A

Project Report

On

**“DRIVER DROWSINESS DETECTION”**



**Prepared by**

18DCS007

RUDRA BARAD

**Under the Supervision of**

Prof. Krishna Patel

Assistant Professor cum Research Fellow

**A Report Submitted to**

Devang Patel Institute of Advance Technology and Research

Charotar University of Science And Technology (CHARUSAT)

Bachelor of Technology (B.Tech) in

Computer Science & Engineering (CSE)

for Partial Fulfillment of the Requirements for the

7th Semester Software Group Project-V (CS445)





**Charotar University of Science and Technology**

**At: Changa, Dist: Anand – 388421**

**November, 2021**

****

**CERTIFICATE**

This is to certify that the report entitled “**Driver Drowsiness Detection System**” is a bonafied work carried out by **Mr. RUDRA BARAD (18DCS007)** under the guidance and supervision of **Assistant Prof. Krishna Patel** for the subject **CS445-Software Group Project-V** (CSE) of 7th Semester of Bachelor of Technology in **DEPSTAR** at Faculty of Technology & Engineering – CHARUSAT, Gujarat.

To the best of my knowledge and belief, this work embodies the work of candidate himself, has duly been completed, and fulfills the requirement of the ordinance relating to the B.Tech. Degree of the University and is up to the standard in respect of content, presentation and language for being referred to the examiner.

|  |
| --- |
| Ms. Krishna Patel  (Asst. Professor cum Research Fellow)  Department of Computer Science & Engineering  DEPSTAR, CHARUSAT, Changa, Gujarat |
| Dr. Amit Ganatra  Principal, DEPSTAR  Dean, FTE  CHARUSAT, Changa, Gujarat.  Prof. Parth Goel  I/C Head of Department  Department of Computer Science & Engineering DEPSTAR, CHARUSAT, Changa, Gujarat | |

**Devang Patel Institute of Advance Technology and Research At: Changa, Ta. Petlad, Dist. Anand, PIN: 388 421. Gujarat**

# Abstract

In the 21st century we are witnessing the highest growth in vehicle sales and vehicle traffic on the road. With such high traffic comes the fear of accidents taking place. Generally, the main cause for an accident taking place is the carelessness of either driver involved in the accident. Sleep deprived drivers are also known to be one of the most frequent causes in the accidents being caused. As much as 30% of the road accidents are known to have been caused by drowsy drivers. Driver drowsiness is a serious problem and needs to be addressed. The Driver drowsiness detection systems play a very significant role in achieving the target. Even one life saved can make a huge difference in someone’s life.

Drowsiness detection is a safety technology that can prevent accidents that are caused by drivers who fell asleep while driving.

The objective of this intermediate Python 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.

# Acknowledgement

I, the developer of the project “Driver Drowsiness Detection”, with immense pleasure and commitment would like to present the project assignment. The development of this project has given me wide opportunity to think, implement and interact with various aspects of management skills as well as the new emerging technologies.

Every work that one completes successfully stands on the constant encouragement, good will and support of the people around. I hereby avail this opportunity to express my gratitude to number of people who extended their valuable time, full support and cooperation in developing the project.

I express deep sense of gratitude towards our Head of the CSE Department, Prof. Parth Goel and project guide Prof. Krishna Patel for the support during the whole session of study and development. It is because of them, that I was prompted to do hard work, adopting new technologies.

I would also like to thank all the mentor for their guidelines throughout the development phase of the project. They encouraged me to look forward to learn and implement new and emerging technologies. They also guided me to go for some user friendly and extremely useful real-life application.

They altogether provided me favorable environment, and without them it would not have been possible to achieve my goal.

