A PROJECT REPORT ON

Driver Drowsiness Detection System

Submitted in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

In

Computer Science and Engineering

By

Robbi Akshaya (21A81A0550)

Valla Siva Naga Tanuja (21A81A0562)

Maddula Lok Satish (21A81A0526)

Koyi Siva Ganapathi (21A81A0525)

Makinedi Hemanth Venkata Satya Sai (21A81A0527)

Under the Esteemed Supervision of

Mr.R.L.Phani Kumar M.Tech.,

Sr. Asst. Professor

Department of Computer Science and Engineering (Accredited by N.B.A.)

SRI VASAVI ENGINEERING COLLEGE(Autonomous)

(Affiliated to JNTUK, Kakinada)

Pedatadepalli, Tadepalligudem-534101, A.P 2020-21

SRI VASAVI ENGINEERING COLLEGE (Autonomous)

Department Of Computer Science and Engineering

Pedatadepalli, Tadepalligudem

Certificate

This is to certify that the Project Report entitled **"Driver Drowsiness Detection System"** submitted by **R.AKSHAYA (21A81A0550), V.S.N.TANUJA (21A81A0562), M.LOK SATISH (21A81A0526), K.SIVA GANAPATHI (21A81A0525), M.HEMANTH VENKATA SATYA SAI (21A81A0527)** for the award of the degree of Bachelor of Technology in the Department of Computer Science and Engineering, during the academic year 2023-2024.

Name of Project Guide

Mr . R . L . Phani Kumar M.Tech.,
Sr. Asst .Professor

Head of the Department

Dr. D Jaya Kumari M.Tech.,Ph.D..
Professor & HOD.

External Examiner

DECLARATION

We hereby declare that the project report entitled "**Driver Drowsiness Detection System**" submitted by us to Sri Vasavi Engineering College(Autonomous), Tadepalligudem, affiliated to JNTUK Kakinada in partial fulfilment of the requirement for the award of the degree of B.Tech in Computer Science and Engineering is a record of Bonafide project work carried out by us under the guidance of **Mr.R.L.Phani Kumar M.Tech.,** Sr. Asst. Professor. We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree in this institute or any other institute or University.

Project Associates

R . Akshaya(21A81A0550)

V.S.N. Tanuja (21A81A0562)

M . Lok Satish (21A81A0526)

K . Siva Ganapathi (21A81A0525)

M . H. V. Satya Sai(21A81A0527)

ACKNOWLEDGEMENT

First and foremost, we sincerely salute to our esteemed institute **SRI VASAVI ENGINEERING COLLEGE,** for giving us this golden opportunity to fulfill our warm dream to become an engineer.

Our sincere gratitude to our project guide **Mr.R.L.Phani Kumar M.Tech.,** Sr. Asst. Professor ,Department of Computer Science and Engineering**,** for his timely cooperation and valuable suggestions while carrying out this project.

We express our sincere thanks and heartful gratitude to **Dr. D. Jaya Kumari**, Professor & Head of the Department of Computer Science and Engineering, for permitting us to do our project.

We express our sincere thanks and heartful gratitude to **Dr. G.V.N.S.R. Ratnakara Rao**, Principal, for providing a favourable environment and supporting us during the development of this project.

Our special thanks to the management and all the teaching and non-teaching staff members, Department of Computer Science and Engineering, for their support and cooperation in various ways during our project work. It is our pleasure to acknowledge the help of all those respected individuals.

We would like to express our gratitude to our parents, friends who helped to complete this project.

Project Associates

R . Akshaya(21A81A0550)

V.S.N. Tanuja (21A81A0562)

M . Lok Satish (21A81A0526)

K . Siva Ganapathi (21A81A0525)

M . H. V. Satya Sai(21A81A0527)

ABSTRACT

Drowsiness detection is an important research area in the field of computer vision and driver safety. It involves the use of image and video processing techniques to detect signs of drowsiness in drivers, such as drooping eyelids or changes in facial expression. This can help to prevent accidents caused by driver fatigue, which is a leading cause of road accidents worldwide.

Drowsiness detection systems typically use machine learning algorithms and computer vision techniques to analyze the driver's face in real-time and identify signs of fatigue. This can involve detecting changes in facial expressions, such as drooping eyelids or yawning, as well as tracking the driver's eye movements to detect signs of drowsiness.

Recent advances in deep learning and neural networks have led to significant improvements in the accuracy and reliability of drowsiness detection systems, making them an increasingly important tool for improving road safety. These systems are now widely used in the automotive industry, and are also being explored for use in other applications, such as aviation and industrial safety.

CONTENTS

LIST OF FIGURES

LIST OF ABBREIVATIONS

CNN	Convolutional Neural Network
EEG	Electroencephalogram
HRV	Heart Rate Variability
SVM	Support Vector Machine
PPG	Photo Plethysmo Graphy
FFT	Fast Fourier Transform
ROC	Receiver Operation Curve
EAR	Eye Aspect Ratio
EOG	Electrooculography
ReLU	Rectified Linear Units

Chapter 1: Introduction

1. INTRODUCTION

1.1. Introduction

On an average 1200 road accidents record daily in India out of which 400 leads to direct death and rest gets effected badly. The major reason of these accidents is drowsiness caused by both sleep and alcohol. Due to driving for long time or intoxication, drivers might feel sleepy which is the biggest distraction for them while driving. This distraction might cost death of driver and other passengers in the vehicle and at the same time it also causes death of people in the other vehicles and pedestrians too.

To prevent these situations, we as a team , propose a system which can detect the drowsiness of the driver using CNN (Convolutional Neural Network) , Python, Open CV. Here, we used Python, Open CV, Keras (Tensor flow Library) to build a system that can detect features from the face of the drivers and alert them if ever they fall asleep while driving. The system detects the eyes and prompts if it is closed or open. If the eyes are closed for 3 seconds or if the blinking count of the eye is more than 10, it will play the alarm to get the driver's attention, to stop cause its drowsy . We have build a CNN network which is trained on a dataset which can detect closed and open eyes. Then Open CV is used to get the live fed from the camera and run that frame through the CNN model to process it and classify whether it opened or closed eyes.

1.2. Motivation

Driver drowsiness detection systems are motivated by several important factors aimed at improving road safety and preventing accidents. Some of the key motivations for developing and implementing these systems include:

1. <u>Preventing Accidents</u> : The primary motivation is to reduce accidents caused by drowsy driving. Fatigue impairs a driver's ability to react quickly and make sound judgments, making them more prone to accidents. Drowsiness detection systems can help intervene before a crash occurs.

2. Saving Lives : Drowsy driving is a significant contributor to road fatalities. By preventing accidents resulting from drowsiness, these systems can save lives and reduce the overall road mortality rate.

3. Reducing Injuries : In addition to fatalities, drowsy driving can lead to severe injuries, both for the drowsy driver and other road users. Detecting drowsiness early can minimize the severity of accidents and reduce injury rates.

4. Enhancing Road Safety : Driver drowsiness detection contributes to overall road safety by keeping drivers alert and attentive. This is essential for the safety of passengers, pedestrians, and other road users.

5. Improving Driver Health : Drowsiness detection systems encourage drivers to take breaks, rest, or change drivers when necessary. Promoting better sleep and healthy driving habits can improve the well-being of drivers.

