

A Capstone Project Report on

DRIVER-DROWSINESS DETECTION USING DEEP LEARNING

Submitted by,
Sabana R

Table of content				Page no.
Abstract				4
Acknowledgments				5
Certification of Completion				6
Chapter 1: INTRODUCTION				7
1.1	Purpose	1.1.1	Human Psychology with Current Technology	7
		1.1.2	Facts and Statistics	7
1.2	Document Conventions			8
1.3	Product Scope			8
1.4	Problem Definition			8
Chapter 2: LITERATURE SURVEY				9
2.1	System Review			9
2.2	Technology Used			9
Chapter 3: REQUIREMENTS				10
3.1	Software Requirements			10
3.2	Hardware Requirements			10
Chapter 4: TRANSFER LEARNING				11
4.1	Building Model			11
4.2	Methodology			11
Chapter 5: PROJECT APPROACH				13
5.1	Steps To Follow			13
Chapter 6: MODEL ARCHITECTURE				14
6.1	Convolutional Neural Networks			14
Chapter 7: Project Folders				15
7.1	File Contents			15
Chapter 8: SYSTEM DESIGN				16
8.1	Use Case Diagram			16
8.2	Activity Diagram			17
8.3	Class Diagram			18
Chapter 9: PROJECT PLANNING				19
9.1	System Model			19
Chapter 10: SOURCE CODE				21
10.1	Implementation			21
10.2	Output			27
Chapter 11: CONCLUSION AND FUTURE SCOPE				28
11.1	Conclusion			28
11.2	Future Scope			28
Chapter 12: REFERENCE				29

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO
1	Bright and Drowsy	8
2	Python	10
3	Web cam	10
4	Transfer Learning	11
5	Methodology	12
6	Process Flow	13
7	Architecture of CNN	14
8	Folder Content	15
9	Use case Diagram	16
10	Activity Diagram	17
11	Class Diagram	18
12	Flow chart of model	19
13	Predicted Images	20
14	Opened Eye	27
15	Closed Eye	27

Abstract

Every year thousands of lives pass away worldwide due to vehicle accidents, and the main reason behind this is the drowsiness in drivers. A drowsiness detection system will help to reduce this accident and save many lives around the world. To defend this problem, we propose a methodology based on Convolutional Neural Networks (CNN) that illustrates drowsiness detection as a task to detect an object. It will detect and localize whether the eyes are open or close based on the real-time video stream of drivers. If the driver closes his eyes for more than 5 seconds, the alarm will beep to wake him up. It saves the lives of billion people.

Acknowledgement

A am using this capstone project as an opportunity to express my gratitude to everyone with me throughout the course. I am thankful for their inspiring guidance, leadership quality, and friendly advice for this project. This creates a very comfortable environment to complete the project at time with complete enthusiasm.

Further, I thank my mentor Mr.Anbu Joel. He has readily shared his immense knowledge as a great guide.

I certify that the work done by me for conceptualizing and completing this project is original and authentic.

Date – 31/07/2022

Name – SABANA R

Certificate of Completion

I certify that the project titled “Driver Drowsiness Detection” was undertaken and completed (31/07/2022)

Mentor – Mr.Anbu Joel

Date – 31/07/2022

Place – Trichy

CHAPTER 1

INTRODUCTION

1.1 Purpose

Title – Driver Drowsiness Detection using Deep learning

1.1.1 HUMAN PSYCHOLOGY WITH CURRENT TECHNOLOGY

Humans have always invented machines and devised techniques to ease and protect their lives, for mundane activities like travelling to work, or for more interesting purpose 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 grandparents wouldn't have thought possible. In modern times, almost everyone in this world uses some sort of 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.

1.1.2 FACTS & STATISTICS

Our current statistics reveal that just in 2015 in India alone, 148,707 people died due to car related accidents. Of these, at least 21 percent were caused due to fatigue causing drivers to make mistakes. This can be a relatively smaller number still, as among the multiple causes that can lead to an accident, the involvement of fatigue as a cause is generally grossly underestimated. Fatigue combined with bad infrastructure in developing countries like India is a recipe for disaster. Fatigue, in general, is very difficult to measure or observe unlike alcohol and drugs, which have clear key indicators and tests that are available easily. Probably, the best solutions to this problem are awareness about fatigue-related accidents and promoting drivers to admit fatigue when needed. The former is hard and much more expensive to achieve, and the latter is not possible without the former as driving for long hours is very lucrative. When there is an increased need for a job, the wages associated with it increases leading to more and more people adopting it. Such is the case for driving transport vehicles at night. Money motivates drivers to make unwise decisions like driving all night even with fatigue. This is mainly because the drivers are not themselves aware of the huge risk associated with driving when fatigued. Some countries have imposed restrictions on the number of hours a driver can drive at a stretch, but it is still not enough to solve this problem as its implementation is very difficult and costly.

1.2 DOCUMENT CONVENTIONS

- Main Heading Font size: 12 (bold fonts)
- Sub-headings Font size: 12 (bold fonts)
- Sub-headings Content Font size: 12 (normal fonts)

1.3 PRODUCT SCOPE

There are many products out there that provide the measure of fatigue level in the drivers which are implemented in many vehicles. The driver drowsiness detection system provides the similar functionality but with better results and additional benefits. Also, it alerts the user on reaching a certain saturation point of the drowsiness measure.

1.4 PROBLEM DEFINITION

Fatigue is a safety problem that has not yet been deeply tackled by any country in the world mainly because of its nature. Fatigue, in general, is very difficult to measure or observe unlike alcohol and drugs, which have clear key indicators and tests that are available easily. Probably, the best solutions to this problem are awareness about fatigue-related accidents and promoting drivers to admit fatigue when needed. The former is hard and much more expensive to achieve, and the latter is not possible without the former as driving for long hours is very lucrative



Figure 1: Bright and drowsy

CHAPTER 2

LITERATURE SURVEY

2.1 SYSTEM REVIEW

This survey is done to comprehend the need and prerequisite of the general population, and to do as such, we went through different sites and applications and looked for the fundamental data. Based on these data, we made an audit that helped us get new thoughts and make different arrangements for our task. We reached the decision that there is a need of such application and felt that there is a decent extent of progress in this field too.

2.2 TECHNOLOGY USED

a. PYTHON - Python is an interpreted, high-level, general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed AND supports multiple programming paradigms, including procedural, object-oriented, and functional programming.

b. JUPYTER Lab - Project jupyter is a non-profit organization created to develop open-source software, open-standards, and services for interactive computing across dozens of programming languages.

c. IMAGE PROCESSING - In computer science, digital image processing is the use of computer algorithms to perform image processing on digital images.

d. DEEP LEARNING -Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example. Deep learning is a key technology behind driverless cars, enabling them to recognize a stop sign, or to distinguish a pedestrian from a lamppost. It is the key to voice control in consumer devices like phones, tablets, TVs, and hands-free speakers. Deep learning is getting lots of attention lately and for good reason. It's achieving results that were not possible before.

CHAPTER 3

REQUIREMENTS

3.1 SOFTWARE REQUIREMENTS

- **Python**
- **OpenCV:** OpenCV is a great tool for image processing and performing many computer vision tasks. It is an open-source library that can be used to perform tasks like face detection, object tracking, and many more tasks.
- **TensorFlow:** Tensorflow is a free and open-source library, developed by the Google Brain team for machine learning and artificial intelligence. Tensorflow has a particular focus on the training and inference of deep neural networks.
- **Keras:** Keras is an open-source software library and it provides a Python interface for artificial neural networks. Keras is more user-friendly because it is an inbuilt python library.

3.2 HARDWARE REQUIREMENTS

- Laptop with basic hardware.
- Webcam



Figure 2: Python



Figure 3: Web cam

CHAPTER 4

TRANSFER LEARNING

4.1 BUILDING MODEL

In this project, we will use transfer learning to build the model. Transfer Learning is a machine learning method where we use a pre-trained model for a new model with the related problem statement.

For example, a model that is used for the recognition of cars can be used for the recognition of trucks.