# Table of Contents

[Abstract i](#_Toc87539118)

[Acknowledgement ii](#_Toc87539119)

[Table of Contents iii](#_Toc87539120)

[List of Figures iv](#_Toc87539121)

[CHAPTER 1: PROJECT DEFINITION 5](#_Toc87539122)

[1.1 Project Summary 6](#_Toc87539123)

[1.2 Purpose 6](#_Toc87539124)

[1.3 Scope 6](#_Toc87539125)

[1.4 Objective 6](#_Toc87539126)

[CHAPTER 2: PROJECT DESCRIPTION 5](#_Toc87539127)

[2.1 Existing Systems 6](#_Toc87539128)

[2.2 Proposed System 6](#_Toc87539129)

[2.3 Project Planning 6](#_Toc87539130)

[2.3.1 Project Effort and Time 6](#_Toc87539131)

[2.3.2 Roles and Responsibilities 6](#_Toc87539132)

[2.4 The Dataset 7](#_Toc87539133)

[2.5 The Model Architecture 7](#_Toc87539134)

[CHAPTER 3: SYSTEMS REQUIREMENTS 9](#_Toc87539135)

[3.1 User Characteristics 10](#_Toc87539136)

[3.2 Hardware Requirements 10](#_Toc87539137)

[3.3 Software Requirements 10](#_Toc87539138)

[CHAPTER 4: MAJOR FUNCTIONALITY 11](#_Toc87539139)

[CHAPTER 5: SYSTEM FLOWCHART 14](#_Toc87539140)

[CHAPTER 6: SCREENSHOTS OF THE PROJECT 17](#_Toc87539141)

[CHAPTER 7: LIMITATIONS 20](#_Toc87539142)

[CHAPTER 8: FUTURE ENHANCEMENT 22](#_Toc87539143)

[CHAPTER 9: CONCLUSION AND DISCUSSION 24](#_Toc87539144)

[CHAPTER 10: REFERENCES 26](#_Toc87539145)

# List of Figures

[Fig 2.1: CNN Architecture 7](#_Toc87540630)

[Fig 2.2: Transfer Learning Architecture 8](#_Toc87540631)

[Fig 5.1: Logical Flow Chart 15](#_Toc87540632)

[Fig 5.2: Process Flow Chart 16](#_Toc87540633)

[Fig 6.1: Open Eyes 18](#_Toc87540634)

[Fig 6.2: Closed Eyes 19](#_Toc87540635)

# CHAPTER 1: PROJECT DEFINITION

## 1.1 Project Summary

Our project is on Driver Drowsiness Detection System.

This Project has been made for user protection. This ML model is well organized with proper dataset and accuracy.

For developing a ML modal, we use PyCharm. It is an open source software. For image we have used OpenCV

* You’ll never get Sleepy at driving time!
* Sound starts if person found Drowsy.
* Help to avoid accident.

## 1.2 Purpose

The main purpose for making this modal was, as we see now a day about 30% of accident are happening due to sleepily or drowsiness of driver while driving. By this we are think that we can help to reduce the people’s life.

## 1.3 Scope

The idea of this project can be a great idea for startup for us. We will share this project to as many people as we can.

## 1.4 Objective

Automotive population is increasing exponentially in our country. The most important drawback concerning the inflated use of vehicles is the rising number of road accidents. Road accidents are doubtlessly a global peril in our country. The frequency of road accidents in India is among the highest in the world according to a report by the transport research wing of the ministry of road transport and highways, about a total of 151,113 population got killed in 480,652 road accidents across India in 2019, an average of 414 a day or 17 an hour. For the same, a total of 4,37,396 road accidents got recorded resulting in the death of 1,54,732 people and injuries to another 4,39,262, according to the latest National Crime Records Bureau (NCRB) data of 2019 in India. Approximately, 1.35 million of the public go to deathbed each year as an outcome of road traffic crashes worldwide as per reports.

ITS aims to achieve traffic efficiency by minimizing traffic related problems not to mention the driver assistance systems that are included like Adaptive Cruise Control, Park Assistance Systems, Intelligent Headlights, Pedestrian Detection Systems, Blind Spot Detection Systems, etc.

# 

# CHAPTER 2: PROJECT DESCRIPTION

## 2.1 Existing Systems

There are 4 types of systems available on Driver Drowsiness Detection: -

1. Physiological analysis, using body sensors to detect brain activity, heart rate etc.
2. Lane motion sensing system, using lane monitoring cameras.
3. Detection systems based on Facial Feature Extraction (FFE).
4. Steering angle sensor analysis-based systems.

## 2.2 Proposed System

Our proposed system is entitled as Driver Drowsiness Detection system. It is designed specifically to avoid accidents, the majority of which is caused because the driver is feeling drowsy.

