**DRIVER DROWSINESS DETECTION SYSTEM USING DEEP LEARNING**

**Borra Umesh Chandra**

1Department of Electrical and Electronics Engineering

SRM Institute of Science and Technology

Chennai, India

|  |
| --- |
|  |

**1.INTRODUCTION**

Driver drowsiness and weariness are some of the significant reasons for engine vehicle mishaps. Improvement of a tiredness recognition framework is considered both a modern and educational test. In the vehicle area, a driver-ready control framework has been created by the OpenCV that cautions the driver associated with being languid. It utilizes a vehicle-mounted camera that is associated with the path takeoff cautioning framework. Following the comparable endeavor, a sluggishness discovery framework had been presented by Bosch, which settles on choices utilizing the information got from the sensor that is situated at the directing wheel, vehicle's driving speed, blinker, and the camera utilized for path help mounted at the front of the vehicle. Be that as it may, such wellbeing frameworks utilized for distinguishing laziness aren't far and wide and is whimsical among drivers since they are accessible just in extravagance vehicles

As of late, the field of AI has made spearheading progress, particularly in the space of profound neural learning. Hence, the utilization of new advancements and plans can be considered as a compelling way to deal with increasing the effectiveness of continuous laziness discovery just as to give a component that can be generally utilized by drivers. In this unique situation, we propose a profound learning-based way to deal with recognizing the sluggishness of drivers continuously utilizing the OpenCV YOLO model, which is a profound learning-based constant article discovery calculation and inception v3 learned model, that plays out the undertaking of drowsy characterization.

Humans have always invented machines and devised techniques to ease and protect their lives, for mundane activities like traveling to work, or for more interesting purposes like aircraft travel. With the advancement in technology, modes of transportation kept on advancing and our dependency on it started increasing exponentially. It has greatly affected our lives as we know it. Now, we can travel to places at a pace that even our grandparents wouldn’t have thought possible. In modern times, almost everyone in this world uses some sort of transportation every day

Some people are rich enough to have their own vehicles while others use public transportation. However, there are some rules and codes of conduct for those who drive irrespective of their social status. One of them is staying alert and active while driving. Neglecting our duties towards safer travel has enabled hundreds of thousands of tragedies to get associated with this wonderful invention every year. It may seem like a trivial thing to most folks but following rules and regulations on the road is of utmost importance.

While on road, an automobile wields the most power and in irresponsible hands, it can be destructive and sometimes, that carelessness can harm lives even of the people on the road. One kind of carelessness is not admitting when we are too tired to drive. In order to monitor and prevent a destructive outcome from such negligence, many researchers have written research papers on driver drowsiness detection systems. But at times, some of the points and observations made by the system are not accurate enough. Hence, to provide data and another perspective on the problem at hand, in order to improve their implementations and to further optimize the solution, this project has been done

**2.METHOD**

The annihilation of one’s life due to road accidents has been increasing due to human fallacy. Various awareness programs have been commemorated to eradicate portent possibilities over crust but the graph stays exponential. In this paper, we’ve tried to incorporate a lightweight system for scrutinizing and conceding Driver’s Drowsiness state using eyes as our subject, with the Transfer Learning approach. Here, we have subsumed the Inception V3 model over MRL Eye Dataset, stratifying it into 02 classes as Open and Closed eyes. In our study, we achieved an accuracy of more than 92% and cocksure our efficacy for creating an insubstantial as well as an exemplary model for Driver Drowsiness Detection.