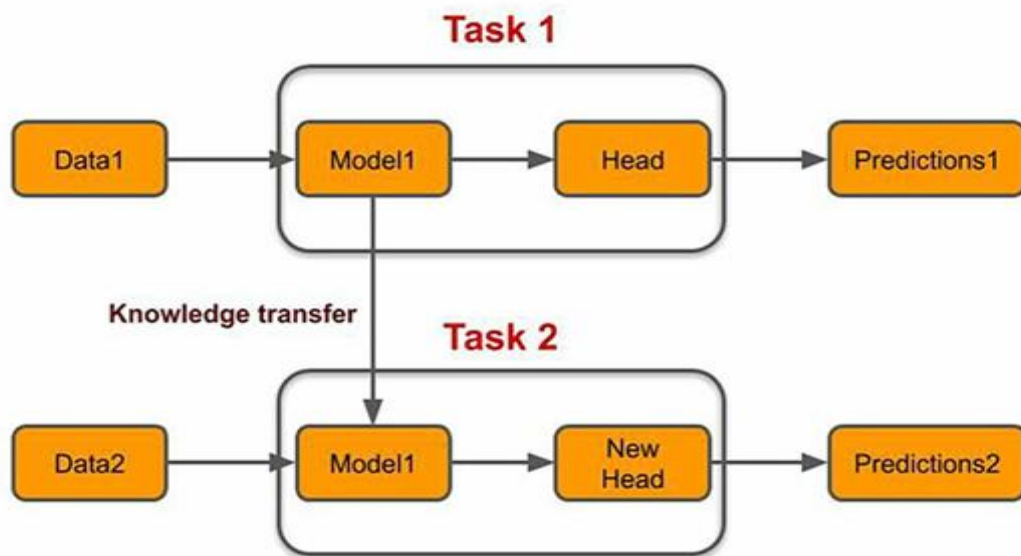


Figure 4: Transfer Learning

Here mainly it focuses on the knowledge that is gained while solving one problem and it applies to a different but related problem.

4.2 METHODOLOGY

The methodology of this project is the first video is captured using a webcam and from the video first face is detected using the Harcascade algorithm and then the eyes are detected. Then we use our deep learning model which is built using transfer learning to know the status of the eye. If it is an open eye then it will say Active and if it is a closed eye then it will check for a few seconds and then it will say the driver is drowsy and will beep an alarm.

We will use Python, OpenCV, TensorFlow, and Keras to build a system that can detect the closed eyes of drivers and alert them if ever they fall asleep while driving. If the driver's eyes are closed, this system will immediately inform the driver. OpenCV that we are going to use now will monitor and collect the driver's images via a webcam that was attached and feed them into the deep learning model and then the model will classify the driver's eyes as 'open' or 'closed.'

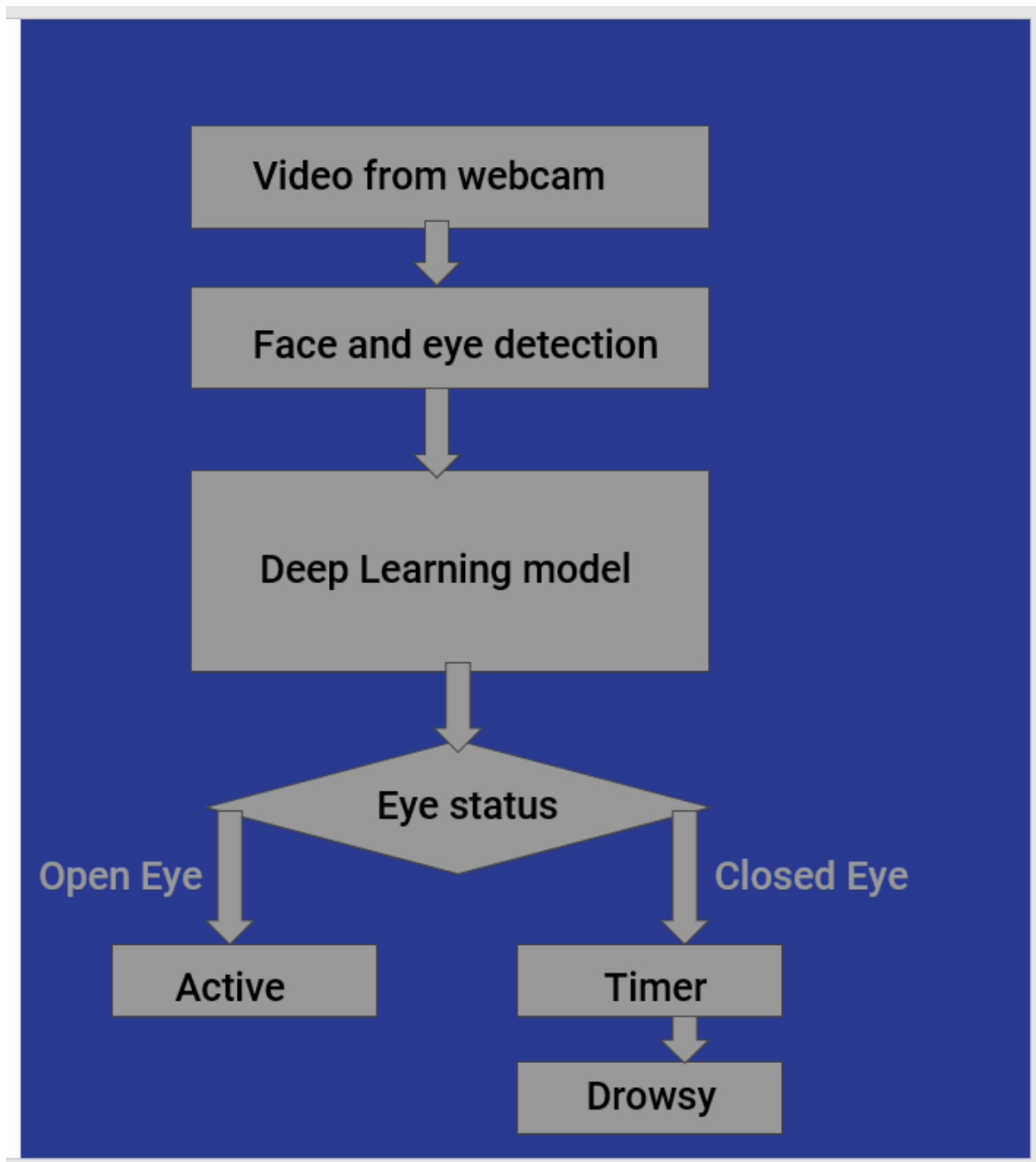


Figure 5: Methodology

CHAPTER 5

PROJECT APPROACH

5.1 STEPS TO FOLLOW

In this Python project, we will be using OpenCV for gathering the images from webcam and feed them into a deep learning model 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.

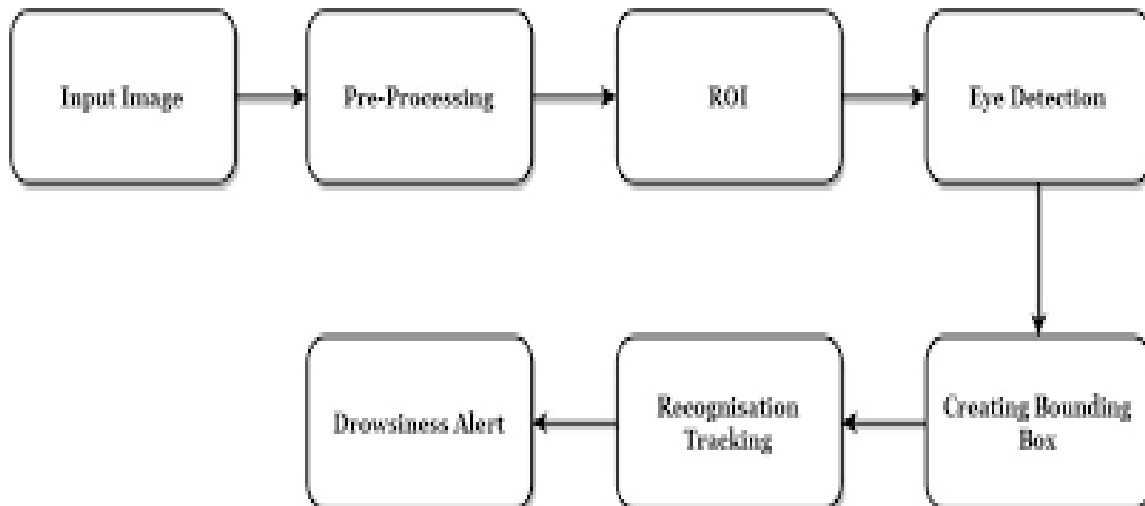


Figure 6: Process Flow

CHAPTER 6

MODEL ARCHITECTURE

6.1 CONVOLUTIONAL NEURAL NETWORKS

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 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. A Relu activation function is used in all the layers except the output layer in which we used Softmax.