The system will work on simple principle of counter, wherein a limit (say 15) is set above which if the driver is found sleepy (or his/her eyes are closed), then audio/ alarm will be played.The counter will increase based on the time for which the driver’s eyes are shut. As a result of which driver will be alerted and can take control of the vehicle.

## 2.3 Project Planning

### 2.3.1 Project Effort and Time

The programmers have given about 1-2 hours per day to make this project.

### 2.3.2 Roles and Responsibilities

Team Members have worked effectively and have undertaken the following tasks:

* + - * Researching in the related field of work
      * Creating a step to follow
      * Connecting to OpenCV
      * Detected the driver face
      * Detected the eye on face
      * Run modal to classify the score whether open or close
      * Finally calculates percentage of Drowsiness

## 2.4 The Dataset

The Dataset used for the CNN model training system is [yawn\_eye\_dataset](https://www.kaggle.com/serenaraju/yawn-eye-dataset-new). It is a free and open-source dataset available on kaggle. The entire dataset has been divided into 2 parts which are used separately for training (75% of the dataset) and testing (25% of the dataset). Each part has 4 different feature values that will be taken into consideration while evaluating. For the transfer Learning method, we have used the [mrl\_eyes\_dataset](http://mrl.cs.vsb.cz/eyedataset). The mrl eye dataset is a giant dataset consisting of only eye characteristics and is also divided into 2 parts, training and testing. This dataset will be used to retrain the model on the pre-trained InceptionV3 model.

## 2.5 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.

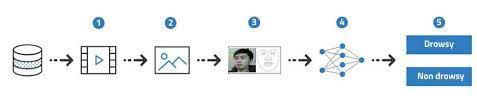


Fig 2.1: CNN Architecture

The second approach to the solution of the Driver Drowsiness detection problem is the transfer learning method. Transfer learning is an efficient method for training models on large datasets. Transfer Learning basically takes a pre-existing model and retrains the model on the dataset provided by the user. The main advantage this technique provides is that the total training time for the model is less. But as we are taking an already trained model on a general dataset tha model lacks the accuracy for the particular data. So, we can say that transfer learning is a tradeoff between model training time and model accuracy.

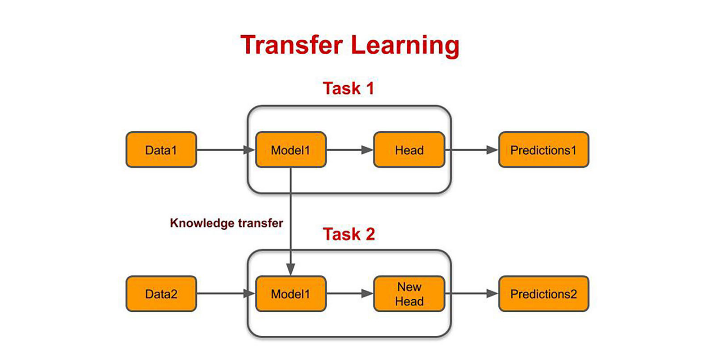


Fig 2.2: Transfer Learning Architecture

# CHAPTER 3: SYSTEMS REQUIREMENTS

## 3.1 User Characteristics

This Model has been made for user protection. ML model is well organized with proper dataset and accuracy.

## 3.2 Hardware Requirements

* Processor: i3/i5/i7 8gen or above
* Processor Speed: 2.16GHZ or above
* RAM: 3 GB RAM or above
* Storage: 20 GB hard disk or above

## 3.3 Software Requirements

* Pycharm
* OpenCV
* TensorFlow

# CHAPTER 4: MAJOR FUNCTIONALITY

When the program starts, we'll start capturing the video of the user using OpenCV. We'll capture all the frames of video and detect whether the eyes are close or open. Let's go in a bit detail.

**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 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.

**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 leye and reye 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 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 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 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. 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().

