ROAD ACCIDENT AVOIDING SYSTEM (RAAS)

By

NC Amina Rashid

ASC Muhammad Vahhaaj

PC Shahid Hussain

PC Raheel Fida Ch

Supervised by:

Dr. Naima Iltaf, PhD

Co-supervised by:

Asst. Prof. Mobeena Shahzad, PhD

Submitted to the faculty of Department of Computer Software Engineering,
Military College of Signals, National University of Sciences and Technology, Islamabad,
in partial fulfillment for the requirements of B.E Degree in Software Engineering.

June 2022

In the name of ALLAH, the Most benevolent, the Most Courteous

CERTIFICATE OF CORRECTNESS AND APPROVAL

This is to officially state that the thesis work contained in this report

“Road Accident Avoiding System”

is carried out by

<u>NC Amina Rashid</u>

<u>ASC Muhammad Vahhaaj</u>

<u>PC Shahid Hussain</u>

<u>PC Raheel Fida</u>

under my supervision and that in my judgement, it is fully ample, in scope and excellence, for the degree of Bachelor of Software Engineering in Military College of Signals, National University of Sciences and Technology (NUST), Islamabad.

Approved by

Supervisor

Co-supervisor

Date: ___________

DECLARATION OF ORIGINALITY

We hereby declare that no portion of work presented in this thesis has been submitted in support of another award or qualification in either this institute or anywhere else.

ACKNOWLEDGEMENTS

Allah Subhan’Wa’Tala is the sole guidance in all domains.

Our parents, colleagues, most of all our supervisor, __________ and co-supervisor, __________ without your guidance, this project would not have been possible.

The group members, who through all adversities worked steadfastly.

Plagiarism Certificate (Turnitin Report)

This thesis has ____ similarity index. Turnitin report endorsed by Supervisor is attached.

NC Amina Rashid
00000245816

ASC Muhammad Vahhaaj
00000278789

PC Shahid Hussain
00000278765

PC Raheel Fida Ch
00000278768

Signature of Supervisor

Signature of Co-supervisor

ABSTRACT

Our traffic and tendency for aggressive and dangerous driving are among the many things that the subcontinent is notorious for. Every year, this results in an increasing number of horrifying automobile accidents that could have been avoided. A lot of the elements that allow this to happen are related to the driver's situation, i.e., the state in which they are driving. The system in place will estimate the risk of an accident at any point during travel using facial recognition, body language analysis, and traffic density. A camera over the dash will give a live feed to the system, which will compare facial expressions and check for indicators of exhaustion and intoxication using Machine Learning Algorithms. It will also use a driver's body language to see whether they are distracted by any electronics or other variables, and warn them about the dangers via a linked app. The driver will be given guidance on how to proceed, and if the motorist follows the advice, the odds of prevention are considerably improved. In the event of unforeseen incidents, the system also serves as a black box, storing data for use in legal procedures (if necessary).

Table of Contents

List of Figures

Chapter 1: Introduction

Given how far humans have come with technology it is only predictable that strides will be made to make everyday life not only easier but safer. Look at some of the biggest leaps yet, vaccines were made to combat sickness, the steam engine was made to make transport easier, faster, durable, comfortable, and much safer than the carts used in the yesteryears and surgical improvements were made to significantly increase chances of recovery. Every single contribution made has either made experiences simpler or safer. It is only logical that we make such a contribution in one of these key aspects of human life. Smart technology is already integrating its way into the transportation sector and it's only predictable that measures be taken to ensure safety in heavy traffic such as that of South-East Asia. The National Highway Traffic Safety Administration (NHTSA) estimates, inattentive driving caused 91,000 crashes in 2017, resulting in 50,000 people being injured, and killing roughly 800 individuals. These figures are merely a rough estimate, the prediction is that up to 6,000 fatal crashes occur each year.

With the advancement in engineering knowledge and scientific leaps, transport safety has found a respectful place in the present research interests.

And yes, one could argue that perhaps self-driving cars may remove the need altogether and provide a much safer alternative but until they become the norm it is best if we improve upon manned vehicles. Needless to say, that driving as a skill will always be needed and so will the technology that makes it less dangerous.

With that in mind, the Road Accident Avoiding System, aims to help people to make smarter and safer decisions if they simply knew how much their state and the traffic density contributes to the risk of them running into an unfortunate accident.

1.1 Overview

It is but natural that any place with an explosive population, a great number of drivers and the lack of respect for safety rules and/or driver health is brewing a disastrous concoction. The world keeps on moving, the days keep on progressing, time doesn’t stop and neither does a person caught in this fast-paced life that allows him/her little time to pay any heed to safety. Our job is to solve this issue technically. With the best of what technology has to offer at our disposal we can provide drivers with the necessary supervision to make sure they are not a danger to themselves. As stated before, the surveys show us a bleak image of what the simplest of details has cost us in terms of human life, property and the trauma that comes with severe

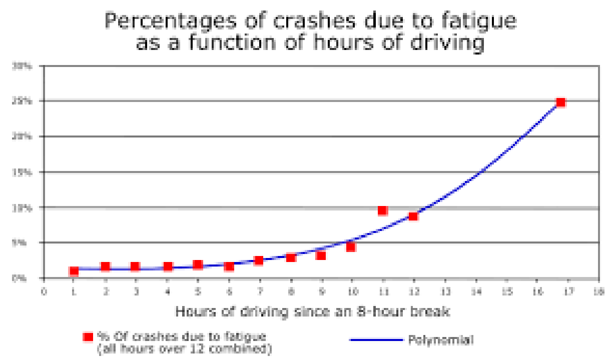


Figure 1: Effects of Fatigue on Safety

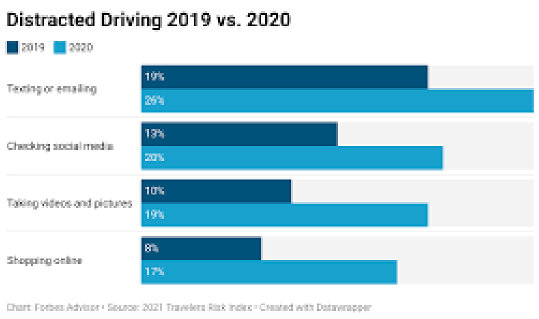


Figure 2: Distracted Driving analysis

injury.