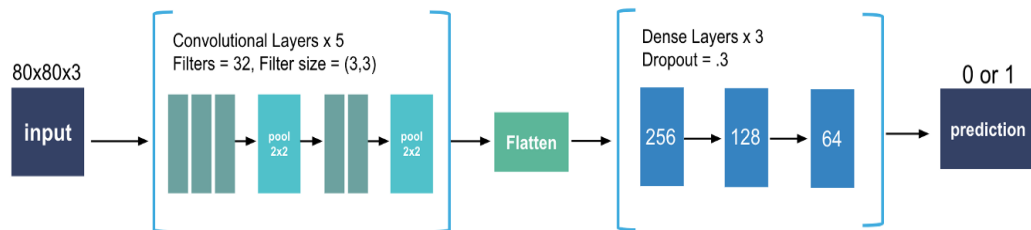


Figure 7: Architecture of CNN

CHAPTER 7

PROJECT FOLDERS

7.1 FILE CONTENTS

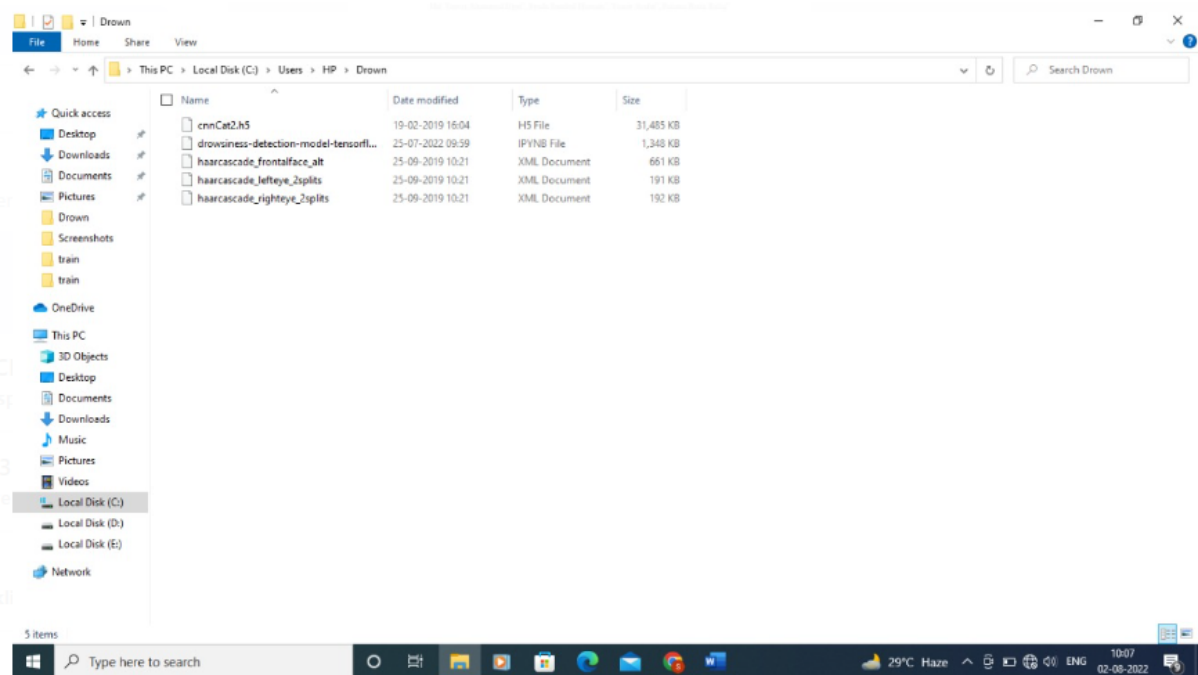


Figure 8: Folder Content

- 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 “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.
- “Drowsiness detection.py” is the main file of our project. To start the detection procedure, we have to run this file.

