/37088473745);[Xiaobo Lu](https://ieeexplore.ieee.org/author/37538346800)

[IEEE Transactions on Circuits and Systems for Video Technology](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=76)

Year: 2020 | Volume: 30, [Issue: 12](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=9280452) | Journal Article | Publisher: IEEE

**[A Mobile Application for Driver's Drowsiness Monitoring based on PERCLOS Estimation](https://ieeexplore.ieee.org/document/8863164/)**

Gabriel Soares;Danilo de Lima;Arthur Miranda Neto

[IEEE Latin America Transactions](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=9907)

Year: 2019 | Volume: 17, [Issue: 02](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8863160) | Journal Article | Publisher: IEEE

**[Design and Implementation of a Drowsiness-Fatigue-Detection System Based on Wearable Smart Glasses to Increase Road Safety](https://ieeexplore.ieee.org/document/8493318/)**

[Wan-Jung Chang](https://ieeexplore.ieee.org/author/37085385685);[Liang-Bi Chen](https://ieeexplore.ieee.org/author/37086046066);[Yu-Zung Chiou](https://ieeexplore.ieee.org/author/37283500200)

[IEEE Transactions on Consumer Electronics](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=30)

Year: 2018 | Volume: 64, [Issue: 4](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8554318) | Journal Article | Publisher: IEEE

**[A Hybrid Scheme for Drowsiness Detection Using Wearable Sensors](https://ieeexplore.ieee.org/document/8664170/)**

[Aqsa Mehreen](https://ieeexplore.ieee.org/author/37086852329);[Syed Muhammad Anwar](https://ieeexplore.ieee.org/author/37604161600);[Muhammad Haseeb](https://ieeexplore.ieee.org/author/38109556600);[Muhammad Majid](https://ieeexplore.ieee.org/author/37085639604);[Muhammad Obaid Ullah](https://ieeexplore.ieee.org/author/37085796429)

[IEEE Sensors Journal](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=7361)

Year: 2019 | Volume: 19, [Issue: 13](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8730435) | Journal Article | Publisher: IEEE

**[Real-Time Driver-Drowsiness Detection System Using Facial Features](https://ieeexplore.ieee.org/document/8808931/)**

[Wanghua Deng](https://ieeexplore.ieee.org/author/37086962467);[Ruoxue Wu](https://ieeexplore.ieee.org/author/37086962777)

[IEEE Access](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6287639)

Year: 2019 | [Volume: 7](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8600701) | Journal Article | Publisher: IEEE

**[Real Time Driver Fatigue Detection System Based on Multi-Task ConNN](https://ieeexplore.ieee.org/document/8949469/)**

[Burcu Kir Savaş](https://ieeexplore.ieee.org/author/37086876154);[Yaşar Becerikli](https://ieeexplore.ieee.org/author/37265887200)

[IEEE Access](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6287639)

Year: 2020 | [Volume: 8](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8948470) | Journal Article | Publisher: IEEE

**[A Human-Vehicle Collaborative Driving Framework for Driver Assistance](https://ieeexplore.ieee.org/document/8556397/)**

[Duy Tran](https://ieeexplore.ieee.org/author/37085343634);[Jianhao Du](https://ieeexplore.ieee.org/author/38235968700);[Weihua Sheng](https://ieeexplore.ieee.org/author/37276312200);[Denis Osipychev](https://ieeexplore.ieee.org/author/37085715249);[Yuge Sun](https://ieeexplore.ieee.org/author/37291597300);[He Bai](https://ieeexplore.ieee.org/author/37086961133)

[IEEE Transactions on Intelligent Transportation Systems](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6979)

Year: 2019 | Volume: 20, [Issue: 9](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8815899) | Journal Article | Publisher: IEEE

**[Real-Time Embedded Vision System for the Watchfulness Analysis of Train Drivers](https://ieeexplore.ieee.org/document/8926370/)**

[Carlo Alberto Avizzano](https://ieeexplore.ieee.org/author/37274154300);[Paolo Tripicchio](https://ieeexplore.ieee.org/author/37594795600);[Emanuele Ruffaldi](https://ieeexplore.ieee.org/author/37394369300);[Alessandro Filippeschi](https://ieeexplore.ieee.org/author/37640067200);[Juan Manuel Jacinto-Villegas](https://ieeexplore.ieee.org/author/37061492300)

[IEEE Transactions on Intelligent Transportation Systems](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6979)

Year: 2021 | Volume: 22, [Issue: 1](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=9307358) | Journal Article | Publisher: IEEE

**[Driver Monitoring Using Sparse Representation With Part-Based Temporal Face Descriptors](https://ieeexplore.ieee.org/document/8628243/)**

[Chien-Yu Chiou](https://ieeexplore.ieee.org/author/37086088187);[Wei-Cheng Wang](https://ieeexplore.ieee.org/author/37085555770);[Shueh-Chou Lu](https://ieeexplore.ieee.org/author/37087227665);[Chun-Rong Huang](https://ieeexplore.ieee.org/author/37068864300);[Pau-Choo Chung](https://ieeexplore.ieee.org/author/37287692900);[Yun-Yang Lai](https://ieeexplore.ieee.org/author/37087224674)

[IEEE Transactions on Intelligent Transportation Systems](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6979)

Year: 2020 | Volume: 21, [Issue: 1](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8946909) | Journal Article | Publisher: IEEE

**[Integration of Ensemble and Evolutionary Machine Learning Algorithms for Monitoring Diver Behavior Using Physiological Signals](https://ieeexplore.ieee.org/document/8753479/)**

[Afsaneh Koohestani](https://ieeexplore.ieee.org/author/37086612021);[Moloud Abdar](https://ieeexplore.ieee.org/author/37085993050);[Abbas Khosravi](https://ieeexplore.ieee.org/author/37407922100);[Saeid Nahavandi](https://ieeexplore.ieee.org/author/37268675200);[Mahereh Koohestani](https://ieeexplore.ieee.org/author/37086925019)

[IEEE Access](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6287639)

Year: 2019 | [Volume: 7](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8600701) | Journal Article | Publisher: IEEE