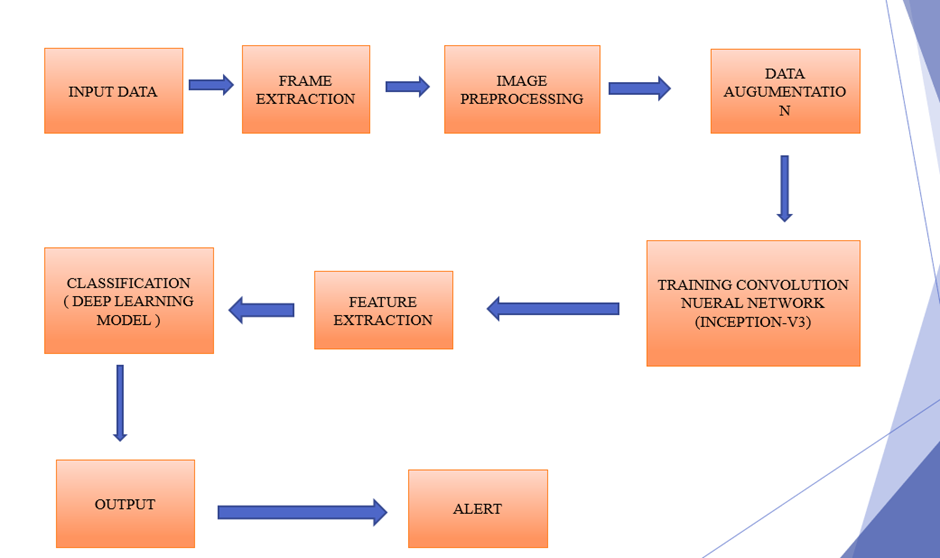


Figure 1. System Design for Driver drowsiness Detection using Convolution Neural Network

# 2.1 Block Description

1. Input From Data Set / Camera

2. Image Pre-processing and Data Augmentation.

3. Image Segmentation

4. Transfer Learning (Inception V3)

5. Feature Extraction

6. Classification

7. Output

**2.2 Convolution Neural Network**

It is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other.The architecture performs a better fitting to the image dataset due to the reduction in the number of parameters involved and reusability of weights.The architecture performs a better fitting to the image dataset due to the reduction in the number of parameters involved and reusability of weights. It is widely used in Pattern Recognition, Image Processing and Visual Computing.

**2.3 Video Processing**

Reading Video Frames For this step, OpenCV-python, a python version of OpenCV library, is used. Either can be done from a dataset or a camera in a real-time manner. Detecting Faces and Eyes Detecting faces with haar cascade frontal face detector and It is an Object Detection Algorithm used to identify faces in an image or a real time video.

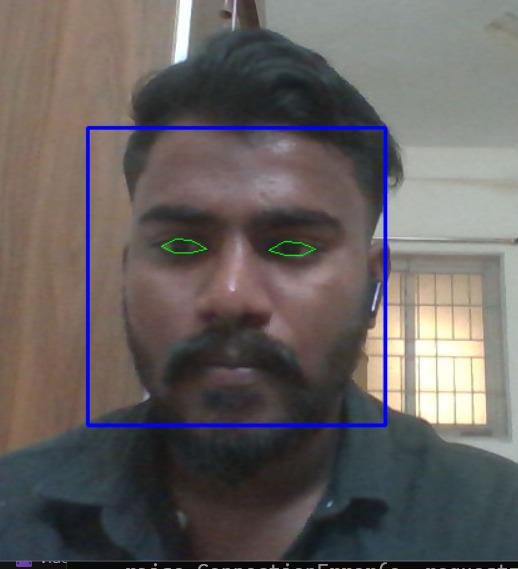


Fig 2: Depiction of facial features using haar cascade

**2.4 Image Preprocessing and Data Augumentation**

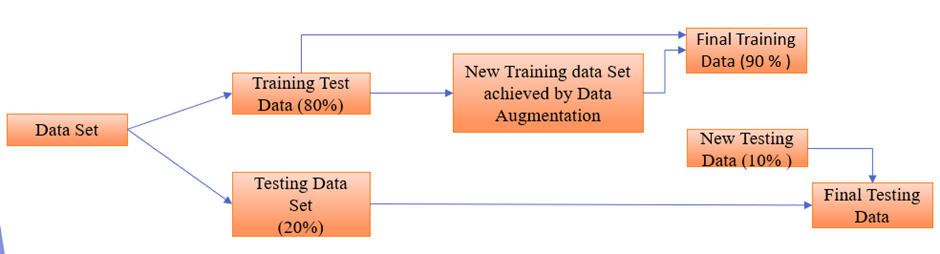


Fig 3: Pre-Processing and Data Augmentation

\*MRL Eye data set is used for the following project.

\*Eye State is the most important attribute for the model.

\*Available data is divided among the testing and training.

\*Data Augmentation: To generate new data using existed data

\*The data is divided for training and testing in the ratio of 80:20, then with the help of data augmentation new data will be generated and the further ratio will be divided as fig-2.