CHAPTER 8

SYSTEM DESIGN

8.1 USE CASE DIAGRAM

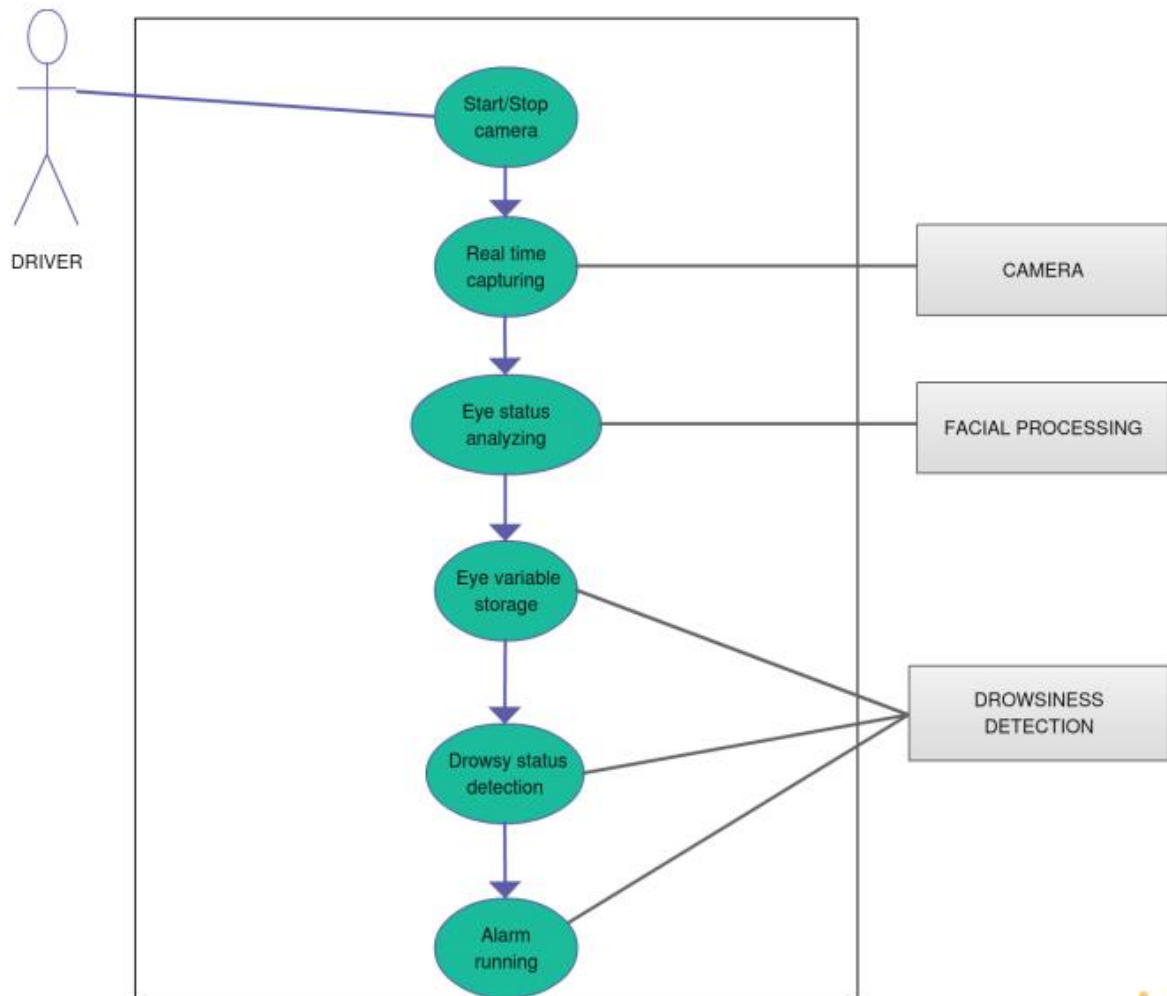


Figure 9: Use case Diagram

8.2 ACTIVITY DIAGRAM

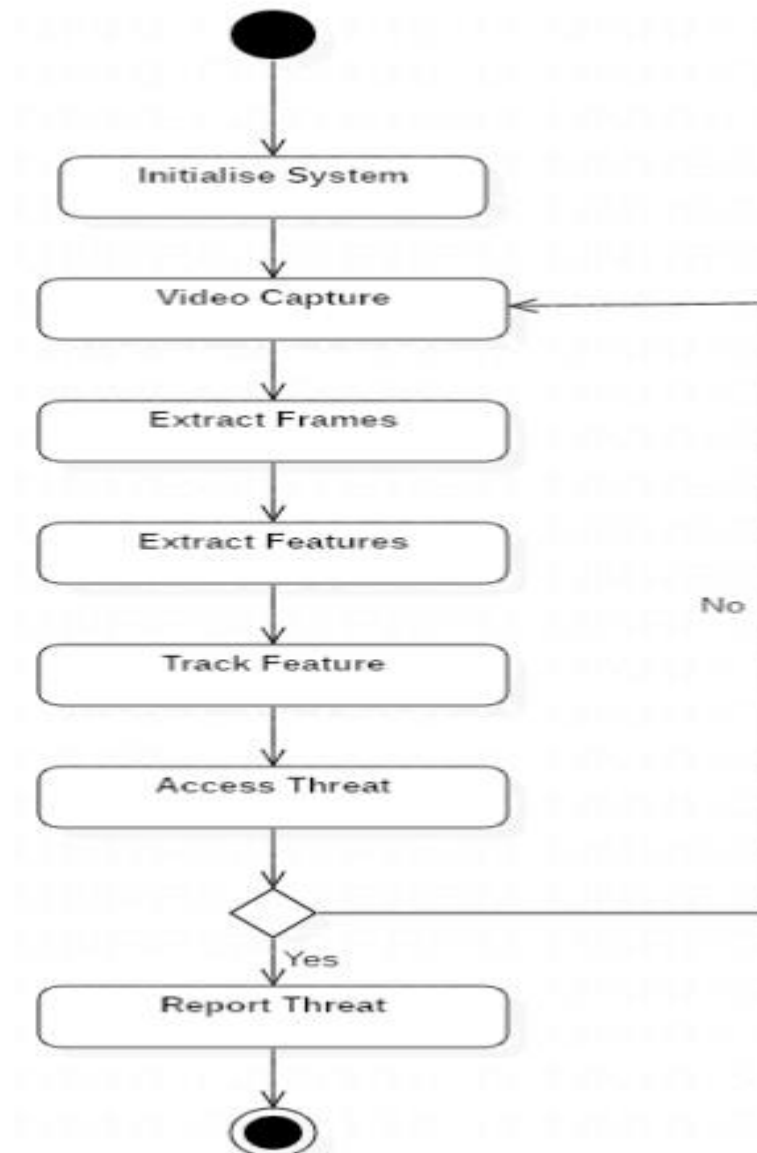


Figure 10: Activity Diagram

8.3 CLASS DIAGRAM

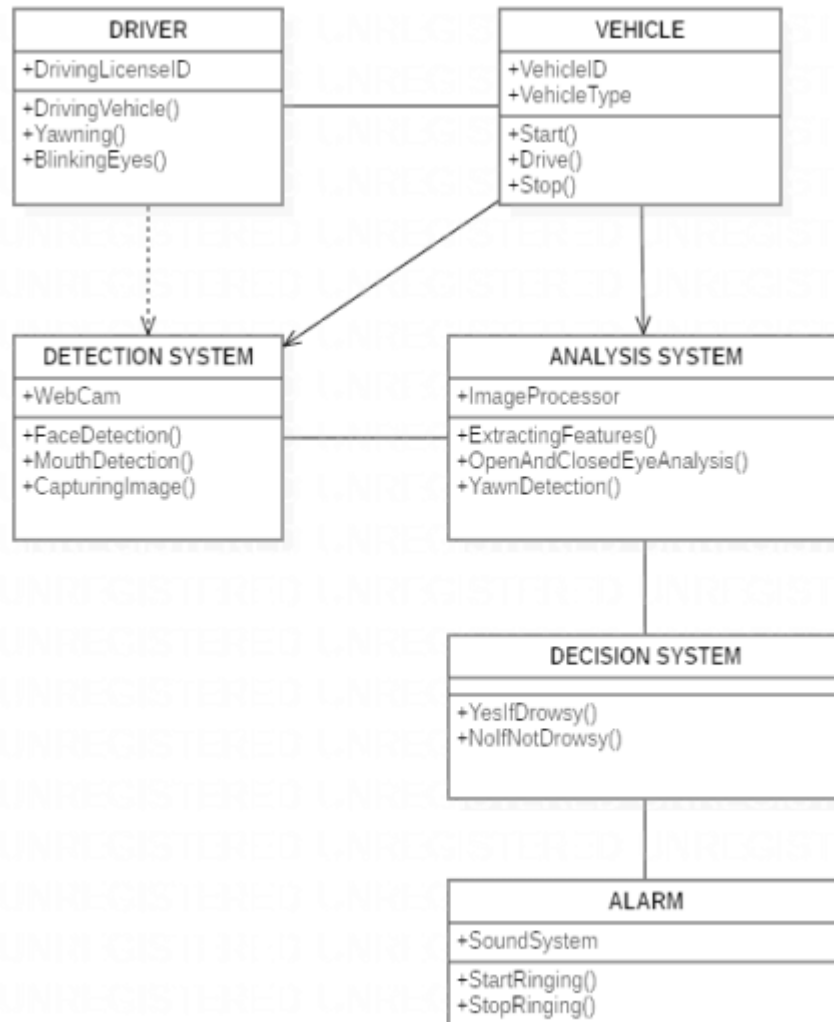


Figure 11: Class Diagram

CHAPTER 9

PROJECT PLANNING

9.1 SYSTEM MODEL

The framework is created utilizing the incremental model. The centre model of the framework is first created and afterwards augmented in this way in the wake of testing at each turn. The underlying undertaking skeleton was refined into expanding levels of ability. At the following incremental level, it might incorporate new execution backing and improvement.