6. Legal and Ethical Responsibility : Governments and organizations have a legal and ethical responsibility to protect citizens from harm on the roads. Implementing drowsiness detection systems aligns with this responsibility.

10. Technological Advancements : Advances in technology, such as computer vision, machine learning, and sensor technology, make it feasible to develop accurate and reliable drowsiness detection systems. The motivation is driven by the potential to harness these technological capabilities for safety.

11. Public Awareness : Increased awareness about the dangers of drowsy driving has contributed to the motivation to develop and adopt these systems. Public campaigns and education efforts emphasize the importance of staying awake and alert while driving.

Overall, the primary motivation for drowsiness detection systems is to save lives, prevent accidents, and enhance road safety by addressing the issue of drowsy driving. These systems are part of a broader effort to leverage technology and promote responsible driving habits.

1.3. Scope

Driver drowsiness detection technology holds significant promise in the realm of road safety and public health. Its scope extends to various areas, starting with its primary mission of preventing accidents caused by drowsy drivers. By detecting signs of fatigue and intervening in a timely manner, these systems can save lives and reduce injuries. They have a crucial role in the transportation industry, from long-haul trucking to public transportation, ensuring the safety of drivers, passengers, and cargo. Moreover, the economic impact of drowsy driving accidents, including healthcare costs, vehicle repairs, and insurance claims, can be mitigated through these systems, resulting in cost savings. With advancements in technology and regulation, the scope continues to evolve, making drowsiness detection user-friendly, legally compliant, and contributing to research insights into driver behavior. Ultimately, its scope spans safety, health, economics, and technology, with the overarching goal of creating safer roads and responsible driving habits.

1.4. Project Outline

Chapter 1: Introduction

Chapter 2: Literature Survey

Chapter 3: System Study and Analysis

Chapter 4: System Design

Chapter 5: Technologies

Chapter 6: Implementation

Chapter 7:Testing

Chapter 8: Screenshots

Chapter 9: Conclusion and FutureWork and References

Chapter 2: Literature Survey

2. Literature Survey

2.1. Paper 1

Title: Drowsiness Detection System Utilizing Physiological Signals.

Author: Trupti K. Dange, T. S. Yengatiwar.

Year of publication: 2013.

Keywords: EOG, ECG, EEG, HRV, SVM, driver drowsiness detection.

Physiological parameters-based techniques detect drowsiness based on drivers' physical conditions, such as heart rate, pulse rate, breathing rate, respiratory rate, and body temperature. These biological parameters are more reliable and accurate in detecting fatigue or drowsiness, as they change physiological parameters like blood pressure and heart rate, alerting drivers when they are near sleep. These invasive measures require electrode placement on the driver's body.

1) EEG-BASED DRIVER FATIGUE DETECTION

The drivers' fatigue detection system using Electroencephalogram (EEG) signals is proposed to avoid the road accidents usually caused due to drivers' fatigue. The proposed method firstly finds the index related to different drowsiness levels. The system takes EEG signal as input which is calculated by a cheap single electrode neuro signal acquisition device. To evaluate the proposed method, data set for simulated car driver under the different levels of drowsiness is collected locally. And result shows that the proposed system can detect all subjects of tiredness.

2) WAVELET ANALYSIS OF HEART RATE VARIABILITY & SVM CLASSIFIER

Li and Chung proposed a driver drowsiness detection system using wavelet analysis of Heart Rate Variability (HRV) and Support Vector Machine (SVM) classifier. The system categorizes alert and drowsy drivers using the wavelet transform of HRV signals over short durations. It uses Photo Plethysmo Graphy (PPG) signal input, Fast Fourier Transform (FFT) and wavelet features, with a Receiver Operation Curve (ROC) and SVM classifier for feature extraction and classification. The system shows improved results compared to the FFT-based method.

3) PULSE SENSOR METHOD

Mostly, previous studies focus on the physical conditions of drivers to detect drowsiness. That's why Rahim detects the drowsy drivers using infrared heart-rate sensors or pulse sensors. The pulse sensor measures the heart pulse rate from drivers' finger or hand. The sensor is attached with the finger or hand, detects the amount of blood flowing through the finger. Then amount of the blood's oxygen is shown in the finger, which causes the infrared light to reflect off and to the transmitter. The sensor picks up the fluctuation of oxygen that are connected to the Arduino as microcontroller. Then, the heart pulse rate is visualizing by the software processing of HRV frequency domain. Experimental results show that LF/HF (Low to high frequency) ratio decreases as drivers go from the state of being awake to the drowsy and many road accidents can be avoided if an alert is sent on time.

4) WEARABLE DRIVER DROWSINESS DETECTION SYSTEM

Mobile based applications have been developed to detect the drowsiness of drivers. But mobile phones distract the drivers' attention and may cause accident. To address the issue, Lenget proposed the wearable-type drowsiness detection system. The system uses selfdesigned wrist band consists of PPG signal and galvanic skin response sensor. The data collected from the sensors are delivered to the mobile device which acts as the main evaluating unit. The collected data are examined with the motion sensors that are built-in in the mobiles. Then five features are extracted from the data: heart rate, respiratory rate, stress level, pulse rate variability, and adjustment counter. The features are moreover used as the computation parameters to the SVM classifier to determine the drowsiness state. The experimental results show that the accuracy of the proposed system reaches up to 98.02 %. Mobile phone generates graphical and vibrational alarm to alert the driver.

5) WIRELESS WEARABLES METHOD

To avoid the disastrous road accidents, Warwick proposed the idea for drowsiness detection system using wearable Bio sensor called Bio-harness. The system has two phases. In the first phase, the physiological data of driver is collected using bio-harness and then analyzes the data to find the key parameters like ECG, heart rate, posture and others related to the drowsiness. In the second phase, drowsiness detection algorithm will be designed and develop a mobile app to alert the drowsy drivers.

6) DRIVER FATIGUE DETECTION SYSTEM

Chellappa presents the Driver fatigue detection system. The basic of the system is to detect the drowsiness when the vehicle is in the motion. The system has three components: external hardware (sensors and camera), data processing module and alert unit. Hardware unit communicates over the USB port with the rest of the system. Physiological and physical factors like pulse rate, yawning, closed eyes, blink duration and others are continuously monitored using somatic sensor. The processing module uses the combination of the factors to detect drowsiness. In the end, alert unit alerts the driver at multiple stages according to the severity of the symptoms.

7) HYBRID APPROACH UTILIZING PHYSIOLOGICAL FEATURES

Awais has proposed a hybrid method for detecting drowsiness by integrating ECG and EEG features. The method extracts time and frequency domain features from EEG, then measures heart rate, HRV, low frequency, high frequency, and LF/HF ratio. Subjective sleepiness is measured to study its relationship with drowsiness. Statistically significant features are selected using t-tests. The method integrates ECG and EEG features using SVM to improve detection performance. The method also examines channel reduction and its impact on detection performance. Results show that combining ECG and EEG improves system performance, reaching 80% accuracy level. This makes the system feasible for practical drowsiness detection.

2.2. Paper 2

Title: Drowsiness Detection with OpenCV (Using Eye Aspect Ratio)

Author: Adrian Rosebrock.

Year of publication: 2017.

Keywords: EAR, SVM, eye blink detection.