It is the need of the hour to fashion some mechanism to avoid such disastrous outcomes considering how with every passing day more and more individuals (licensed or otherwise) add to the number of drivers, narrowing the space and margin for error in ongoing traffic. Therefore, this system not only reduces the chances of accidents but also instills healthier driving habits within the people that use it for their own good.

1.2 Problem Statement

Pakistan is a third world underdeveloped country with the 5th largest population in the world. Naturally that means more traffic, more chances of collision, less room for error. To put things into perspective:

1) According to statistics, 7,000 to 10,000 persons have died in Pakistani traffic accidents every year during the last ten years, the core reason being distracted driving.
2) One of the main factors contributing to traffic accidents is driver weariness (RTC).
3) Information was given by the National Highways and Motorway Police (NHMP). Using Australian Transportation Safety Bureau (ATSB) criteria for identifying fatigue-related RTC, data from all RTC on highways from 2003 to 2012, as well as data from 2003 to 2011 on the National Highways (N-5), were thoroughly analysed. There were 1750 RTCs on roadways between 2003 and 2012, 497 of which were fatigue-related. There were 5080 RTC on (N-5) between 2003 and 2011, 483 of which were fatigue-related.
4) While the average rate of fatigue-related RTC on highways was 28 percent, it was found that the rate on National Highways (N-5) was only 10 percent.As a result, fatigue related RTC is increasingly common on highways.

And with increase in population the situation keeps becoming more and more dire.

1.3 Proposed Solution

The technology is designed to detect driver irresponsibility, carelessness, and potential distractions that could lead to road accidents, hence preventing potentially fatal circumstances. It assists drivers in maintaining safe driving practices and serves as a warning (emitting a beeping sound through the speaker) if the driver is determined to be distracted on the road. It addresses common causes of car accidents, such as texting while driving, sleepiness, fatigue, and distracted driving, such as failing to pay attention to the road. In the event of unforeseen inciden3ts, the system also serves as a black box, storing data for use in legal procedures (if necessary).

1.4 Working Principle

The project works on the principles of digital image processing (DIP) and machine learning (ML) algorithms. The project is organized into various modules, each of which is intertwined with the next. The following is a list of modules:

- Datasets
- Dataset training and processing
- Output extraction
- Decision based upon Output
- Integration
- GUI

1.4.1 Datasets:

An integral part of the project is the preparation of datasets. The dataset comprises of images of various facial features particularly the eyes, expressions, pupil dilation, the state of the features when the driver is drowsy, distracted, or even intoxicated although that is more relevant in western countries than in Pakistan.

1.4.1.1 Drowsiness Detection Dataset:

This Data comprises of sample images that serve as the basis for comparison with the driver, the primary focus is the degree of openness of the eyes and their movement as eyes tend to remain still for a considerable duration when a person is fatigued due to loss of focus on the road

1.4.1.2 Distraction Detection Dataset:

This Data comprises of sample images of various poses and directions a person might look towards when checking their phone, when texting while driving or eating while driving. Their movement is a bit erratic as well.

The presence of safety gears i.e., seatbelts does not require an extensive dataset as the belt does not have varying features, we require just enough training that can simply detect the object

1.4.2 Dataset training and processing:

The designated datasets are used to train models using machine learning.

1.4.2.1 HaaR algorithm:

It is an object detection algorithm that finds faces in still photos and moving pictures.

1.4.3 Output Extraction:

The outputs we get from the system are based on the actions/state of the individual whether there's any signs of distraction/fatigue detected.

1.4.4 Decision based upon Outputs:

The extracted outputs, the state/actions detected will determine what follows

1.4.4.1 Drowsiness based decision:

The primary target of the system is to detect drowsiness upon which the system will alert the driver through sound and a notification on the screen. The driver is advised to stop the car on the roadside if possible or let someone else drive if possible.

1.4.4.2 Distraction based decision:

The secondary target of the system is to detect distraction upon which the driver will also be alerted through sound/notification. The driver is advised to keep their eyes on the road and stay alert.

1.4.4.3 Safety gear-based decision:

The 3rd target of the system is to detect the presence of safety gears upon which the driver will also be alerted through sound/notification. The driver is advised to take all possible measures to protect him/herself if an accident was unavoidable.

1.4.5 Integration:

These many modules are then combined into a single standalone entity for a compact solution.

1.4.6 GUI:

The library used for the GUI is Tkinter and provides a visual as well as a medium of interaction between the user and the system.

1.4.7 Raspberry-PI:

The Raspberry Pi used for this project is Raspberry Pi 3, Raspberry Pi 4 is also compatible and very user friendly. It is, in layman’s terms, a computer that can be programmed and easily fit into a desired system and is helpful in implementing the project. It receives video feed from its designated camera that will aid the algorithm to detect faces and then carry out analysis to find any traces of the concerned states.

Figure 3: Raspberry-PI

1.5 Objectives

1.5.1 General Objectives:

"To create a unique state-of-the-art software integrated hardware prototype that is powered by Digital Image Processing and Machine Learning (ML), potentially saving lives."

1.5.2 Academic Objectives:

- Development of a Road Accident Avoiding System
- To implement Machine Learning, Digital Image Processing, Python and advanced AI technology and techniques.
- Teamwork
- To design a project that could be deemed fit to save lives be it one or many.

1.6 Scope

This project is applicable to all vehicles, could help all drivers, and finds use where ever there are roads. Other areas of impact are

1.6.1 New/Amateur Drivers

The system could assist new drivers to get used to better accident avoiding measures and gain confidence in their driving skills.

1.6.2 Traffic Police

The system's camera feed could be offered as a black box of the accident for the police in their investigations to help them rule out any foul play.

1.6.3 Forensic Scientists/Engineers

The data from the system could help forensic scientists and engineers collect more information in the events of a crime such as homicide, carjacking etcetera and to better analyze system failures in vehicles.

1.6.4 Health and Safety Advisors

The system would greatly ease the jobs of health and safety advisors if they recommend it to their clients.

1.7 Deliverables

The final product would be a working system capable of detecting the mentioned states in any driver behind the wheel, the final process would look something like this:

- RAAS will use Facial detection using DIP and ML.