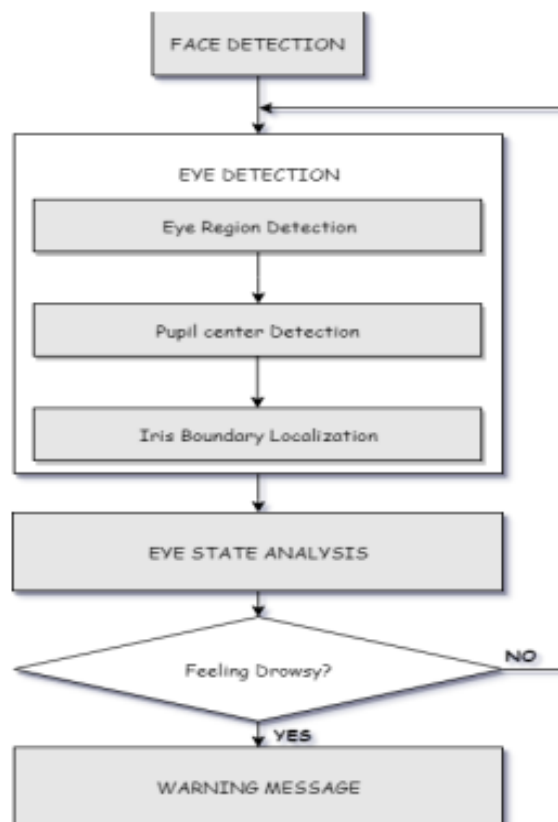


Figure 12: Flow chart of model

9.2 PREDICTION

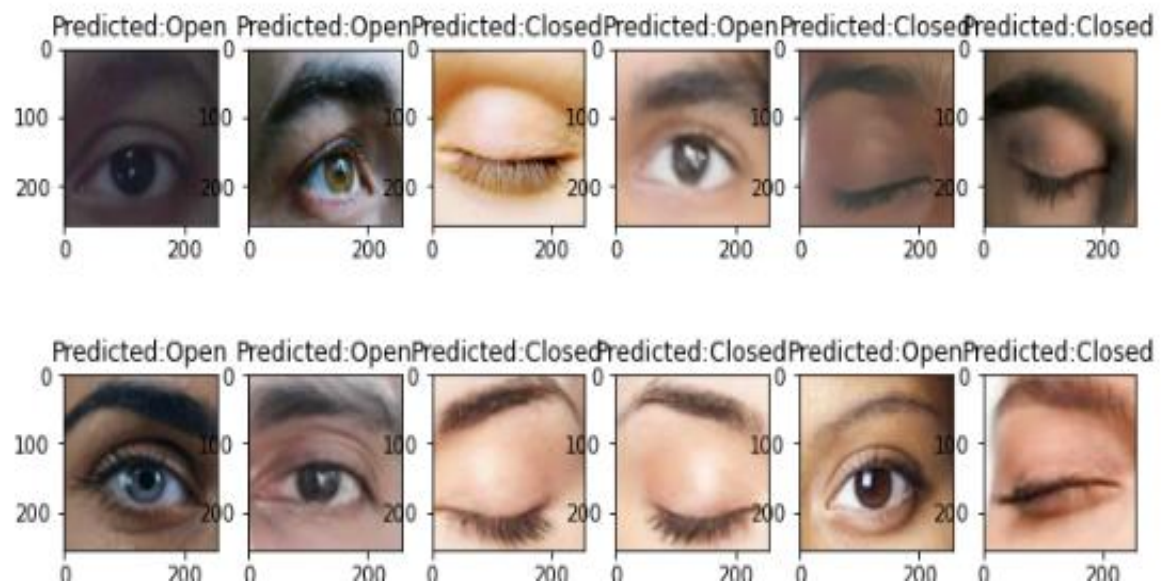


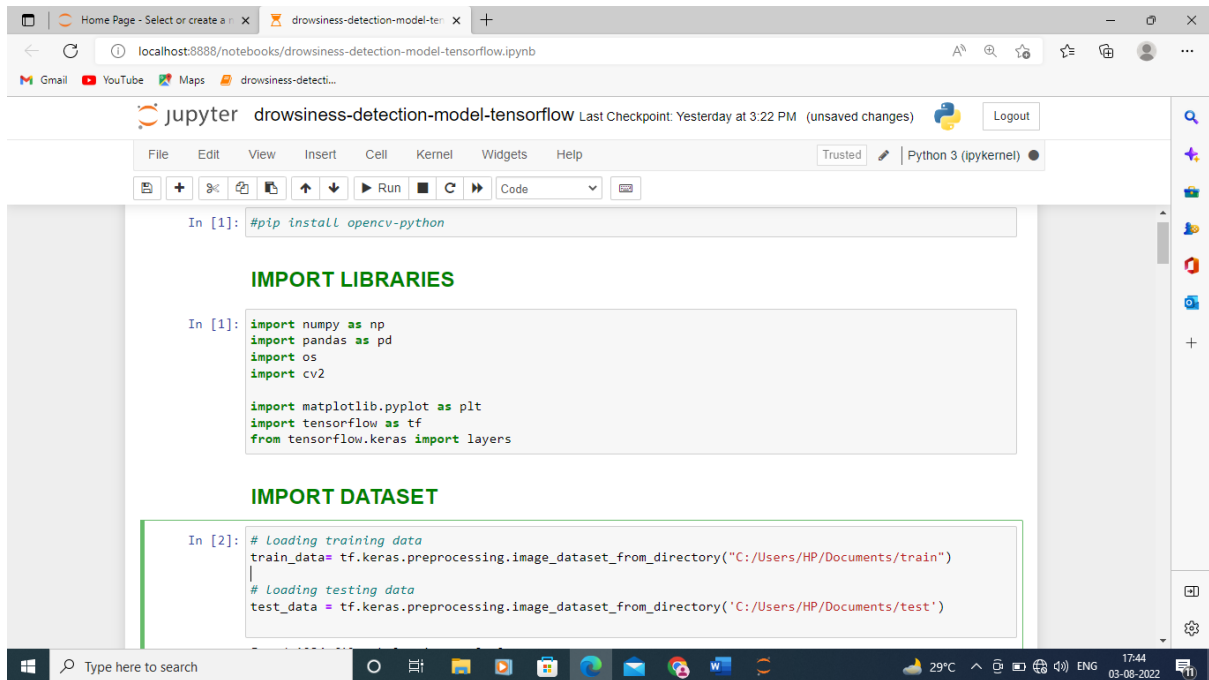
Figure 13: Predicted Images

If the eyes closed, the alarm will start to beep. So that, we can save the driver life.

CHAPTER 10

SOURCE CODE

10.1 IMPLEMENTATION



```
In [1]: #pip install opencv-python

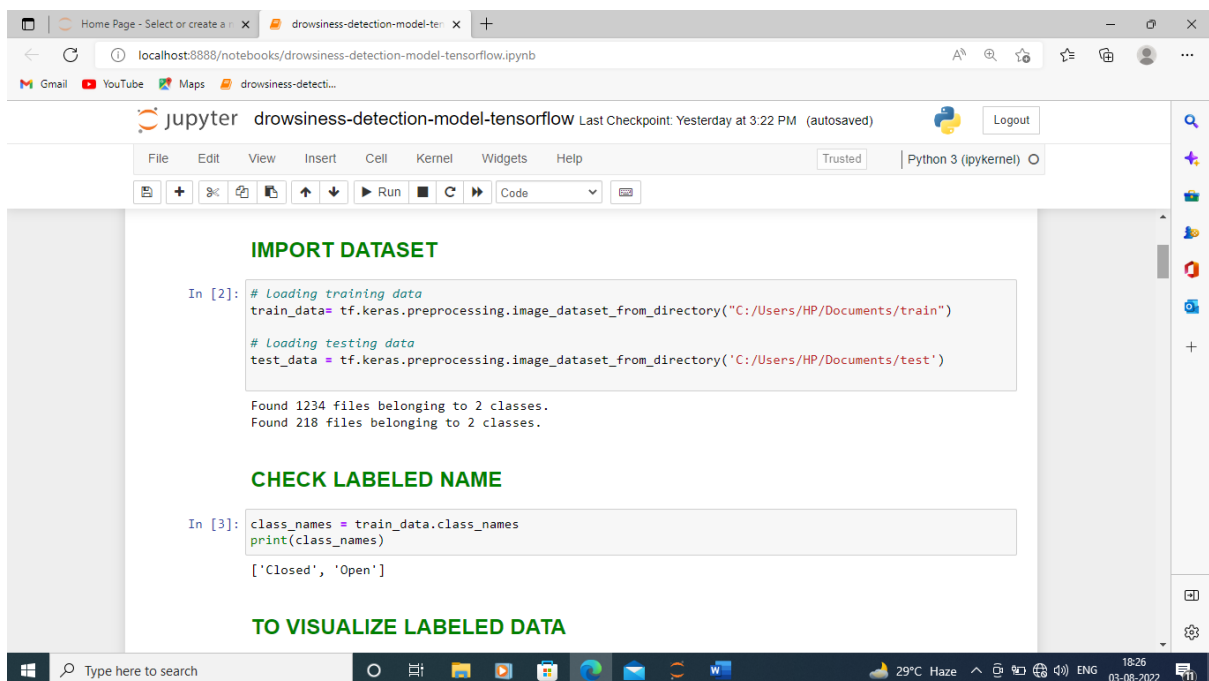
IMPORT LIBRARIES

In [1]: import numpy as np
import pandas as pd
import os
import cv2

import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras import layers

IMPORT DATASET

In [2]: # Loading training data
train_data = tf.keras.preprocessing.image_dataset_from_directory("C:/Users/HP/Documents/train")
# Loading testing data
test_data = tf.keras.preprocessing.image_dataset_from_directory('C:/Users/HP/Documents/test')
```



```
IMPORT DATASET

In [2]: # Loading training data
train_data = tf.keras.preprocessing.image_dataset_from_directory("C:/Users/HP/Documents/train")
# Loading testing data
test_data = tf.keras.preprocessing.image_dataset_from_directory('C:/Users/HP/Documents/test')

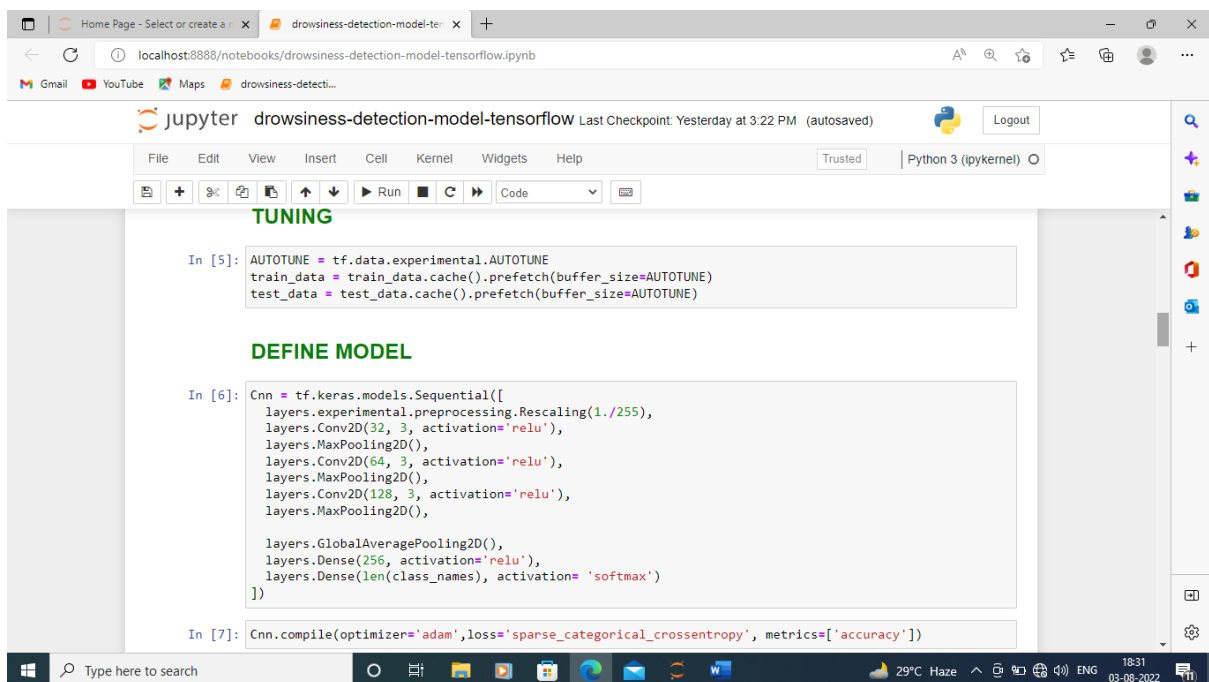
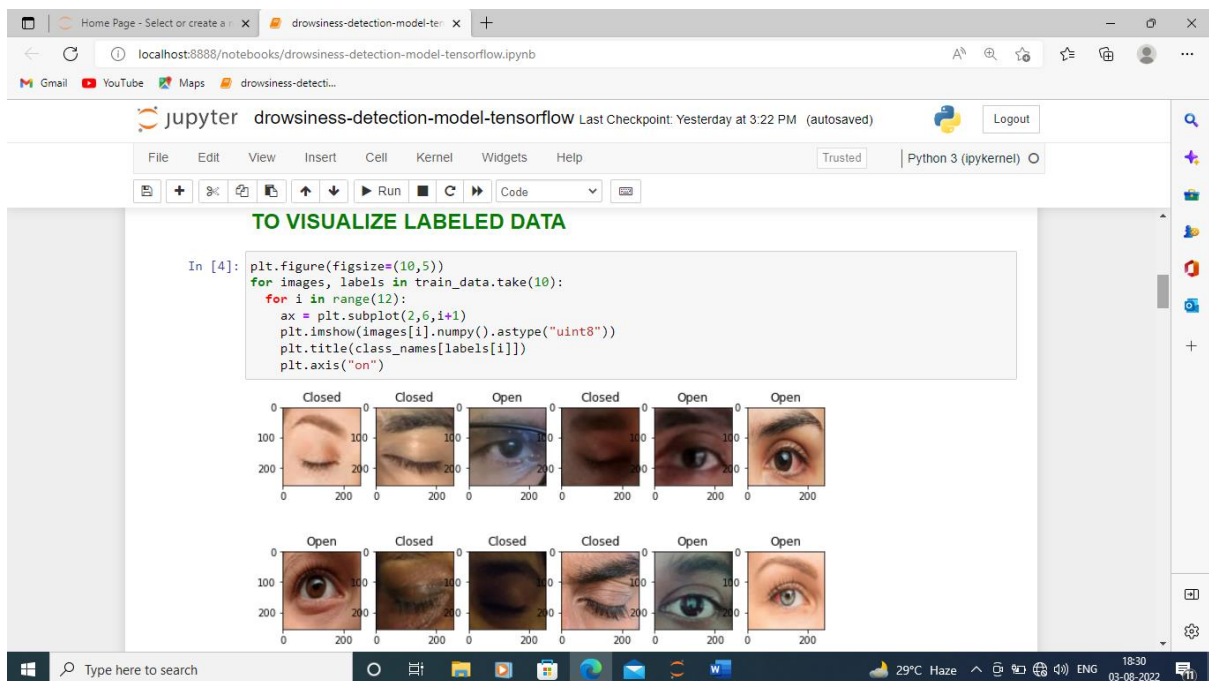
Found 1234 files belonging to 2 classes.
Found 218 files belonging to 2 classes.

CHECK LABELED NAME

In [3]: class_names = train_data.class_names
print(class_names)

['Closed', 'Open']

TO VISUALIZE LABELED DATA
```