A real-time algorithm to detect eye blinks in a video sequence from a standard camera is proposed. Recent landmark detectors, trained on in-the wild datasets exhibit excellent robustness against a head orientation with respect to a camera, varying illumination and facial expressions. We show that the landmarks are detected precisely enough to reliably estimate the level of the eye opening. The proposed algorithm therefore estimates the landmark positions, extracts a single scalar quantity – eye aspect ratio (EAR) – characterizing the eye opening in each frame.

Typically, the face and eyes are detected by a Viola-Jones type detector. Next, motion in the eye area is estimated from optical flow, by sparse tracking, or by frame-to-frame intensity differencing and adaptive thresholding. Finally, a decision is made whether the eyes are or are not covered by eyelids.

Nowadays, robust real-time facial landmark detectors that capture most of the characteristic points on a human face image, including eye corners and eyelids, are available, Most of the state-of-the-art landmark detectors formulate a regression problem, where a mapping from an image into landmark positions or into other landmark parametrization is learned. These modern landmark detectors are trained on “in-the-wild datasets” and they are thus robust to varying illumination, various facial expressions, and moderate non-frontal head rotations

Proposed method:

The eye blink is a fast closing and reopening of a human eye. Each individual has a little bit different pattern of blinks. The pattern differs in the speed of closing and opening, a degree of squeezing the eye and in a blink duration. The eye blink lasts approximately 100-400 ms. We propose to exploit state-of-the-art facial landmark detectors to localize the eyes and eyelid contours. From the landmarks detected in the image, we derive the eye aspect ratio (EAR) that is used as an estimate of the eye opening state. Since the per frame EAR may not necessarily recognize the eye blinks correctly, a classifier that takes a larger temporal window of a frame into account is trained. The EAR is mostly constant when an eye is open and is getting close to zero while closing an eye.

A real-time eye blink detection algorithm was presented. We quantitatively demonstrated that regression-based facial landmark detectors are precise enough to reliably estimate a level of eye openness. While they are robust to low image quality (low image resolution in a large extent) and in-the-wild phenomena as non-frontality, bad illumination, facial expressions, etc.

The proposed SVM method that uses a temporal window of the eye aspect ratio (EAR), outperforms the EAR thresholding.

$$\text{EAR} = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

2.3. Paper 3

Title: Real Time Driver Fatigue Detection Based on SVM Algorithm.

Authors: Burcu Kir Savas, Yasar Becerkli.

Year of publication: 2018.

Keywords: fatigue detection , SVM , driving safety , video processing

PROPOSED SYSTEM

Vladimir N. Vapnik's proposed SVM-based driver fatigue prediction system aims to enhance driver safety by distinguishing tired drivers from non-tired ones. The system uses five stages: PERCLOS, count of yawn, internal zone of mouth opening, count of eye blinking, and head detection to extract attributes from video recordings. The classification stage is performed using Support Vector Machine (SVM), with real-time video recordings used during the training phase. The study involved 10 volunteers aged 18-30, who drove training simulations during the experiments. The results of driver fatigue detection were classified using SVM classification algorithm. The data was divided into two groups: 80% training data set (568 data set) and 20% test data set (145 data set). Cross validation was selected as 10.

Measurements are as follows:

• TP means that a real fatigue is detected;

• TN means that a non-fatigue situation is correctly detected as non-fatigue by the system;

• FP means that a non-fatigue situation is incorrectly detected as real fatigue;

• FN means that a real fatigue situation is incorrectly detected as non-fatigue.

The accuracy of driver fatigue calculated as

$$\text{Accuracy} = (TP+TN)/(TP+TN+FP+FN)$$

As a result of the study, a system was designed to investigate the effects of fatigue and 8 insomnia on drivers physical transient behaviors, and to simulate drowsy drivers as a result of research. In this system, behavioral detection model is used for detecting fatigue drivers. The proposed system has five stages: PERCLOS, count of yawn, internal zone of the mouth opening, count of eye blinking and head detection to extract attributes from video recordings. As a result, the system is classified as SVM in two classes (not fatigue / fatigue). In this study a Real Time Driver Fatigue Detection SVM Based on SVM Algorithm is presented whose test] results showed that the accuracy rate of fatigue detection is up to 97.93%

2.4. Paper 4

Title: Driver drowsiness detection using ANN image processing.

Authors: T. Vesselenyi1 , S. Moca1 , A. Rus1 , T. Mitran1 , B. Tătaru1.

Keywords: EEG, EOG, ANN, Deep Learning

Year of publication: 2017.

The 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.

EEG (Electroencephalography) and EOG (Electrooculography) signals measurement and on the eye state (closed or opened) image classification.

The EEG method monitors the brain activity through a sensor placed on a specific part of the scalp, The EOG method tracks the eye movements by measuring the signals from the muscles which are acting on the eye.The eye image analysis can monitor the opened or the closed state of the eye.

The EEG and EOG sensors, electrodes which have to be fixed with a conductive gel and in most devices must transmit the signal by wire, present a major discomfort. Research in the field of advanced materials and MEMS technology may solve these problems, as for example the use of dry electrodes for EEG.

EEG Developments field are supported by efforts to create brain – computer interfaces for different applications, including devices that help disabled people. In this research the central point is to distinguish between low and high alpha rhythm peaks, which can make the difference between alert and drowsy states.

To use EOG signals acquisitioned from 3 sensors (EOG1, EOG2, EOG3). After preprocessing, four types of different signals were identified. The combination of these four types of signals give the information to distinguish between left, right, up and down movements of the eye.

Drowsiness detection using the processing of the driver's eye images

Artificial neural networks, including Deep Learning methods like Deep Belief Networks, Restricted Boltzmann Machines, and Deep Autoencoders, were used to classify a driver's drowsiness or alert state. The study used the Matlab Neural Network Toolbox with 1 layer ANN and the autoencoder module to analyze 200 images of a driver during a regular driving process. The drowsiness was linked to images with closed eyes, while the alert state was linked to images with open eyes. These methods are widely used in various applications, including image classification.

2.5. Paper 5

Title: Deep CNN: A Machine Learning Approach for Driver Drowsiness Detection Based on Eye State.

Authors: Venkata Rami Reddy Chirra, Srinivasulu Reddy Uyyala and Venkata Krishna Kishore Kolli.

Year of publication: 2019.

Keywords: Deep CNN, Feature extraction

PROPOSED DEEP CNN BASED DROWSINESS DETECTION SYSTEM
Proposed system algorithm

(1) Viola-jones face detection algorithm is used to detect the face the images and given as input to Viola-jones eye detection algorithm

(2) Once the face is detected, Viola-jones eye detection algorithm is used to extract the eye region from the facial images and given as input to CNN.

(3) CNN with four convolutional layers is used to extract the deep features and those features are passed to fully connected layer.

(4) Soft max layer in CNN classify the images in to sleepy or non-sleepy images. Face detection and eye region extraction

Whole face region may not be required to detect the drowsiness but only eyes region is enough for detecting drowsiness. At first step by using the Viola-jones face detection algorithm face is detected from the images. Once the face is detected, Viola-jones eye detection algorithm is used to extract the eye region from the facial images. it is the first algorithm used for face detection. For the face detection the Viola-Jones algorithm having three techniques those are Haar-like features, Ada boost and Cascade classifier. In this work, Viola-Jones object detection algorithm with Haar cascade classifier was used and implemented using OPEN CV with python. Haar cascade classifier uses Haar features for detecting the face from images.