- Users will then be under observation during their drive while the camera focuses on their core features along with other indicators such as posture, blink rate, perspiration pupil size, and movement of all individual features and the body.
- The feed will be transmitted to the backend system where it will be analyzed and compared against existing data to check for anomalies.
- Based on the gathered data the computer will then follow the coded protocol as per the situation.
- The user will be alerted through sound and notifications along with instructions about what the statistics advise for a driver in their state to reduce the chances of putting them in danger.
- RAAS will successfully improve the driving experience for end-users and provide the authorities valuable data after collection.

1.8 Relevant Sustainable Development Goals

3 – Physical and mental well-being/Good Health and well-being

This system can be used by drivers of all experience and ages who operate a variety of cars. The goal is to improve their driving safety, as well as reducing the number of accidents. The initiative aims to improve the user's lifestyle, their choices and make responsible drivers out of them because any sort of minimal distraction/drowsiness reduces their reaction time and has the same effect as the impairment caused by 1% alcohol level and if we deploy this project and avoid such circumstances altogether, we can certainly save lives.

1.9 Structure of Thesis

The literature review, as well as the background and analysis study on which this thesis is founded, are included in Chapter 2.

The design and development of the project are covered in Chapter 3.

The code is evaluated and analyzed in depth in Chapter 4.

The project's conclusion is found in Chapter 5.

Chapter 6 focuses on the work that needs to be done for this project to be commercialized.

Chapter 1 Summary

So far, we’ve covered the purpose of the project it’s intended goals, why is it necessary and whom does it help, the working principles, the algorithms, data, the system’s integration, and the key technology it uses. A well rounded backgrounded is in place to start building understanding of the project in detail, in the next chapter we’ll go over the existing systems and their impact on driver habits as well as how RAAS aims to improve them.

Chapter 2: Literature Review

A new product is created by altering and improving the characteristics of similar items that have already been released. A literature review is a crucial phase in the transformation of a concept into a new product. Similarly, a careful examination of all similar projects is required for the development of a product and its replacement in the context of a transportation system. The following are the main points of our investigation.

- Background in the industry
- Existing solutions and their shortcomings
- Research Papers

2.1 Background in the Industry

Drowsiness is one of the leading causes of serious automobile accidents in our daily lives. According to the National Highway Traffic Safety Administration, around 100,000 collisions occur in the United States each year as a result of driver drowsiness, resulting in 1,550 deaths, 71,000 injuries, and $12.5 billion in costs. According to another survey, the US government and businesses spend an estimated $60.4 billion per year on accidents caused by drowsy driving, and consumers pay an additional $16.4 billion in property damage, health claims, missed time, and productivity as a result of drowsy driving. The National Sleep Foundation (NSF) found in 2010 that 54 percent of adult drivers had driven while fatigued and 28 percent had fallen asleep behind the wheel.

According to the German Road Safety Council (DVR), one in every four highway traffic fatalities is caused by driver tiredness.

The enormous mortality, injuries, and property damage caused by sleepiness necessitate a significant effort in building an effective system that can detect drowsiness and take appropriate measures before accidents occur. The United States Department of Transportation has also made advances in the development of intelligent vehicles to prevent such tragedies. With the growing interest in intelligent transportation systems, developing a reliable and practical sleepiness detection system is an important first step. Many studies are being undertaken with this goal in mind.

2.2 Existing solutions and their shortcomings

The existing solutions are quite generic. They manually handle this problem. They are virtually nonexistent on ground and does not have a functioning application. Furthermore, skilled driver is preferred for driving and driving practices are done before issue license. Their limitations are following:

- They are not practically to use.
- Its virtuality nonexistent on ground
- It does not have a functioning application
- Furthermore, it’s all very prototyped (it only tells yes or no) but it does not keep the data, nor it is compact solution that initiates warning and advice

Chapter 2 Summary

Chapter 2 was a window into the past systems of similar nature, their shortcomings, clearly when the matter at hand involves human lives, we simply cannot leave room for error or let flaws be brushed under the rug. Now that we have established what we have to tackle let’s look into the process of making such a system.

Chapter 3: Interfacing and Detection

Everything is a process, and much like any physical structure a software project has certain steps, certain needs must be met in order to carry it out, certain foundations must build, ground rules must be set. Gradually those steps take the shape of something that is in working order and helps change some aspect of human lives for the better. Let's look at some of the steps that make this possible.

3.1 Driver Detection

We are aware of the hazards and dangers of driving while under the influence of drowsiness, dizziness, tiredness, or sleepiness, or even texting while driving, but it is still a neglected component of a regular everyday duty, ladies and gentlemen, driving of a road vehicle. According to the National Safety Council, drowsy driving causes around 100,000 collisions, 71,000 injuries, and 1,550 fatalities each year (NSC).

According to AAA, drowsy driving accounts for an estimated 9.5 percent of all crashes.

However, we can assume that the true figure is significantly higher because it is difficult to determine whether a motorist was fatigued or sleepy at the time of the crash. In certain circumstances, dizzy driving is as dangerous as driving while intoxicated.

Here are some more statistics concerning driving while tired or drowsy. Drowsy driving is similar to drunk driving — 18 hours without sleep is equivalent to a blood alcohol content (BAC) of.05 percent (CDC); this occurs most frequently between midnight and 6 a.m., or in the late afternoon. People experience dips in their circadian rhythm—the human body's internal clock that regulates sleep—at certain times of day; frequently involve only a single driver (and no passengers) running off the road at high speeds with no evidence of braking; and frequently occur on rural roads and highways.

Detection of drivers is an essential part of our project. When a driver is detected, he is continuously monitored to detect presence of road accident-causing problems like sleepiness, tiredness, dizziness, and drowsiness. It also detects presence of tother situations that often lead to fatal road accidents such as identifying distractors if present, monitoring for signals of ignorance by the driver and not using seatbelts or safety gear and not paying attention at the road ahead. The process consists of three parts.

3.1.1 Preparing Dataset

A dataset is, quite simply, a bundle of data pieces that can be treated by a machine learning computer as a single unit for a systematic and prediction purposes. a good dataset should correspond to certain quality and quantity standards. For smooth and fast training, you should make sure your dataset is relevant and well-balanced.

In our project we have used advanced computer technologies such as Machine Learning, Digital Image Processing or Computer Vision and Artificial Intelligence.

Machine learning typically works with two data sets: training and test.

The first set you use is the training set, the largest of the three. Running a training set through a neural network teaches the net how to weigh different features, adjusting them coefficients according to their likelihood of minimizing errors in your results.