Home Page - Select or create a notebook x drowsiness-detection-model-tensorflow x +

localhost:8888/notebooks/drowsiness-detection-model-tensorflow.ipynb

jupyter drowsiness-detection-model-tensorflow Last Checkpoint: Yesterday at 3:22 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

TRAIN CNN

```
In [8]: Value = Cnn.fit(train_data, validation_data= test_data, epochs = 15)
```

```
Epoch 1/15
39/39 [=====] - 350s 8s/step - loss: 0.6922 - accuracy: 0.5340 - val_loss:
0.6836 - val_accuracy: 0.5734
Epoch 2/15
39/39 [=====] - 192s 5s/step - loss: 0.6856 - accuracy: 0.5948 - val_loss:
0.6674 - val_accuracy: 0.5963
Epoch 3/15
39/39 [=====] - 137s 4s/step - loss: 0.6698 - accuracy: 0.6175 - val_loss:
0.6560 - val_accuracy: 0.6055
Epoch 4/15
39/39 [=====] - 133s 3s/step - loss: 0.6504 - accuracy: 0.6191 - val_loss:
0.6122 - val_accuracy: 0.6789
Epoch 5/15
39/39 [=====] - 133s 3s/step - loss: 0.6235 - accuracy: 0.6613 - val_loss:
0.5774 - val_accuracy: 0.7385
Epoch 6/15
39/39 [=====] - 133s 3s/step - loss: 0.5724 - accuracy: 0.7075 - val_loss:
0.5375 - val_accuracy: 0.7523
Epoch 7/15
39/39 [=====] - 131s 3s/step - loss: 0.5582 - accuracy: 0.7431 - val_loss:
0.4850 - val_accuracy: 0.8073
Epoch 8/15
```

Type here to search 29°C Haze 18:31 03-08-2022

Home Page - Select or create a notebook x drowsiness-detection-model-tensorflow x +

localhost:8888/notebooks/drowsiness-detection-model-tensorflow.ipynb

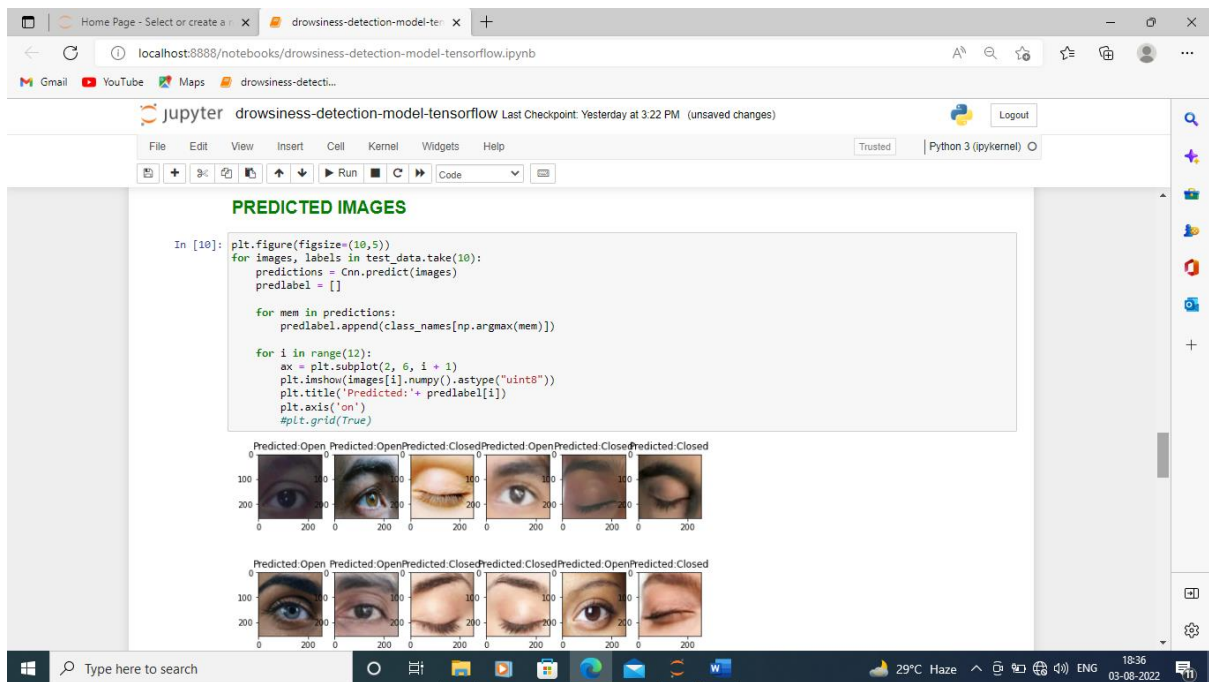
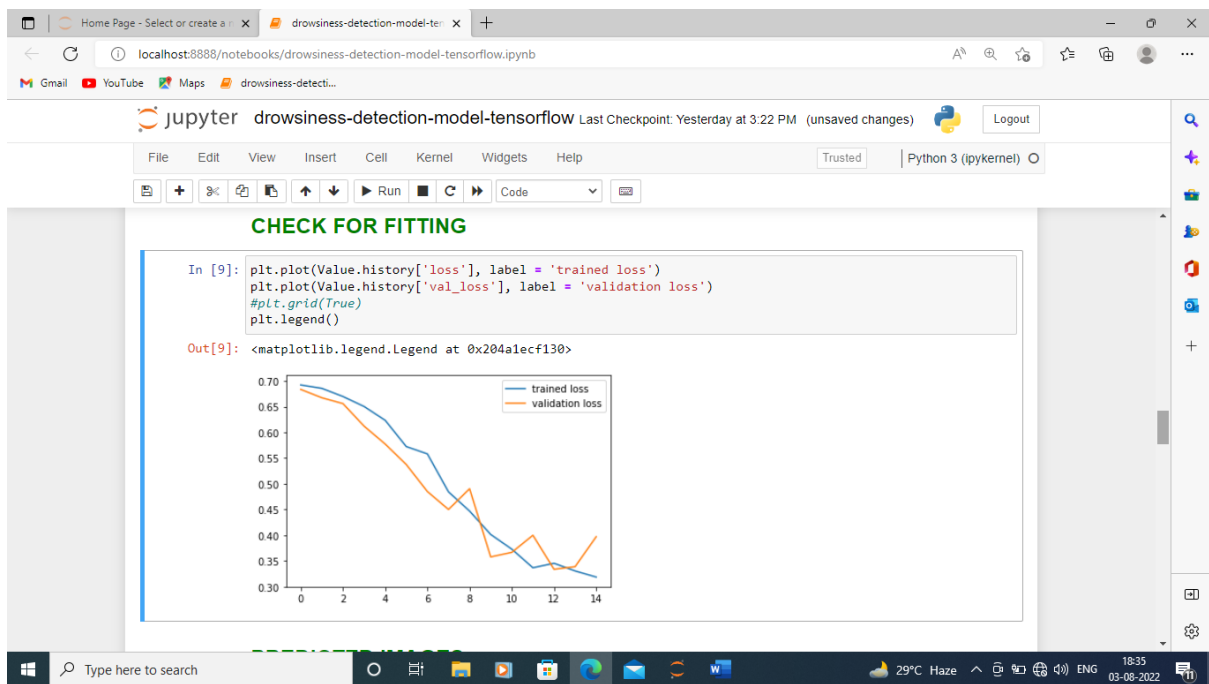
jupyter drowsiness-detection-model-tensorflow Last Checkpoint: Yesterday at 3:22 PM (autosaved) Logout