Feature extraction and classification

Feature extraction is one type of dimensionality reduction where useful parts of an image represented as a feature vector.

Convolutional Neural Network

Convolutional Neural Network (CNN) is used in the proposed system for detection of driver drowsiness. Since a feature vector is needed for each drowsy image to compare with existing features in a database to detect either drowsy or not. Usually CNNs requires fixed size images as input so preprocessing is required. The preprocessing includes extracting the key frames from video based on temporal changes and store in database. From these stored images, feature vectors are generated in convolution layers of CNN. These feature vectors are then used for the detecting the driver drowsiness. CNN have layers like convolutional layers, pooling (max, min and average) layers, ReLU layer and fully connected layer. Convolution layer is having kernels (filters) and each kernel having width, depth and height. This layer produces the feature maps as a result of calculating the scalar product between the kernels and local regions of image. CNN uses pooling layers (Max or Average) to minimize the size of the feature maps to speed up calculations. In this layer, input image is divided into different regions then operations are performed on each region. In Max Pooling, a maximum value is selected for each region and places it in the corresponding place in the output. ReLU (Rectified Linear Units) is a nonlinear layer. The ReLU layer applies the max function on all the values in the input data and changes all the negative values to zero. The following equation shows the ReLU activation function.

CONCLUSION

In this proposed work a new method is proposed for driver drowsiness detection based on eye state. This determines the state of the eye that is drowsy or non-drowsy and alert with an alarm when state of the eye is drowsy. Face and eye region are detected using ViolaJones detection algorithm. Stacked deep convolution neural network is developed to extract features and used for learning phase. A SoftMax layer in CNN classifier is used to classify the driver as sleep or non-sleep. Proposed system achieved 96.42% accuracy. Proposed system 465 effectively identifies the state of driver and alert with an alarm when the model predicts drowsy output state continuously. In future we will use transfer learning to improve the performance of the system.

Chapter 3: System Study and Analysis

3. SYSTEM STUDY AND ANALYSIS

3.1 Problem Statement

The advancement of technology has led to the integration of driver drowsiness detection systems to enhance road safety. These systems play a crucial role in real-time monitoring of driver behavior and physiological indicators to identify signs of drowsiness. However, challenges remain in terms of the accuracy and reliability of detection, adaptability to varying conditions, and addressing privacy concerns. To further the field of driver drowsiness detection, the problem statement is to develop innovative solutions that improve accuracy, adaptability while complying with regulatory standards, ensuring the safety and well-being of drivers and other road users.

3.2 Existing System

The existing system extracts the frames from video to obtain 2-D images. Face is detected in the frames using Haar-Adaboost face detection method. After detecting the face, facial landmarks like positions of eye, nose and mouth are marked on the images. From the facial landmarks, position of eyes and mouth are quantified. Using these extracted features and machine learning methods, a decision is obtained about the drowsiness of the driver. Convolutional neural network is applied for classification of eyes, which detects drowsiness of driver by considering blinking of eyes.

3.3 Limitations of Existing System

- The system doesn't give correct results in varying lighting conditions like bright sunlight or low light. This can lead to inaccurate face detection, landmark localization, and feature extraction.
- This method is having complex computations due to large number of landmarks.
- Factors such as temporary blinking, eye irritation, or driver habits could contribute to misinterpretation.

3.4 Proposed System

This is a system which can detect the drowsiness of the driver using CNN (Convolutional Neural Network) , Python, Open CV. Here, we used Python, Open CV, Keras (tensor flow) to build a system that can detect features from the face of the drivers and alert them if ever they fall asleep while driving. The system detects the eyes and prompts if it is closed or open. If the eyes are closed for 3 seconds it will play the alarm to get the driver's attention, to stop cause its drowsy. We have build a CNN network which is trained on a dataset which can detect closed and open eyes. Then Open CV is used to get the live fed from the camera and run that frame through the CNN model to process it and classify weather it opened or closed eyes.

3.5 Advantages of Proposed System

- In this, we used haarcascade classifier which is faster and efficient than haar adaboost.
- The system works even when a driver is wearing spectacles.
- We used frame count to avoid misinterpretation due to temporary eye blinking.

3.6 Functional Requirements

These are the functional requirements that are required.

- Recording the driver's behavior, the moment the trip begins.
- Capturing Driver's facial features over the course of long trip
- Eye tracking
- Displays frame count of open and closed eyes
- Alerts the driver by raising an alarm based on the frame count of closed eyes.

3.7 Non-Functional Requirements

These are the non-functional requirements that are required.

- Camera capturing the video should be of high resolution.

- System should work even in low light conditions.
- Alarm raised should be of high volume to wake the driver up.

3.8 User Interface Requirements

3.8.1 System Requirements

3.8.1.1 Software requirements

These are the software requirements that are required.

- Operating system: Windows 7, Mac OS, Linux
- Language: Python 3.6
- IDE: Visual Studio Code
- Libraries used:

  OpenCV
  Keras
  Numpy
  Pandas

3.8.1.2 Hardware Requirements

These are the hardware requirements that are required.

- Processor: 64-bit, quad-core, 2.5 GHz minimum per core
- RAM: 4 GB or more
- Display:1024 x 768 or higher resolution monitors
- Camera: A webcam
- Speaker