The second set is your test set. It functions as a seal of approval, and you don't use it until the end. After you've trained and optimized your data, you test your neural net against this final random sampling.

In this model we used a dataset of people to train the classifier of our system. Kaggle was used to obtain this dataset as it is an open repository. Kaggle offers a no-setup, customizable, Jupyter Notebooks environment. It is a huge repository of community published data & code.

The dataset that we used has over 1300+ images of faces with different facial attributes from varying races and communities. We have used this dataset to account for all skin colours, face orientations and maximum capability.

Inside Kaggle, you'll discover all of the code and data you'll need to complete your data science projects. We have access to over 50,000 public datasets and 400,000 public notebooks, allowing us to quickly complete any analysis. Road Accident Avoiding System (RAAS) is very good at detecting facial features of a user/driver and the system does good job at alarming the driver to take the necessary steps to avoid a potential road accident.

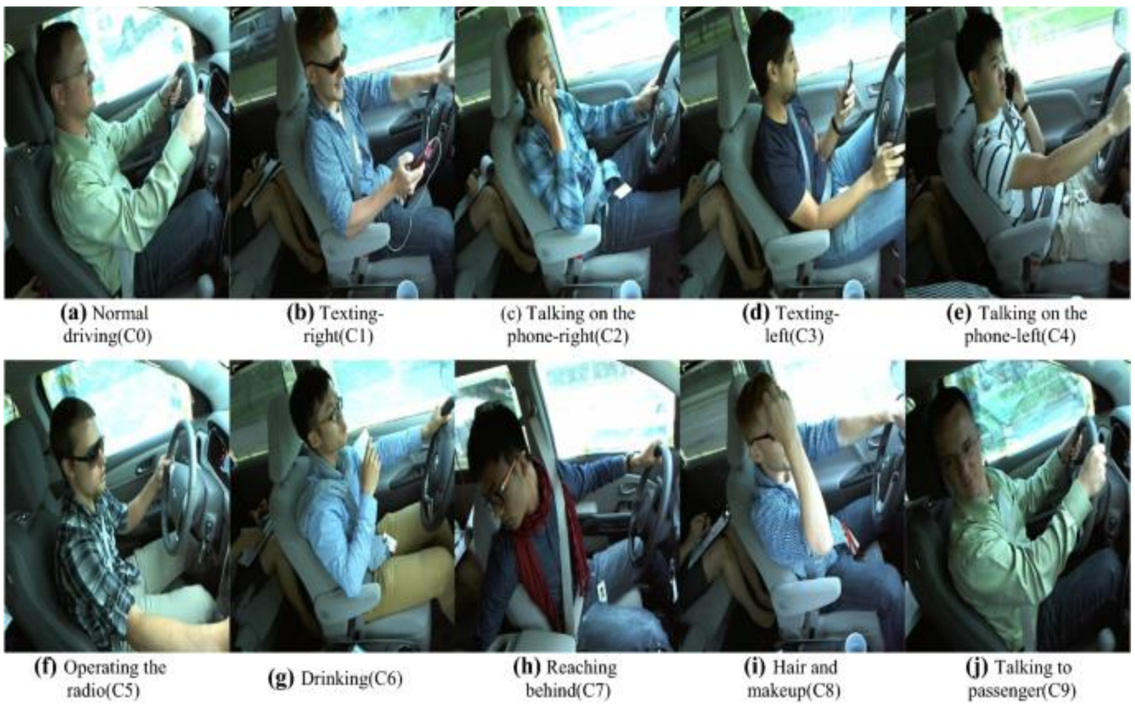


Figure 4: Dataset Classes

3.2 Problem Detection

Over the last decade, there has been a greater emphasis on driver tiredness and the dangers of driving when sleepy or distracted. Driver weariness (or sleepiness) and distraction are frequently used interchangeably (Dinges, 1995). Fatigue is defined as an incapacity or disinclination to continue an activity, usually because the activity has been ongoing for "too long" (Brown, 1994). Sleepiness is one of the key subcomponents of fatigue, which is frequently regarded a generic term. Even though they are connected to the same concept, they should usually be treated separately.

However, there are still issues to address. Individual differences between drivers, both when alert and especially when driving while sleep deprived, are among the most critical. Another important issue is the lack of a reference method (ground truth) that may be used in an artefact-prone environment. The automobile industry is interested in developing driver assistance systems for tired drivers, and the indicators used must be captured using discreet, dependable, and cost-effective sensors. Although progress in this area is rapid, sensors are still in their infancy.

3.2.1 Using Digital Image Processing

Several different measures and indicators of sleepiness are described in the literature. Physiological measures such as EEG and EOG are often used in research, but they are not feasible for commercial use because they are too obtrusive or impractical. Camera-based systems can provide several measures of sleepiness, for example blink behaviour and nodding, and they are frequently reported in literature. Camera-based detection systems are suitable for driving and there are several commercially available devices. Another type of indicators is driving behaviour measures, including e.g. lateral position, steering wheel movements and speed. Context information, such as time of the day and trip duration, have also been suggested as indicators of sleepiness (Boverie, Giralt et al., 2008).

The OpenCV module designed in python is used along with the camera to constantly monitor the driver and extract useful data by processing facial features. Some common factors that the system looks for are, eye activity (camera), blinking frequency, blink duration, eyelid distance, head/face activity, nodding frequency, face position, yawn frequency, facial actions as well as, to some extent, contextual information such as trip duration, time of the day.

3.3 Output Shown on GUI and sound feedback

After that the problems are found via camera and using practises of Digital Image Processing. The next step is to visually notify the user and to produce sounds so that the driver can be warned about the potential hazard in case he is not looking at the screen.

The final solution that we have adopted to caution the driver is using Graphical User Interface to show a pop-up int form of red warning "ALERT" and to produce a variety of sound which depends on the type of problem detected and is generated by using PyGame module designed in Python programming language.

The GUI of the system is designed in Tkinter and it consists of various windows that are displayed at various stages of drivers journey. What is Tkinter you ask? It is the Python interface for the Tk GUI toolkit. It is the typical Python GUI library. GUI applications may be made quickly and simply using Python and Tkinter. An effective object-oriented interface for the Tk GUI toolkit is provided by Tkinter.