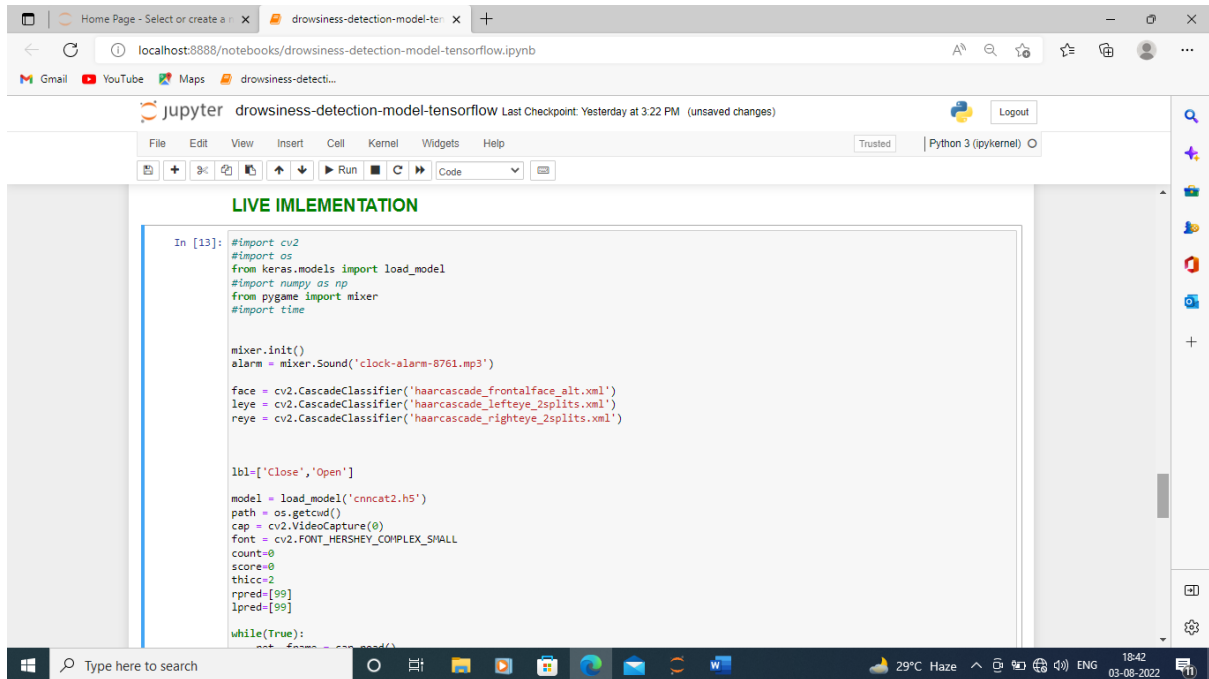
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

```
39/39 [=====] - 131s 3s/step - loss: 0.5582 - accuracy: 0.7431 - val_loss:
0.4850 - val_accuracy: 0.8073
Epoch 8/15
39/39 [=====] - 134s 3s/step - loss: 0.4844 - accuracy: 0.7934 - val_loss:
0.4501 - val_accuracy: 0.8211
Epoch 9/15
39/39 [=====] - 130s 3s/step - loss: 0.4469 - accuracy: 0.8015 - val_loss:
0.4905 - val_accuracy: 0.7844
Epoch 10/15
39/39 [=====] - 134s 3s/step - loss: 0.4015 - accuracy: 0.8331 - val_loss:
0.3579 - val_accuracy: 0.8899
Epoch 11/15
39/39 [=====] - 129s 3s/step - loss: 0.3729 - accuracy: 0.8436 - val_loss:
0.3667 - val_accuracy: 0.8578
Epoch 12/15
39/39 [=====] - 130s 3s/step - loss: 0.3367 - accuracy: 0.8647 - val_loss:
0.3999 - val_accuracy: 0.8486
Epoch 13/15
39/39 [=====] - 136s 3s/step - loss: 0.3455 - accuracy: 0.8574 - val_loss:
0.3338 - val_accuracy: 0.8899
Epoch 14/15
39/39 [=====] - 132s 3s/step - loss: 0.3305 - accuracy: 0.8622 - val_loss:
0.3389 - val_accuracy: 0.8853
Epoch 15/15
39/39 [=====] - 131s 3s/step - loss: 0.3186 - accuracy: 0.8703 - val_loss:
0.3972 - val_accuracy: 0.8486
```