# CHAPTER 5: SYSTEM FLOWCHART

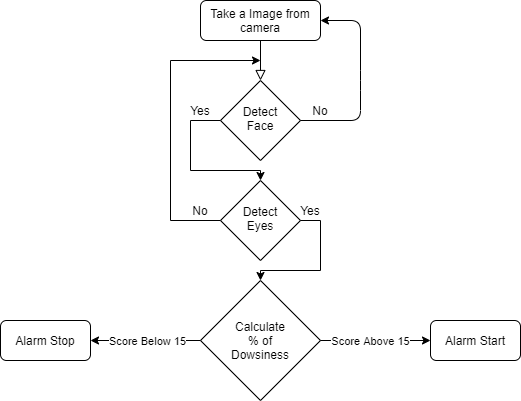
****

Fig 5.1: Logical Flow Chart

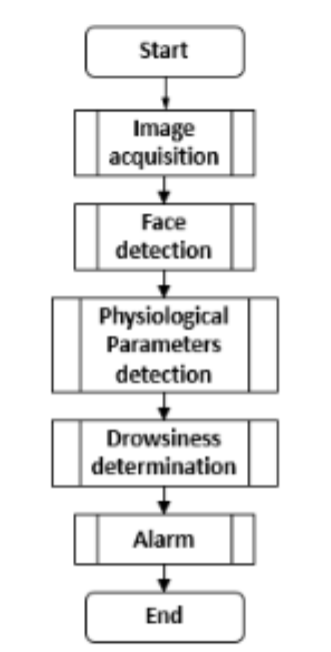
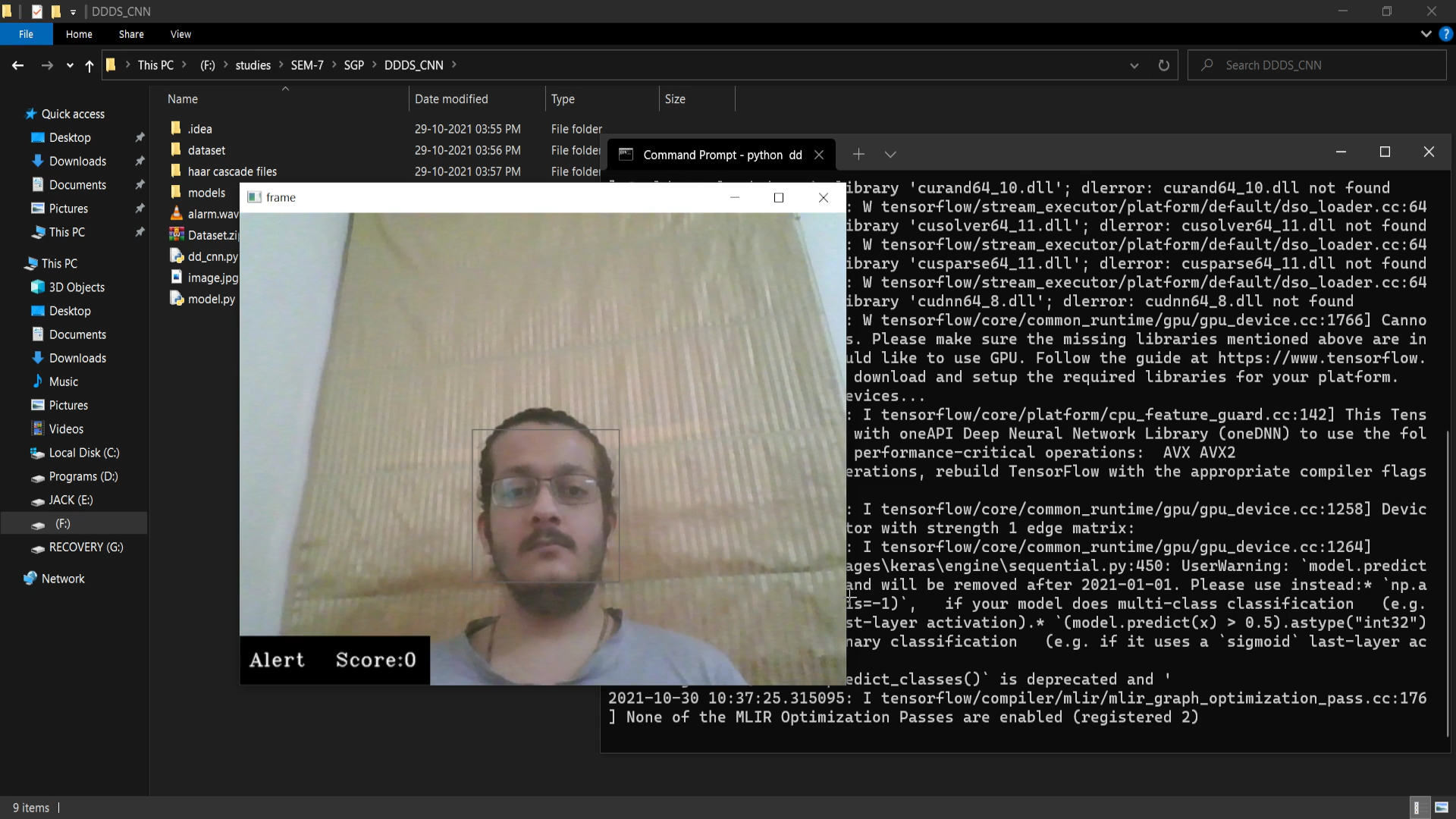