First, the user is greeted by a welcome window. Where he must either login or register. After the driver logins he is presented to the main screen or the primary screen of the application and here he has various options, two of them being start and exit. If the user presses on start he is taken to the third window where the interface opens with a video feed of the driver being displayed via dashcam along with OpenCV features. The face of the driver is displayed in a box on the screen that highlights the important aspects of the face like essential facial features. It also signals the driver that now he is being monitored and continuously observed for signs of dizziness, drowsiness, sleepiness, tiredness, absence of safety gears, usage of distractors and not paying attention at the road. In this window he can be shown many popup windows depending on the situations and hazardous situations that may arise. After this when the driver completes his journey and exits the application, he is presented with a final screen that shows various interesting/essential statistics about the trip and also prompts for the option to save useful data. You have two options when using PyGame: Sounds or Music. When you call for music to play or sounds to play, they just begin to do so in the background. It offers a combination to manage all sound-related tasks. You may stream mp3, ogg, and other types of audio using the music sub-module.

3.3.1 Ways to create GUI and use of sound type

There were many ways the GUI of the system could be designed but Tkinter was used solely for its simplicity and because it is a part of python, nothing extra to download. Very simple syntax. The text widget is remarkably powerful and very easy to work with. The canvas widget is also very easy and powerful. No other toolkit provides the same mix of ease and power than those two widgets. Uses native widgets on the mac and windows. Tk is rock solid with few cross-platform idiosyncrasies. Tkinter's binding mechanism is far superior to wxPython's; it's more flexible and consistent. Tkinter's three geometry managers - pack, place, and grid - are much more powerful and easy to use than the wxPython sizers.

In case of PyGame, it is the "standard" library for writing games in Python. It is the most popular and has been around the longest and has the most tutorials. The mixer module has a limited number of channels for playback of sounds. Usually, programs tell pygame to start playing audio and it selects an available channel automatically. The default is 8 simultaneous channels, but complex programs can get more precise control over the number of channels and their use.

All sound playback is mixed in background threads. When you begin to play a Sound object, it will return immediately while the sound continues to play. A single Sound object can also be actively played back multiple times.

People with hearing loss have difficulty hearing and understanding speech. Despite significant advances in hearing aids and cochlear implants, these devices are frequently not enough to enable users to hear and understand what is being communicated in different settings.

So, it also plays a role in Hearing Assistive Technology (HAT).

3.4 Decision making

The main part of Road Accident Avoiding System (RAAS) that helps in processing visual input and making decisions is the Machine Learning (ML) algorithm. The algorithm used is a classifier that is first trained by using a training dataset along with test dataset that is used to calculate the accuracy of problem solving. An algorithm known as a classifier in machine learning automatically arranges or categorises data into one or more of a set of "classes."

Automating processes that required manual labour in the past is made easier by machine learning algorithms. They can significantly cut costs and increase productivity for enterprises. Different ANNs were applied to either measure tiredness or forecast when a motorist will get intoxicated. The best models made use of data on eyelid closure, gaze and head movements, and driving time. These models had the highest percentages of successful detection or prediction. Since the model can anticipate when the driver's state will become impaired to within 5 minutes, performance on prediction is quite promising.

Additionally, modelling drowsiness as a continuum can result in more accurate detection systems that provide results that go beyond just determining if the driver is awake or asleep. Recurrent neural networks or dynamic neural networks could be used to provide the model temporality in the future, as well as the addition of other elements like context information (traffic, type of road, weather etc.).