**2.5 Training a Convolution Neural Network**

There are three processes included in this section:

1. Image Segmentation

Partitioning image into different sections by using contour, Contour is an continues line or curve that cover or bound the region of object in an image. OpenCV read images in BGR format. Image turns into feature vector, then it is given as input to the Convolution neural network

1. Transfer Learning

It is a research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. It only imports the knowledge but not the result of it as it is trained in more than 1000 scenarios by industry. Inception v3 is a widely-used image recognition model that has been shown to attain greater than 78.1% accuracy on the ImageNet dataset and The Knowledge of solving will be transferred but not past output, this way the model goes for the best of its accuracy.

1. Training the model (Inception V3)

Inception Layer is a combination of all those layers namely, 1×1 Convolutional layer, 3×3 Convolutional layer, 5×5 Convolutional layer with their output filter banks concatenated into a single output vector forming the input of the next stage. Max pooling is a pooling operation that selects the maximum element from the region of the feature map covered by the filter. Thus, the output after max-pooling layer would be a feature map containing the most prominent features of the previous feature. Dropout layers help in controlling for overfitting as it drops a faction of parameters. Dense layer is the regular fully-connected layer with a specific activation function 1×1 Convolutional layer before applying another layer, which is mainly used for dimensionality reduction and all Layers must be train to get higher accuracy.

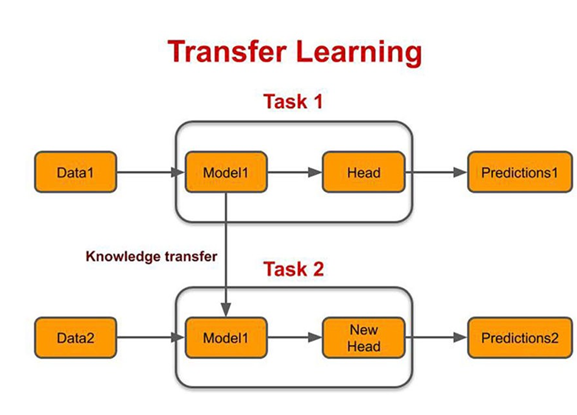


Fig4: Depicts about Basic principle of transfer learning

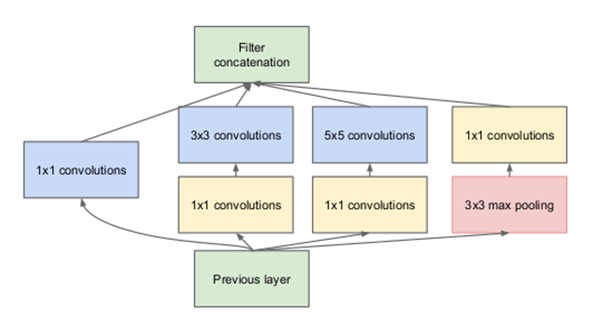


Fig 5: Basic Architecture of Inception V3 Model

**3.ALGORITHM:**

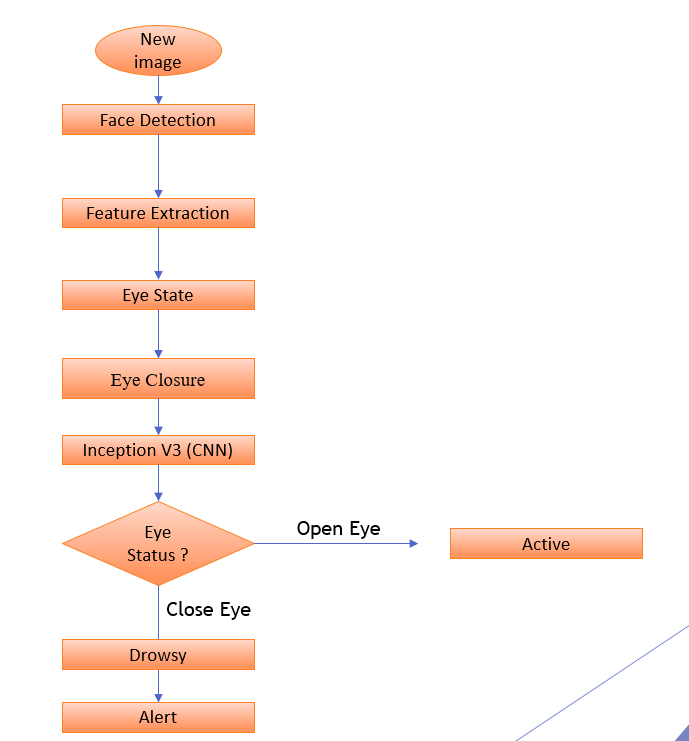
****

Fig 6: Algorithm of Driver drowsiness Detection System

**3.1 Pseudocode**

After training of model using transfer learning, it will be linked to OpenCV for execution

Importation of Haar Cascade:

1) ace\_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade\_frontalface\_default.xml')

eye\_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade\_eye.xml')

OpenCV Face Detection :

''This script uses OpenCV's haarcascade (face and eye cascade) to detect face

and eyes in a given input image.'''