Fig 5.2: Process Flow Chart

# 

# CHAPTER 6: SCREENSHOTS OF THE PROJECT

Driver Drowsiness Detection System will detect when the eyes are open there will be NO red line and alarm sound, it shows driver is awake.



# 

Fig 6.1: Open Eyes

Driver Drowsiness Detection System will detect when the eyes are closed there will be a red line and alarm will start beeping, it will awake driver before a mishap happens.

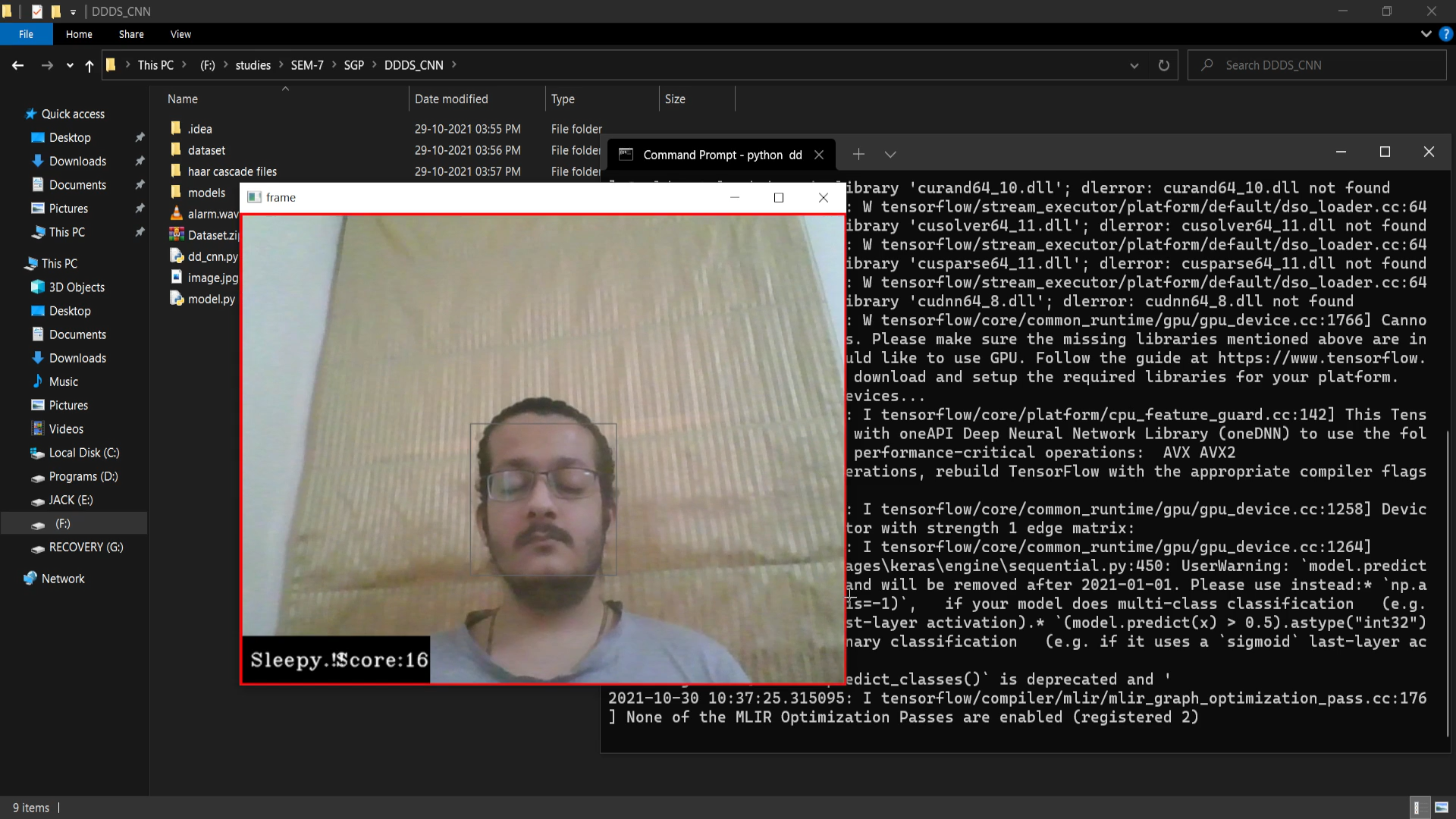


Fig 6.2: Closed Eyes

# 

# CHAPTER 7: LIMITATIONS

Following are the currently persisting limitations in the application:

* Handling vast amount of data (larger dataset) was bit challenging.
* Detection in Low Light environment is not as accurate as with ambient light
* Slight lag is observed in the process as frame wise image is captured and is compared.
* In transfer learning part there is a slight tradeoff between Speed and accuracy

# CHAPTER 8: FUTURE ENHANCEMENT

Following are some aspirations specifying our future enhancements:

* There is always a scope of increasing accuracy to make the system near perfect.
* We can use MPEG encoding protocol to overcome the slight delay problem in the program.
* The program software can be installed in a dedicated hardware setup to improvise the performance.

# CHAPTER 9: CONCLUSION AND DISCUSSION

**Self-Analysis of Project Viabilities**

The project was a good experience for us. We think that from this project we learnt a lot about how a dataset works. We learnt how to develop an ML model with large data set.

**Problem Encountered**

* Learning python was bit challenging.
* Moving from Development to ML.
* Managing Large Dataset with ML model

**Summary of Project Work**

The project was a great experience of working in a team. The importance of time bound and coordinated execution of work was realized. It gave us an experience to develop ML model for detection of sleepy person at time of driving.