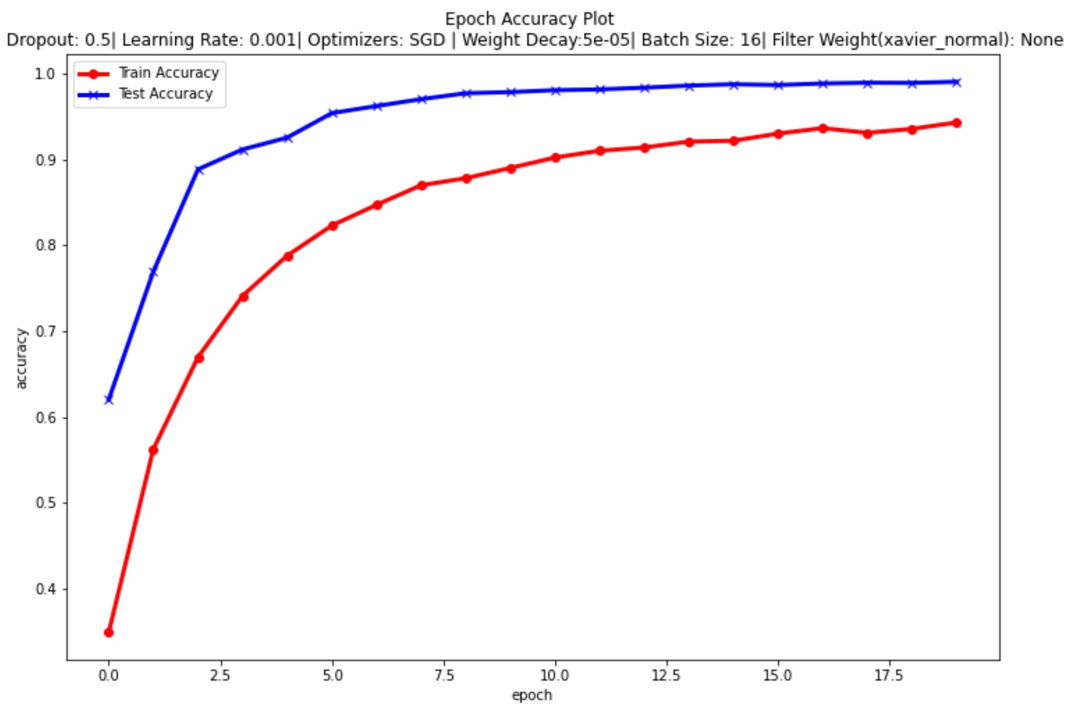


Figure 5: Model Epoch accuracy

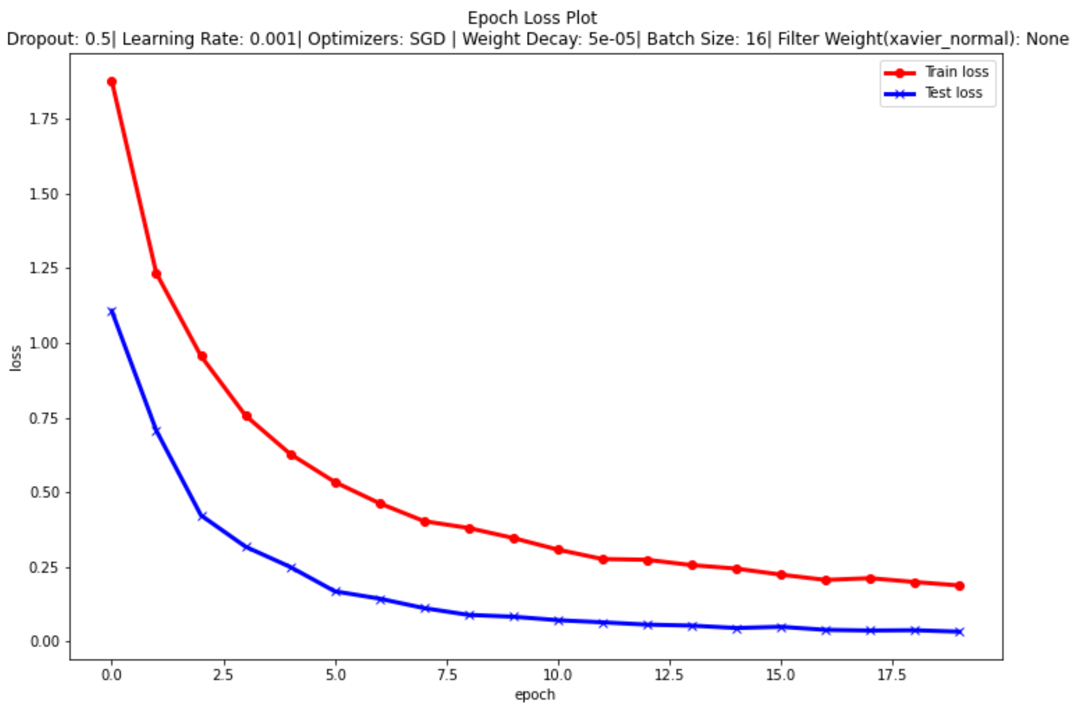
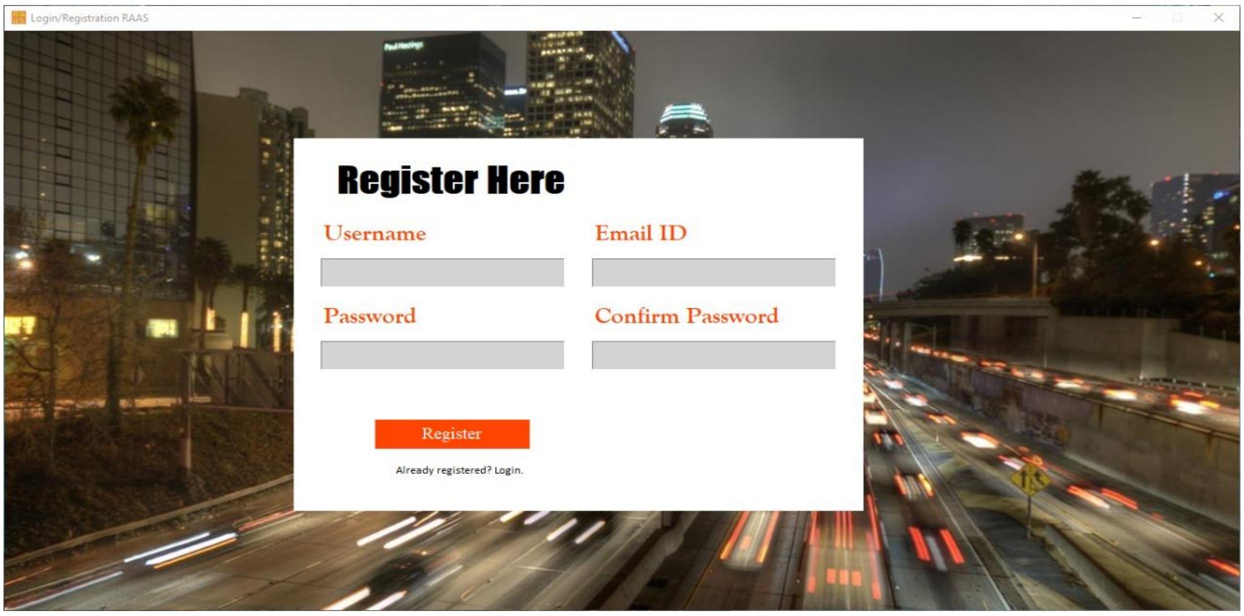


Figure 6: Model Epoch loss

3.4.1 Working of GUI

Instead of text-based UIs, written command labels, or text navigation, the graphical user interface (GUI) enables people to interact with electronic devices through graphical icons and auditory indicators like primary notation.

The Road Accident Avoiding System (RAAS) is a real-time driver alertness monitoring computer vision-based tool with Graphical User Interface (GUI). The Multi-Layer Perceptron (MLP), which is used to measure PERCLOS, detects the eye state (open or closed) in each frame to reliably determine the driver awareness level (percentage of time eyelids are close). A webcam with passive illumination and a consumer-grade computer are used to develop the application. The GUI, which was created using the Tkinter module, makes it simple to set up monitoring parameters, view the results of driver monitoring in real time, and access log files for earlier monitoring.