Type here to search 29°C Haze 18:32 03-08-2022





```
In [13]: #import cv2
#import os
from keras.models import load_model
#import numpy as np
from pygame import mixer
#import time

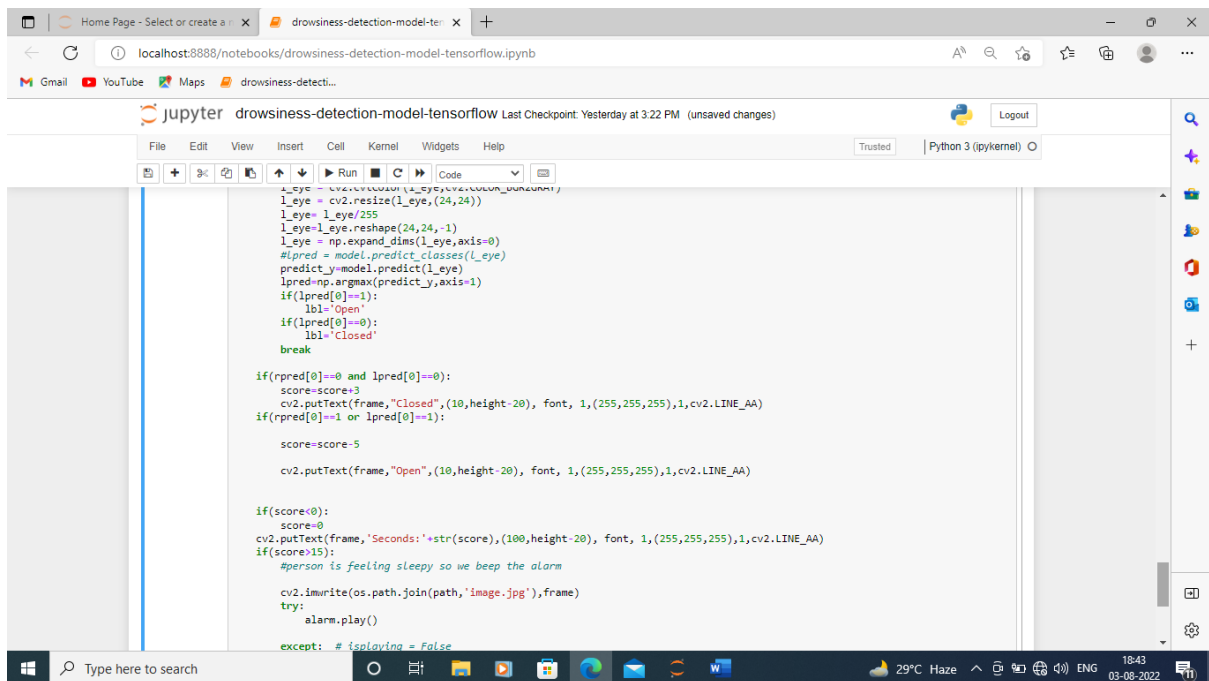
mixer.init()
alarm = mixer.Sound('clock-alarm-8761.mp3')

face = cv2.CascadeClassifier('haarcascade_frontalface_alt.xml')
leye = cv2.CascadeClassifier('haarcascade_lefteye_2splits.xml')
reye = cv2.CascadeClassifier('haarcascade_righteye_2splits.xml')

lbl=['Close','Open']

model = load_model('cnn-cat2.h5')
path = os.getcwd()
cap = cv2.VideoCapture(0)
font = cv2.FONT_HERSHEY_COMPLEX_SMALL
count=0
score=0
thicc=2
rpred=[99]
lpred=[99]

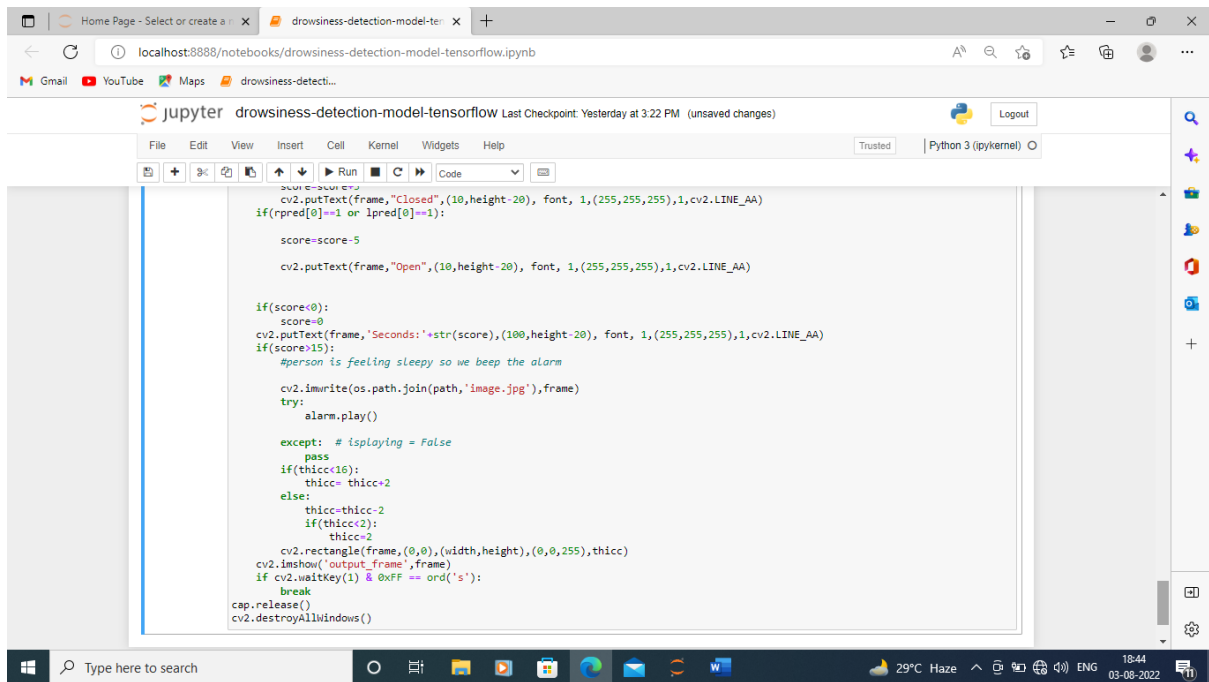
while(True):
    ret, frame = cap.read()
    #face = faceCascadeClassifier('haarcascade_frontalface_alt.xml')
```



```
l_eyeye = cv2.cvtColor(l_eyeye, cv2.COLOR_BGR2GRAY)
l_eyeye = cv2.resize(l_eyeye, (24,24))
l_eyeye = l_eyeye/255
l_eyeye=l_eyeye.reshape(24,24,-1)
l_eyeye = np.expand_dims(l_eyeye,axis=0)
#lpred = model.predict_classes(l_eyeye)
predict_y=model.predict(l_eyeye)
lpred=np.argmax(predict_y,axis=-1)
if(lpred[0]==1):
    lbl='Open'
if(lpred[0]==0):
    lbl='Closed'
break

if(rpred[0]==0 and lpred[0]==0):
    score=score+3
cv2.putText(frame,"Closed", (10,height-20), font, 1,(255,255,255),1,cv2.LINE_AA)
if(rpred[0]==1 or lpred[0]==1):
    score=score-5
cv2.putText(frame,"Open", (10,height-20), font, 1,(255,255,255),1,cv2.LINE_AA)

if(score<0):
    score=0
cv2.putText(frame,"Seconds:"+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:
        alarm.play()
    except: # isPlaying = False
```



10.2 OUTPUT

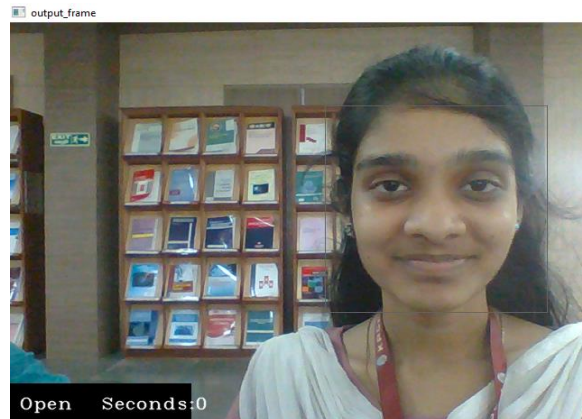


FIGURE 14: OPENED EYE

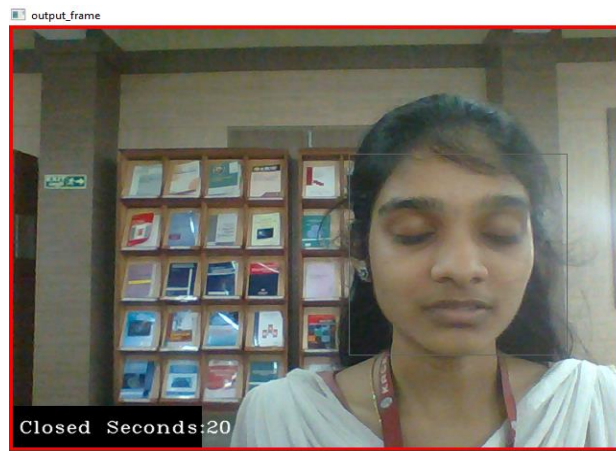


FIGURE 15: CLOSED EYE

The alarm will be ringing. It indicates the red colour border.

CHAPTER 11

CONCLUSION AND FUTURE SCOPE

11.1 CONCLUSION

It completely meets the objectives and requirements of the system. The framework has achieved an unfaltering state where all the bugs have been disposed of. The framework cognizant clients who are familiar with the framework and comprehend it's focal points and the fact that it takes care of the issue of stressing out for individuals having fatigue-related issues to inform them about the drowsiness level while driving.

11.2 FUTURE SCOPE

The model can be improved incrementally by using other parameters like blink rate, yawning, state of the car, etc. If all these parameters are used it can improve the accuracy by a lot. We plan to further work on the project by adding a sensor to track the heart rate in order to prevent accidents caused due to sudden heart attacks to drivers. Same model and techniques can be used for various other uses like Netflix and other streaming services can detect when the user is asleep and stop the video accordingly. It can also be used in application that prevents user from sleeping.

CHAPTER 12

REFERENCE

1. E. Rogado, J.L. García, R. Barea, L.M. Bergasa, Member IEEE and E. López, February, 2013, “Driver Fatigue Detection System”, Proceedings of the IEEE International Conference on Robotics and Biometrics, Bangkok, Thailand.
2. Miaou, “Study of Vehicle Scrap page Rates,” Oak Ridge National Laboratory, Oak Ridge, TN,, S.P.,April 2012
3. drowsiness-detection-using-convolutional-neural-networks-face-recognition-and-tensorflow-56cdfc8315ad
4. H. Singh, J. S. Bhatia, and J. Kaur, “Eye tracking based driver fatigue monitoring and warning system”, in Proc. IEEE IICPE, New Delhi, India, Jan. 2014.
5. open/closed eye analysis for drowsiness detection; p.r.tabrizi and r. a. zoroofi.