Chapter 4: System Design

4. SYSTEM DESIGN

4.1 System Architecture Design

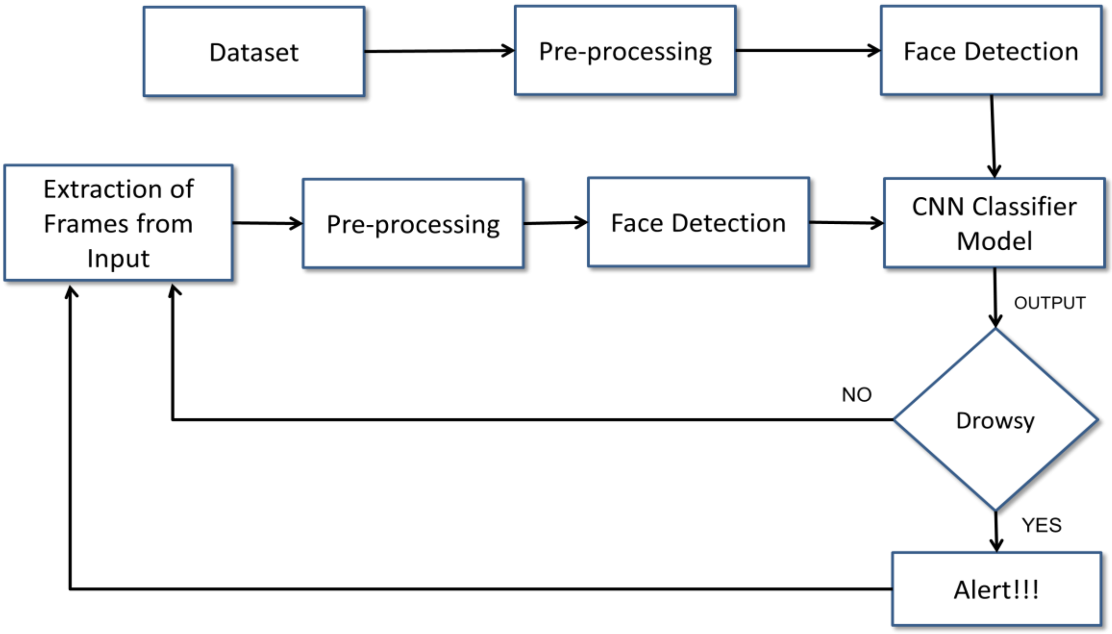


Fig. 4.1.1 System Architecture

4.2 UML Diagrams

4.2.1 Use Case Diagram:

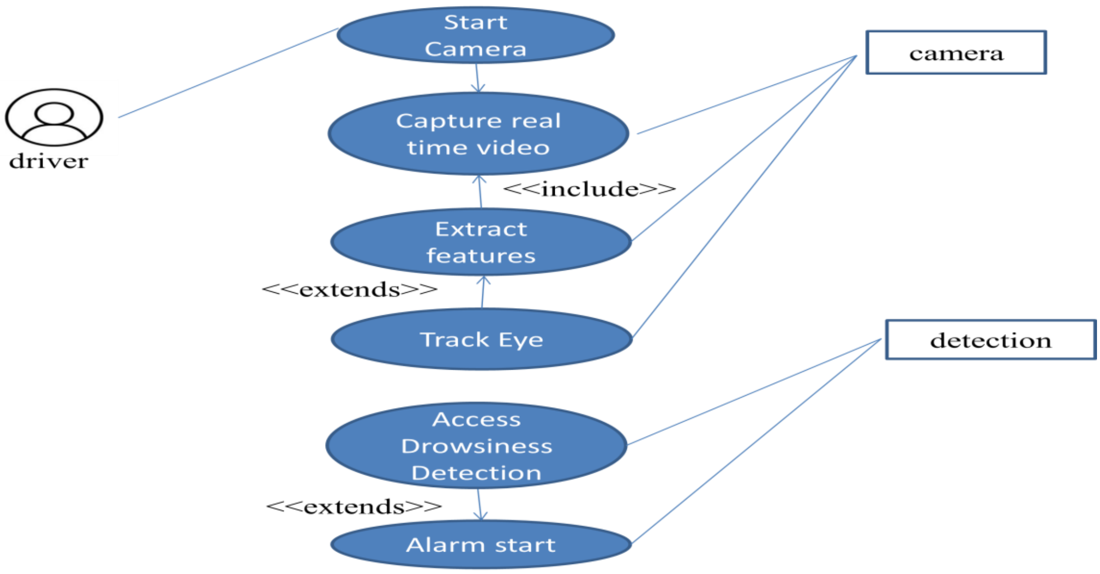


Fig. 4.2.1 Use Case diagram

4.2.2 Data Flow Diagram:

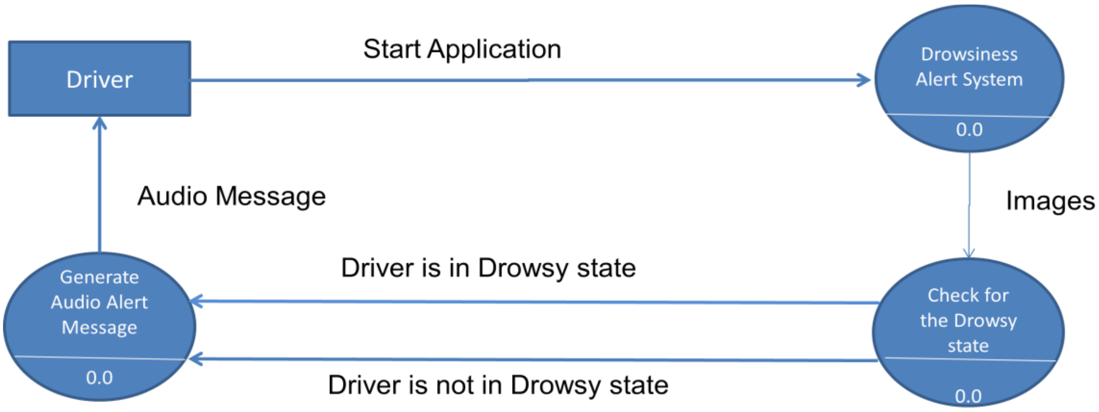


Fig. 4.2.2 Data Flow diagram

4.2.3 Class Diagram:

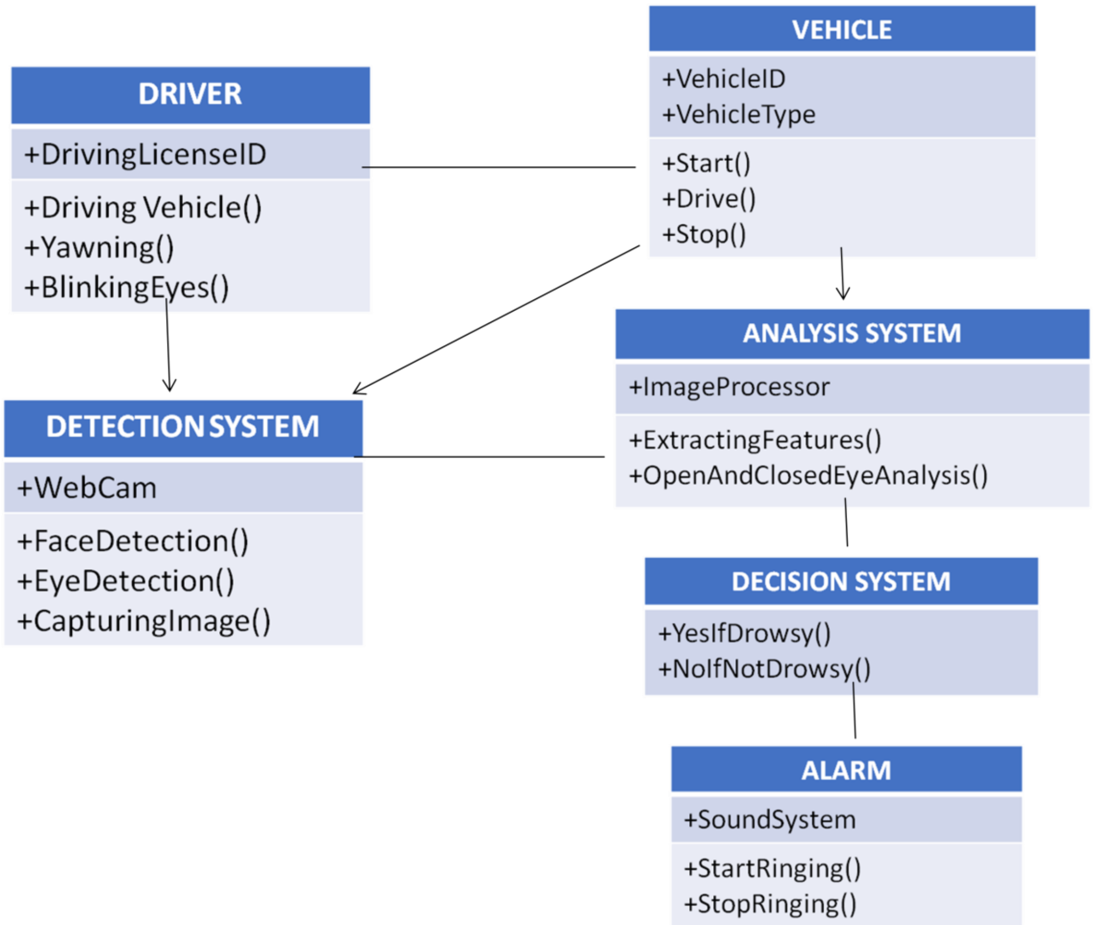


Fig. 4.2.3 Class diagram

Chapter 5: Technologies

5. TECHNOLOGIES

5.1 Python

5.1.1 About Python

Python is a high-level programming language with dynamic semantics, making it ideal for Rapid Application Development and scripting. Its simple, easy-to-learn syntax emphasizes readability, reducing maintenance costs. Python supports modules and packages, encouraging modularity and code reuse. The Python interpreter and standard library are available in source or binary form for all major platforms, and the edit-test-debug cycle is fast due to no compilation step. Debugging Python programs is easy, with the interpreter raising exceptions when an error is found. A source-level debugger allows inspection of variables, evaluation of expressions, and code stepping. The fast edit-test-debug cycle makes it an effective approach. Python's easy-to-learn syntax and portability make it popular. The followings facts give us the introduction to Python:

1.Python was developed by Guido van Rossum at Stichting Mathematisch Centrum in the Netherlands.

2.It was written as the successor of programming language named ‘ABC’.

3.It’s first version was released in 1991.

4.The name Python was picked by Guido van Rossum from a TV show named Monty Python’s Flying Circus.

5.It is an open source programming language which means that we can freely download it and use it to develop programs. It can be downloaded from www.python.org.

6.Python programming language is having the features of Java and C both. It is having the elegant ‘C’ code and on the other hand, it is having classes and objects like Java for object-oriented programming.

7.It is an interpreted language, which means the source code of Python program would be first converted into bytecode and then executed by Python virtual machine.

5.1.2 Required Python Libraries

OpenCV (Open Source Computer Vision Library) :

OpenCV is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms. OpenCV is written natively in C++ and has a templated interface that works seamlessly with STL containers.

Numpy:

NumPy, created in 2005 by Travis Oliphant, is the core package for scientific computing in Python. It offers a multidimensional array object, derived objects, and routines for fast operations on arrays. The ndarray object is at the core of the package, providing numerous supporting functions. NumPy arrays are stored at one continuous place in memory, allowing efficient access and manipulation. Although partially written in Python, most parts require fast computation in C or C++.

Tensorflow:

TensorFlow is an open-source software library. TensorFlow was originally developed by researchers and engineers working on the Google Brain Team within Google's Machine Intelligence research organization for the purposes of conducting machine learning and deep neural networks research, but the system is general enough to be applicable in a wide variety of other domains as well. Google open-sourced TensorFlow in November 2015.

Keras:

Keras is a python based open-source library used in deep learning (for neural networks). Keras was released in March 2015. It can run on top of TensorFlow, Microsoft CNTK or Theano. It is very simple to understand and use, and suitable for fast experimentation. It is designed to be fast and easy for the user to use. Keras models can be run both on CPU as well as GPU. Keras is the best platform out there to work on neural network models. The API that Keras has a user-friendly where a beginner can easily understand. Keras has the advantage that it can choose any libraries which support it for its backend support. Keras provides various pre-trained models which help the user in further improving the models the user is designing.

Matplotlib:

Matplotlib is a low level graph plotting library in python that serves as a visualization utility. Matplotlib was created by John D. Hunter. Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

Sklearn:

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. It was originally called scikits.learn and was initially developed by David Cournapeau as a Google summer of code project in 2007. Later, in 2010, Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, and Vincent Michel, from FIRCA (French Institute for Research in Computer Science and Automation), took this project at another level and made the first public release (v0.1 beta) on 1st Feb. 2010.

Playsound module in Python:

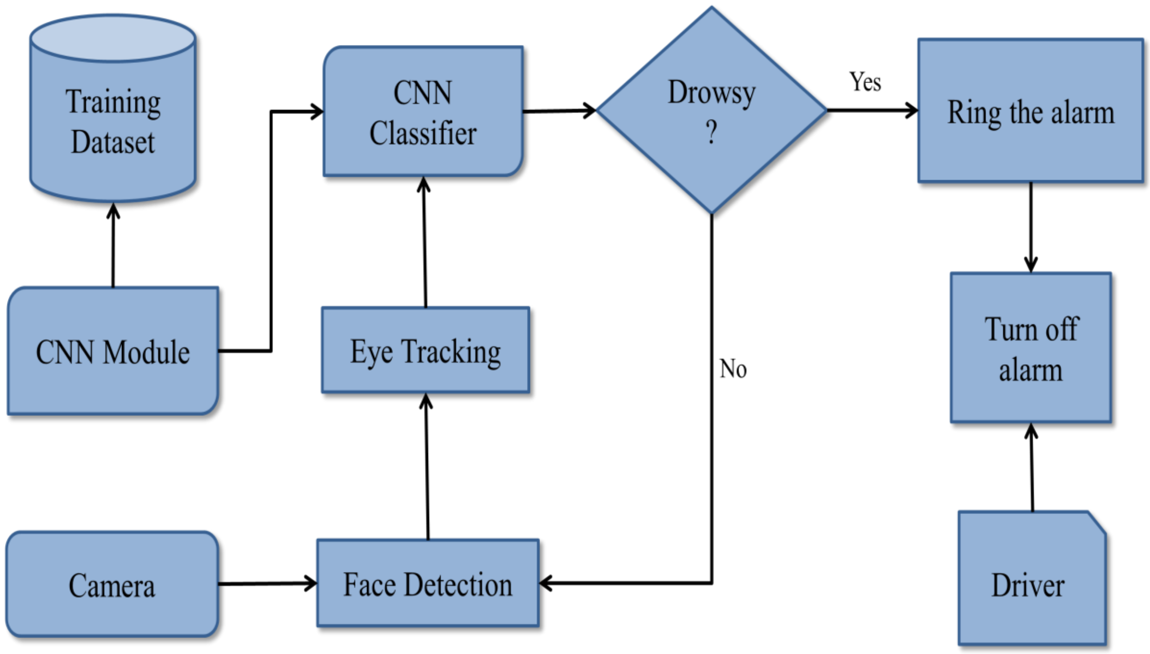
The playsound module is a cross platform module that can play audio files. This doesn’t have any dependencies, simply install with pip in your virtual environment and run. Implementation is different on platforms. It uses windll.winm on Windows, AppKit.NSSound on Apple OS X and GStreamer on Linux. The playsound module contains a function named playsound(). It works with both WAV and MP3 files.