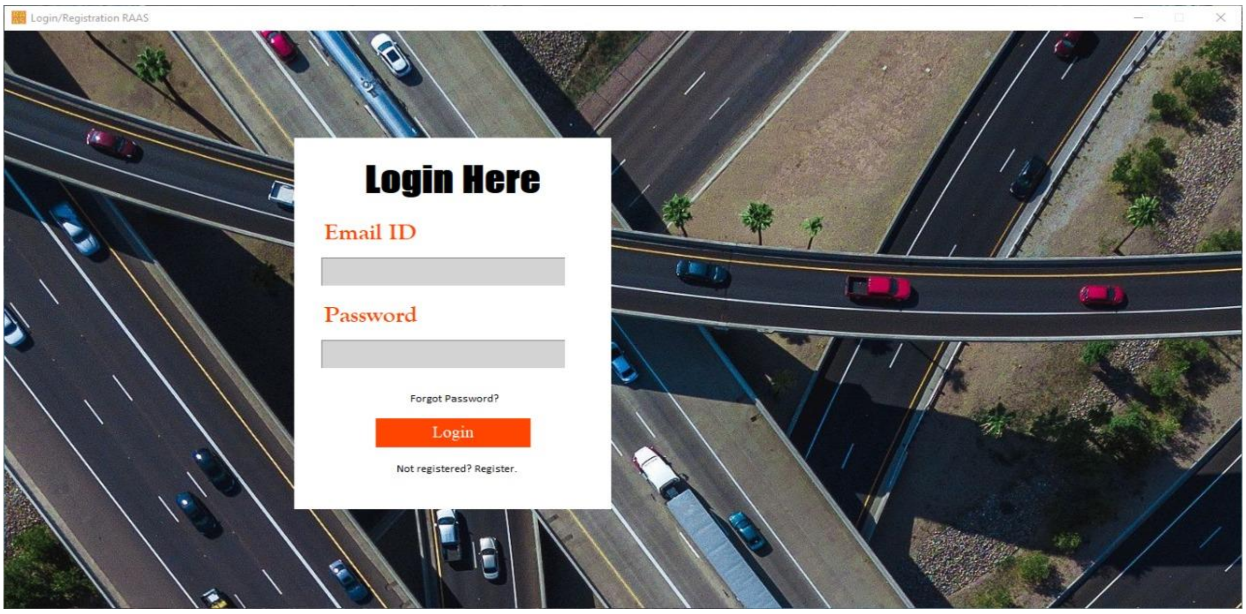


Figure 7: System login and register

Use Case Description

Register

Use Case: Register
Actors: User / Driver
Use Case Description: User can create an account by adding their personal information.
Normal Flow: • User opens the application. • User clicks the register button. • User enters information and registers.
Alternate Flow: • An error message is displayed if some error occurs.
Preconditions: User should have executed the application. A registration form should be displayed to the user. User should have an email address.
Postconditions: User is signed up to application and information is stored in database server. The welcome screen is shown.
Includes: N/A
Extends: N/A

Login

Use Case: Login
Actors: User / Driver
Use Case Description: Once the account is created the user can login to their account Anytime.
Normal Flow: • User opens the application. • User clicks the login option. • User enters username and password and clicks login button.
Alternate Flow:

• If incorrect username or password is entered show an error message and login will not be successful.
Preconditions: User should have executed the application. A login form should be displayed to the user. A user must also already be signed up.
Postconditions: If credentials are correct, welcome screen will be rendered.
Includes: N/A
Extends: N/A

Start System

Use Case: Start System
Actors: User / Driver
Use Case Description: The application is started using this operation by the logged in driver.
Normal Flow: • User opens the application. • User clicks the login button and then logs in. • User presses start.
Alternate Flow: • An error message is displayed if application is not able to access camera.
Preconditions: User should have clicked start operation. A webcam needs to be connected.
Postconditions: Facial geometrics will be shown. Computer Vision tools will be shown.
Includes: N/A
Extends: N/A

Capture Real Time Video

Use Case: Capture Real Time Video

Actors: Camera
Use Case Description: Camera will capture the video of user.
Normal Flow: • User logs into the application. • User clicks on the start application. • Video feed is shown along with face, eyes, chin etc. detection.
Alternate Flow: • An error message is displayed if some error occurs.
Preconditions: User should have executed the application. A user must be logged in.
Postconditions: The application will be ready for Computer Vision applications.
Includes: N/A
Extends: N/A

Extract Features

Use Case: Extract Features
Actors: Camera
Use Case Description: Camera will be used for detecting facial features.
Normal Flow: • User logs into the application. • User clicks on the start application. • Video feed is shown along with face, eyes, chin etc. detection.
Alternate Flow: • If camera is not accessible the program will close.
Preconditions: User should be logged in.
Postconditions:

The driver can imitate safe and monitored driving. The display panel will show useful information.
Includes: N/A
Extends: N/A

Access System for checking all cases

Use Case: **Access System for checking all cases**
Actors: Camera
Use Case Description: The central system or algorithm will be performing various tasks to contribute towards the whole working of software.
Normal Flow: • User clicks on the start option and is now being monitored.
Alternate Flow: • An error message is displayed if some error occurs.
Preconditions: User should have executed the application. User should be logged in.
Postconditions: Various functionalities are imposed.
Includes: N/A
Extends: N/A

Violation of Safety Gears

Use Case: Violation of Safety Gears
Actors: Camera
Use Case Description: The system detects if the essential gears are not worn by the driver.
Normal Flow: • User starts the main program. • User starts driving without wearing safety gears like seat belt etc.
Alternate Flow: • An error message is displayed if some error occurs.
Preconditions: User clicks on the start option and is now being monitored.

Postconditions: User is alerted.
Includes: N/A
Extends: N/A

Drowsiness

Use Case: Drowsiness
Actors: Camera
Use Case Description: The system detects if the driver is drowsy.
Normal Flow: • User starts the main program. • User starts driving while being drowsy, feeling tired or dizziness.
Alternate Flow: • An error message is displayed if some error occurs.
Preconditions: User clicks on the start option and is now being monitored.
Postconditions: User is alerted.
Includes: N/A
Extends: N/A

Distraction

Use Case: Distraction
Actors: Camera
Use Case Description: The system detects if the driver is distracted.
Normal Flow: • User starts the main program. • User starts driving and is detected looking extremely left and right and not paying attention on the main road.
Alternate Flow: • An error message is displayed if some error occurs.
Preconditions:

User clicks on the start option and is now being monitored.
Postconditions: User is alerted.
Includes: N/A
Extends: N/A

Alarm Start

Use Case: Alarm Start
Actors: Alert System
Use Case Description: The system alerts and warns the user via speaker by producing sound.
Normal Flow: • User is driving. • User is being monitored. • The driving state of the driver poses self-harm or danger.
Alternate Flow: • An error message is displayed if some error occurs.
Preconditions: A problem has been detected that may result in potential accident of the driver and loss of life may be imminent.
Postconditions: User is alerted via loud sound.
Includes: N/A
Extends: N/A

Google Map API Integration

Use Case: Google Map API Integration
Actors: Alert System
Use Case Description: The system uses Application Programming Interface of Google Maps for enhancing the security of driver.
Normal Flow: • User starts the main program.