#Import necessary libraries

import cv2 as cv

import numpy as np

#Load face cascade and hair cascade from haarcascades folder

face\_cascade = cv.CascadeClassifier("haarcascades/haarcascade\_frontalface\_default.xml")

eye\_cascade = cv.CascadeClassifier("haarcascades/haarcascade\_eye.xml")

#Read image in img and convert it to grayscale and store in gray.

#Image is converted to grayscale, as face cascade doesn't require to operate on coloured images.

img = cv.imread('images/test.jpeg')

gray = cv.cvtColor(img, cv.COLOR\_BGR2GRAY)

#Detect all faces in image.

faces = face\_cascade.detectMultiScale(gray, 1.3, 5)

#Draw a rectangle over the face, and detect eyes in faces

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

cv.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)

#ROI is region of interest with area having face inside it.

roi\_gray = gray[y:y+h, x:x+w]

roi\_color = img[y:y+h, x:x+w]

#Detect eyes in face

eyes = eye\_cascade.detectMultiScale(roi\_gray)

for (ex,ey,ew,eh) in eyes:

cv.rectangle(roi\_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)

cv.imshow('Image', img)

cv.waitKey(0)

cv.destroyAllWindows()

MAIN CODE

'''This script detects if a person is drowsy or not,using dlib and eye aspect ratio

calculations. Uses webcam video feed as input.'''

#Import necessary libraries

from scipy.spatial import distance

from imutils import face\_utils

import numpy as np

import pygame #For playing sound

import time

import dlib

import cv2

#Initialize Pygame and load music

pygame.mixer.init()

pygame.mixer.music.load('audio/alert.wav')

#Minimum threshold of eye aspect ratio below which alarm is triggerd

EYE\_ASPECT\_RATIO\_THRESHOLD = 0.3

#Minimum consecutive frames for which eye ratio is below threshold for alarm to be triggered

EYE\_ASPECT\_RATIO\_CONSEC\_FRAMES = 50

#COunts no. of consecutuve frames below threshold value

COUNTER = 0

#Load face cascade which will be used to draw a rectangle around detected faces.

face\_cascade = cv2.CascadeClassifier("haarcascades/haarcascade\_frontalface\_default.xml")

#This function calculates and return eye aspect ratio

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

#Load face detector and predictor, uses dlib shape predictor file

detector = dlib.get\_frontal\_face\_detector()

predictor = dlib.shape\_predictor('shape\_predictor\_68\_face\_landmarks.dat')

#Extract indexes of facial landmarks for the left and right eye

(lStart, lEnd) = face\_utils.FACIAL\_LANDMARKS\_IDXS['left\_eye']

(rStart, rEnd) = face\_utils.FACIAL\_LANDMARKS\_IDXS['right\_eye']

#Start webcam video capture

video\_capture = cv2.VideoCapture(0)

#Give some time for camera to initialize(not required)

time.sleep(2)

while(True):

#Read each frame and flip it, and convert to grayscale

ret, frame = video\_capture.read()

frame = cv2.flip(frame,1)

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

#Detect facial points through detector function

faces = detector(gray, 0)

#Detect faces through haarcascade\_frontalface\_default.xml

face\_rectangle = face\_cascade.detectMultiScale(gray, 1.3, 5)

#Draw rectangle around each face detected

for (x,y,w,h) in face\_rectangle:

cv2.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)

#Detect facial points

for face in faces:

shape = predictor(gray, face)

shape = face\_utils.shape\_to\_np(shape)

#Get array of coordinates of leftEye and rightEye

leftEye = shape[lStart:lEnd]

rightEye = shape[rStart:rEnd]