Chatpter-6: Implementation

6. IMPLEMENTATION

6.1. Implementaion Steps

- Data Collection
- Data Preprocessing
- CNN Model Design
- Model Training
- System Integeration
- Testing and Evaluation



6.2 CNN Algorithm

```
model = Sequential()
model.add(Conv2D(256, (3, 3), activation="relu", input_shape=(145,145,3)))

model.add(MaxPooling2D(2, 2))
model.add(Conv2D(128, (3, 3), activation="relu"))

model.add(MaxPooling2D(2, 2))
model.add(Conv2D(64, (3, 3), activation="relu"))

model.add(MaxPooling2D(2, 2))
model.add(Conv2D(32, (3, 3), activation="relu"))

model.add(MaxPooling2D(2, 2))
model.add(Flatten())
```

model.add(Dropout(0.5))
model.add(Dense(64, activation="relu"))

model.add(Dense(4, activation="softmax"))
model.compile(loss="categorical_crossentropy", metrics=["accuracy"],

optimizer="adam")
model.summary()

```
import cv2
import numpy as np

from keras.models import load_model
from keras.preprocessing.image import img_to_array
from playsound import playsound
from threading import Thread


def start_alarm(sound):
    """Play the alarm sound"""
    playsound('data/alarm.mp3')


classes = ['Closed', 'Open']
face_cascade =
cv2.CascadeClassifier("data/haarcascade_frontalface_default.xml")
left_eye_cascade =
cv2.CascadeClassifier("data/haarcascade_lefteye_2splits.xml")
right_eye_cascade =
cv2.CascadeClassifier("data/haarcascade_righteye_2splits.xml")
cap = cv2.VideoCapture(0)
model = load_model("drowiness_new7.h5")
count = 0
alarm_on = False
alarm_sound = "data/alarm.mp3"
status1 = ''
status2 = ''
while True:
    _, frame = cap.read()
    height = frame.shape[0]
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, 1.3, 5)
    for (x, y, w, h) in faces:
        cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 0), 1)
        roi_gray = gray[y:y+h, x:x+w]
        roi_color = frame[y:y+h, x:x+w]
```

```
        left_eye = left_eye_cascade.detectMultiScale(roi_gray)
        right_eye = right_eye_cascade.detectMultiScale(roi_gray)
        for (x1, y1, w1, h1) in left_eye:
            cv2.rectangle(roi_color, (x1, y1), (x1 + w1, y1 +
h1), (0, 255, 0), 1)
            eye1 = roi_color[y1:y1+h1, x1:x1+w1]
            eye1 = cv2.resize(eye1, (145, 145))
            eye1 = eye1.astype('float') / 255.0
            eye1 = img_to_array(eye1)
            eye1 = np.expand_dims(eye1, axis=0)
            pred1 = model.predict(eye1)
            status1=np.argmax(pred1)
            #print(status1)
            #status1 = classes[pred1.argmax(axis=-1)[0]]
            break

        for (x2, y2, w2, h2) in right_eye:
            cv2.rectangle(roi_color, (x2, y2), (x2 + w2, y2 +
h2), (0, 255, 0), 1)
            eye2 = roi_color[y2:y2 + h2, x2:x2 + w2]
            eye2 = cv2.resize(eye2, (145, 145))
            eye2 = eye2.astype('float') / 255.0
            eye2 = img_to_array(eye2)
            eye2 = np.expand_dims(eye2, axis=0)
            pred2 = model.predict(eye2)
            status2=np.argmax(pred2)
            #print(status2)
            #status2 = classes[pred2.argmax(axis=-1)[0]]
            break

        # If the eyes are closed, start counting
        if status1 == 2 and status2 == 2:
        #if pred1 == 2 and pred2 == 2:
            count += 1
            cv2.putText(frame, "Eyes Closed, Frame count: " +
str(count), (10, 30), cv2.FONT_HERSHEY_COMPLEX, 1, (0, 0, 255),
1)
            # if eyes are closed for 10 consecutive frames, start
the alarm
            if count >= 10:
                cv2.putText(frame, "Drowsiness Alert!!!", (100,
height-20), cv2.FONT_HERSHEY_COMPLEX, 1, (0, 0, 255), 2)
                if not alarm_on:
```

```
                    alarm_on = True
                    # play the alarm sound in a new thread
                    t = Thread(target=start_alarm,
args=(alarm_sound,))
                    t.daemon = True
                    t.start()
        else:
            cv2.putText(frame, "Eyes Open", (10, 30),
cv2.FONT_HERSHEY_COMPLEX, 1, (0, 255, 0), 1)
            count = 0
            alarm_on = False

    cv2.imshow("Drowsiness Detector", frame)

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

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

Chapter-7: Testing

7.TESTING

7.1 Testing Objectives

1. Accurate drowsy driver identification
2. Generalization to real-world scenarios
3. Real-time efficiency for embedded systems
4. Non-intrusive and clear alerts
5. Consistent and reliable performance

7.2 Testing Strategies

Testing a driver drowsiness detection system is crucial to ensure its accuracy, reliability, and effectiveness. Here are some testing strategies and approaches you can use to evaluate such a system:

7.2.1. Unit Testing:

- Focus on individual components: Unit testing focuses on testing the individual components of the system, such as the modules responsible for image processing, machine learning, and alerting mechanisms.
- White-box testing: White-box testing involves examining the internal structure of the code to ensure that it meets the design specifications.
- Code coverage: Code coverage metrics can be used to ensure that all parts of the code are adequately tested.

7.2.2. Integration Testing:

- Testing interactions between components: Integration testing focuses on testing the interactions between different components of the system.
- Black-box testing: Black-box testing involves testing the system from an external perspective, without knowing its internal structure.
- End-to-end testing: End-to-end testing involves testing the entire system from beginning to end, simulating real-world usage scenarios.