The application is a user friendly and can be run by any person with the help of PyCharm in desktop. The preparation of this project has helped a lot to learn the much unknown features of OpenCV, Keras, TensorFlow.

The proposed system was first implemented in web browsers and later in android applications to be easily accessible as these gadgets are basic requirements of people nowadays and can be found with any and everybody. For monitoring drowsiness of the driver’s alertness, Smartphone is taken as a minimum requirement and by extracting facial and eye gestures alongside its movements as input. This gave a high performance, nonetheless can be taken to a higher degree of performance by fusing more data and information into the same. Some examples are weather forecast, mechanical data of vehicles, signing in and monitoring the driver daily at different drives and the list is endless. Hence, here the real time processing is attained utilizing OpenCV and other required packages for an inclined outcome that gives results with flying colors and the boost in expansion of this technology is exceptionally perceived.

# 

# CHAPTER 10: REFERENCES

1. https://www.infoq.com/presentations/understanding-deep-learning/
2. https://www.youtube.com/playlist?list=PL3GCZkoyKK4ehcDlwaVD8fdiS-w2U6nU
3. https://www.youtube.com/playlist?list=PL9ooVrP1hQOFJ8UZl86fYfmB1\_P5yGzBT
4. https://www.youtube.com/playlist?list=PLS1QulWo1RIa7D1O6skqDQ-JZ1GGHKK-K
5. https://www.youtube.com/watch?v=9jA0KjS7V\_c&list=PLZoTAELRMXVPGU70ZGsckrMdr0FteeRUi
6. https://elitedatascience.com/keras-tutorial-deep-learning-in-python
7. https://www.pyimagesearch.com/2018/07/19/opencv-tutorial-a-guide-to-learn-opencv/
8. https://hackr.io/tutorials/learn-opencv
9. https://opencv-python-tutroals.readthedocs.io/en/latest/py\_tutorials/py\_ml/py\_table\_of\_contents\_ml/py\_table\_of\_contents\_ml.html
10. https://www.tensorflow.org/resources/learn-ml/basics-of-machine-learning
11. https://developers.google.com/machine-learning/crash-course/first-steps-with-tensorflow/toolkit
12. https://www.coursera.org/learn/introduction-tensorflow