#Calculate aspect ratio of both eyes

leftEyeAspectRatio = eye\_aspect\_ratio(leftEye)

rightEyeAspectRatio = eye\_aspect\_ratio(rightEye)

eyeAspectRatio = (leftEyeAspectRatio + rightEyeAspectRatio) / 2

#Use hull to remove convex contour discrepencies and draw eye shape around eyes

leftEyeHull = cv2.convexHull(leftEye)

rightEyeHull = cv2.convexHull(rightEye)

cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)

cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)

#Detect if eye aspect ratio is less than threshold

if(eyeAspectRatio < EYE\_ASPECT\_RATIO\_THRESHOLD):

COUNTER += 1

#If no. of frames is greater than threshold frames,

if COUNTER >= EYE\_ASPECT\_RATIO\_CONSEC\_FRAMES:

pygame.mixer.music.play(-1)

cv2.putText(frame, "You are Drowsy", (150,200), cv2.FONT\_HERSHEY\_SIMPLEX, 1.5, (0,0,255), 2)

else:

pygame.mixer.music.stop()

COUNTER = 0

#Show video feed

cv2.imshow('Video', frame)

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

break

#Finally when video capture is over, release the video capture and destroyAllWindows

video\_capture.release()

cv2.destroyAllWindows()

**4.RESULTS AND DISCUSSION**

By using Haar Features, we extract the region of eye and face. It detects frame in grey scale format for better accuracy and send to model to detect drowsiness. Threshold Time for drowsiness in project is 0.4 for model training and score to recognize eye drowsiness is 15. If the features from eyes, face indicates that he’s drowsy it gives an immediate alarm sound.

The whole model has been facilitated and the eyes state has been detected with glasses and without glasses to ensure the model's efficacy. The model got an accuracy of 92% over 84,000 instances, giving a liable and effective stratagem for inhibiting accidents due to drowsiness. The Buzzer beams when the score has been increased over 15, the eye closure movement has been detected, After training the model, The model will be verified through test data up to certain number of epoch as follows