7.2.3. System Testing:

- Evaluation of overall system performance: System testing evaluates the overall performance of the system against its functional and non-functional requirements.
- Testing under varying conditions: The system should be tested under varying conditions, such as different lighting conditions, driver positions, and driving conditions.
- Performance testing: Performance testing measures the system's ability to handle different workloads and data volumes.

7.2.4. Acceptamce Testing:

- User validation: Acceptance testing involves validating the system with the end-users to ensure that it meets their expectations and requirements.
- Sign-off from stakeholders: Stakeholders, such as product managers, customers, and regulatory bodies, should sign off on the system's acceptance.
- Deployment planning: Acceptance testing should lead to deployment planning and roll-out strategies for the system.

7.3.Test Cases

The following table represents test cases observed while doing the project to detect the driver’s drowsiness.

Table-1:System Testing

Test Cases	Eyes detected	Eyes closure	Result
Case 1	No	No	Fail (No result)
Case 2	No	Yes	Fail (No result)
Case 3	Yes	No	Pass (Eyes open)
Case 4	Yes	Yes	Pass (Eyes closed, Voice alert)

The system's approach states that the driver is feeling fatigued if their eyes are closed for an extended period of time (greater than the specified number of frames). From this point on, one of these exceptional cases occurs, and the related outcome occurs. When there is an obstruction, accuracy slightly decreases. The right results depend greatly on the ambient lighting conditions.

Chapter-8: Screenshots

8. SCREENSHOTS

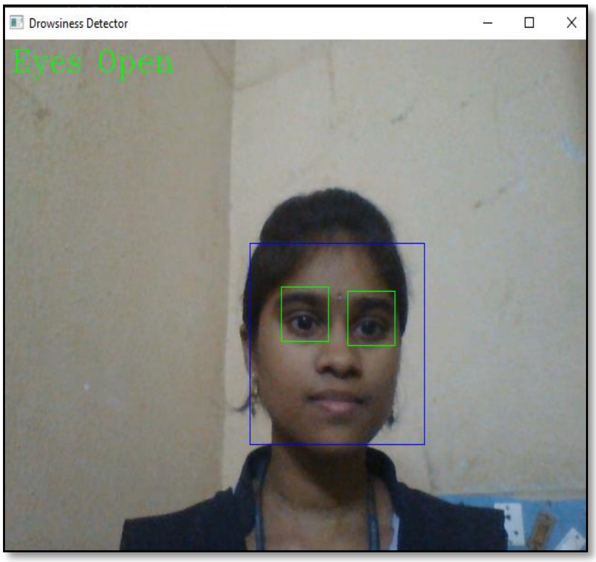


Fig. 8.1 Open Eyes

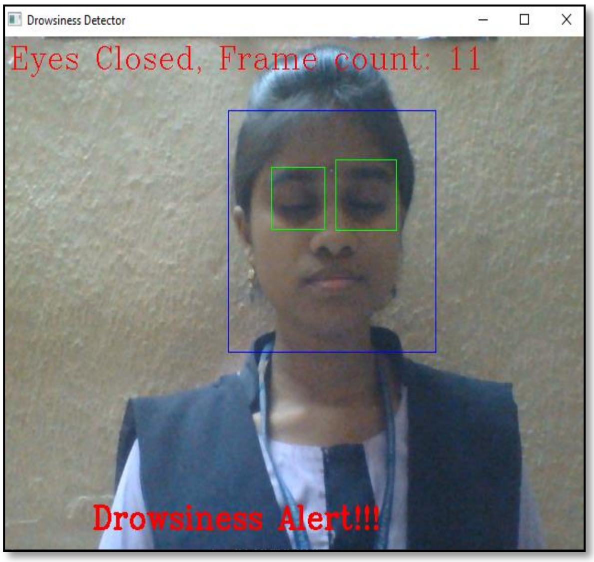


Fig. 8.2 Closed Eyes

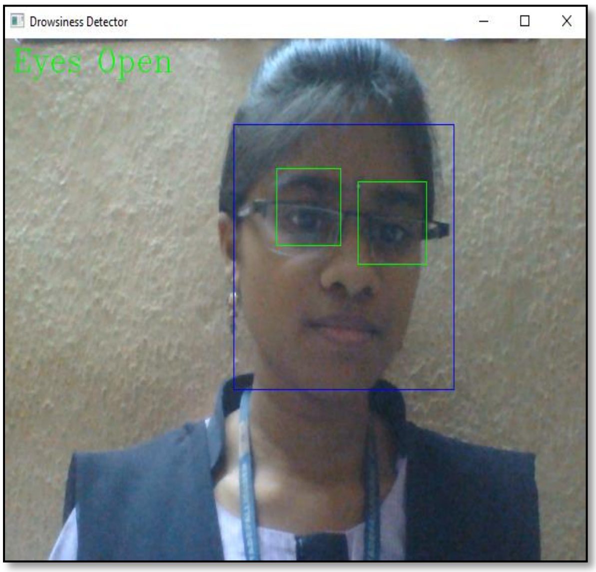


Fig. 8.3 Open Eyes with Spectacles

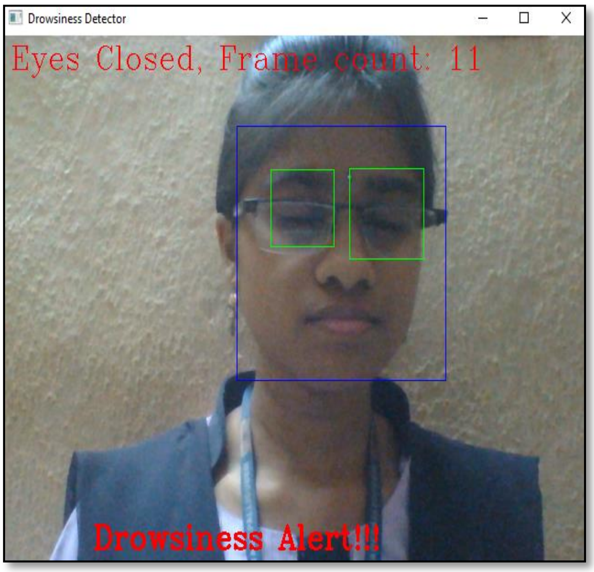


Fig. 8.4 Closed Eyes with Spectacles

Chapter-9: Conclusion, Future Work And References

9.CONCLUSION, FUTURE WORK AND REFERENCES

9.1.Conclusion:

In order to detect a driver’s drowsiness, facial features and eyes were identified on the video of an individual driving. We used OpenCV to detect faces and eyes using a haar cascade classifier. Convolutional neural network was implemented to classify eyes as open or closed. Drowsiness was determined on the basis of frequency of closed eyes. An alarm was set to ring after the detection to alert the driver. There will be limitations concerning the detection of drivers’ conditions and facial expressions due to factors like darkness, light reflection, obstructions by drivers’ hands and wearing of sunglasses. Convolutional neural gives better performance and facial extraction method accompanies it, as an additional drowsiness detection technique which is often used with other facial extraction methods.

9.2. Future Work:

- The system can be made more accurate using various other parameters such as State of the Car, Detecting Foreign Substances on Face etc.
- Develop a user-friendly interface to display the drowsiness detection results to the driver and allow for user interaction.
- Deploy the system and regularly update the model with new data to adapt to changing conditions and improve accuracy.
- It can be used to develop an IOT device that can be installed in the car which can stop the engine after detecting the driver’s drowsiness.
- Ensure compliance with privacy and data protection regulations when collecting and processing driver data.

9.3. References:

[1] Rosebrock, Adrian. "Drowsiness detection with OpenCV." *Py Image Search* (2017).

[2] Savaş, Burcu Kır, and Yaşar Becerikli. "Real time driver fatigue detection based on SVM algorithm." *2018 6th International Conference on Control Engineering & Information Technology (CEIT)*. IEEE, 2018.

[3] Vesselenyi, Tiberiu, et al. "Driver drowsiness detection using ANN image processing." *IOP conference series: materials science and engineering*. Vol. 252. No. 1. IOP Publishing, 2017.

[4] Chirra, Venkata Rami Reddy, Srinivasulu Reddy Uyyala, and Venkata Krishna Kishore Kolli. "Deep CNN: A Machine Learning Approach for Driver Drowsiness Detection Based on Eye State." *Rev. d'Intelligence Artif.* 33.6 (2019): 461-466.

[5] Aditya Ranjan, Karan Vyas, Sujay Ghadge, Siddharth Patel, Suvarna Sanjay Pawar, "Driver Drowsiness Detection System Using Computer Vision.", in International Research Journal of Engineering and Technology(IRJET), 2020.

[6] Saini, Vandna, and Rekha Saini. "Driver drowsiness detection system and techniques: a review." International Journal of Computer Science and Information Technologies 5.3 (2014): 4245- 4249.

[7] Alshaqaqi, Belal, et al. "Driver drowsiness detection system." 2013 8th international workshop on systems, signal processing and their applications (WoSSPA). IEEE, 2013.

[8] Deng, Wanghua, and Ruoxue Wu. "Real-time driver-drowsiness detection system using facial features." Ieee Access 7 (2019): 118727-118738.

[9] Park, Sanghyuk, et al. "Driver drowsiness detection system based on feature representation learning using various deep networks." Asian Conference on Computer Vision. Cham: Springer International Publishing, 2016.

[10] Hashemi, Maryam, Alireza Mirrashid, and Aliasghar Beheshti Shirazi. "Driver safety development: Real-time driver drowsiness detection system based on convolutional neural network." SN Computer Science 1 (2020): 1-10.