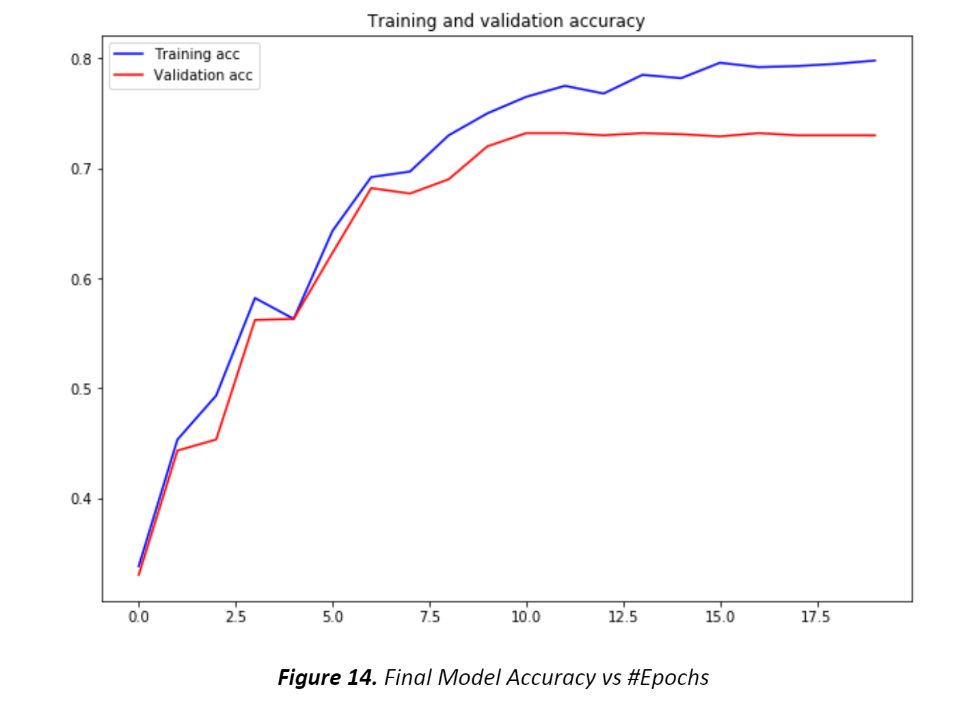


Fig 7: Training, Validation accuracy vs number of epoch

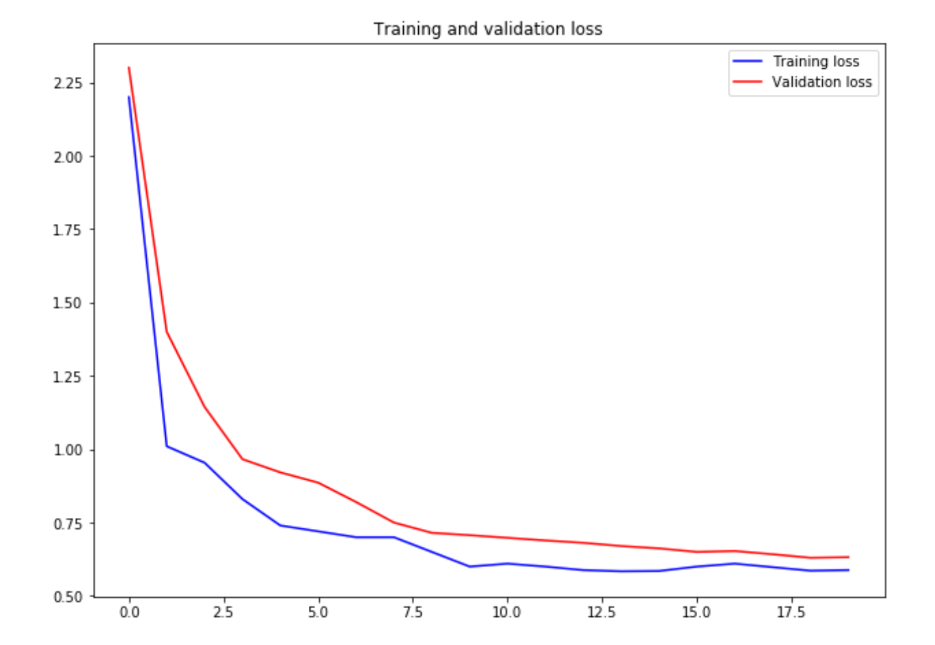


Fig 8: Training, Validation loss vs number of epoch

**5.CONCLUSION**

The model proposed offers a viable solution for Driver’s Drowsiness Detection and thereby prohibiting accidents. The other methods are effective however whilst applied in real-time can affect output for large datasets which includes HRV, EOG, PPG, and so forth. Thus, a light-weight embedded machine may be idea of as a genuine pathway for successful integration of this technique i.e., Driver’s Drowsiness Detection with an alarm over motors. Increased rate of fatality can be avoided and a veritable system can be accessible by means of refuting pointless approaches. Different Deep Learning models can be used and the variety of Classes can be expanded to make the system greater correct and specific the use of Transfer Learning.

**6.REFERENCES**

* [1] 1-Hartman, K. and J. Strasser, Saving Lives Through Advanced Vehicle Safety Technology: Intelligent Vehicle Initiative Final Report. 2005, Department of Transportation: Washington, DC.
* [2] A. Liu, Z. Li, L. Wang and Y. Zhao, "A practical driver fatigue detection algorithm based on eye state," *2010 Asia Pacific Conference on Postgraduate Research in Microelectronics and Electronics (Prime Asia)*, Shanghai, 2010, pp. 235-238.
* [3] Zhong, G., Ying, R., Wang, H., Siddiqui, A., & Choudhary, G., Drowsiness Detection with Machine Learning. [Online]. Available: <https://towardsdatascience.com/drowsiness-detection-with-machine-learning-765a16ca208a> (Date of Access 02 / 06 /2020)
* [4] Computer Vision Lab, National Tsuing Hua University, Driver Drowsiness Detection Dataset. [Online]. Available: <http://cv.cs.nthu.edu.tw/php/callforpaper/datasets/DDD/> (Date of Access 20 / 04 /2020)
* [5] V. Kazumi and J. Sullivan, “One millisecond face alignment with an ensemble of regression trees”, *2014 IEEE Conference on Computer Vision and Pattern Recognition* 1867-1874, 2014.

**BIOGRAPHIES OF AUTHORS (10 PT)**

|  |  |
| --- | --- |
|  | **Borra Umesh Chandra**     is a student of B. Tech Electrical and Computer Engineering (Ongoing – 7th Sem - batch 2024) at SRM INSTITUTE OF SCIENCE AND TECHNOLOGY, KATTANGULATHUR. |