• User is being monitored on the display panel. • The route of the vehicle is kept track of and useful information is retrieved from Google Maps.
Alternate Flow: • An error message may pop up if internet connectivity is not found.
Preconditions: User clicks on the start option and is now being monitored.
Postconditions: The track of vehicle is recorded.
Includes: N/A
Extends: N/A

3.5 Proposed Diagrams

3.5.1 UML Class Diagram

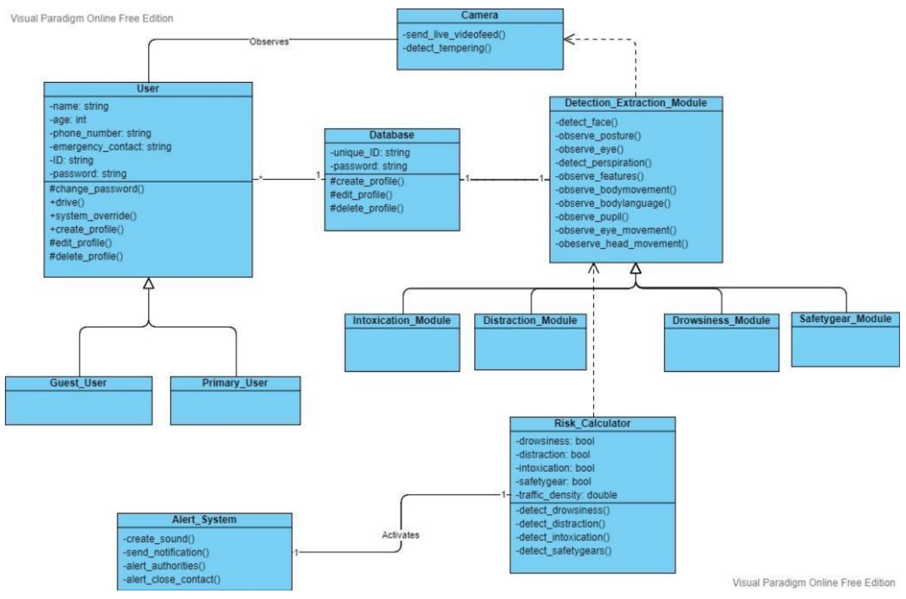


Figure 8: UML Class diagram

3.5.2 UML Use Case Diagram

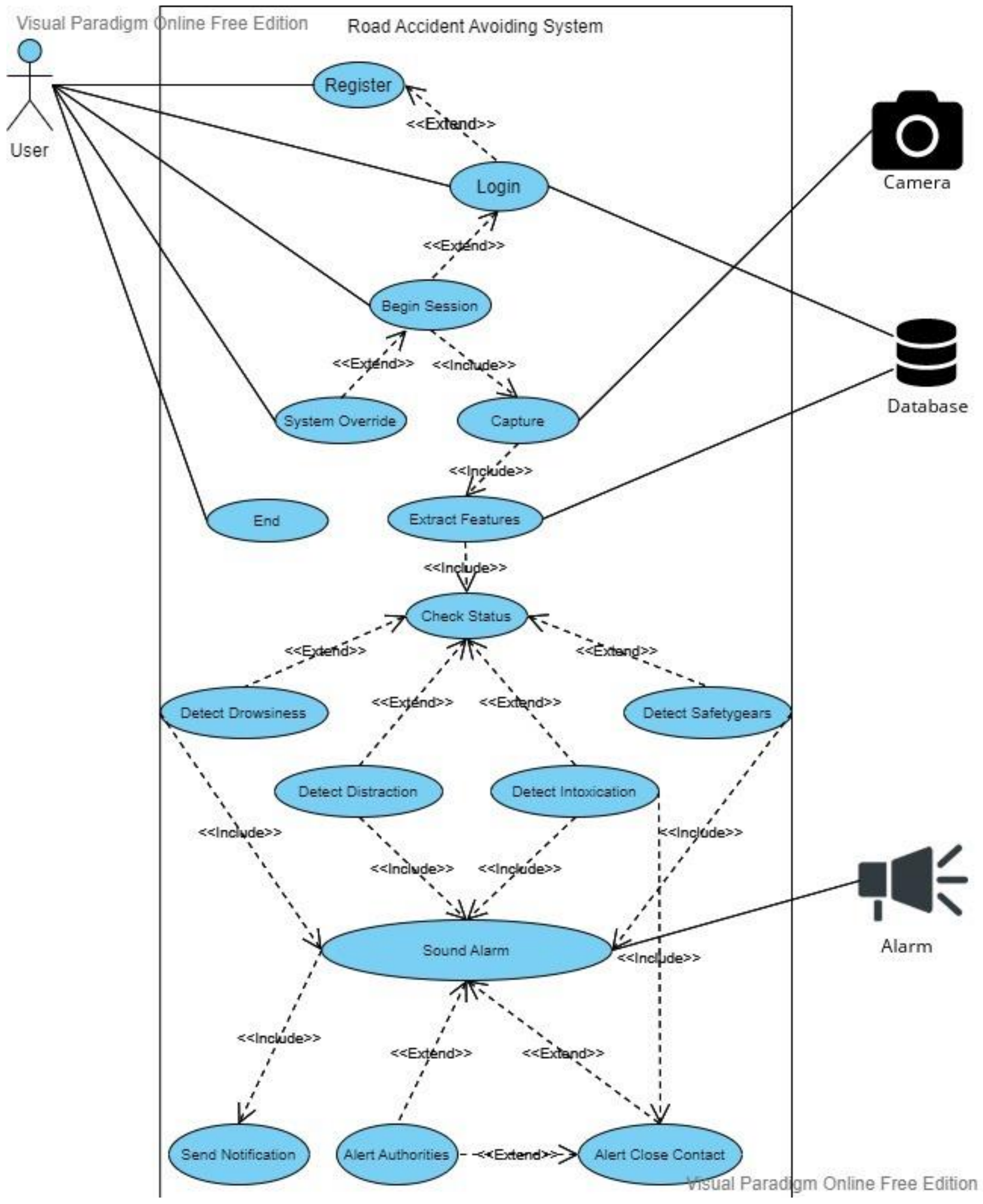


Figure 9: Use Case diagram

3.5.3 Block Diagram

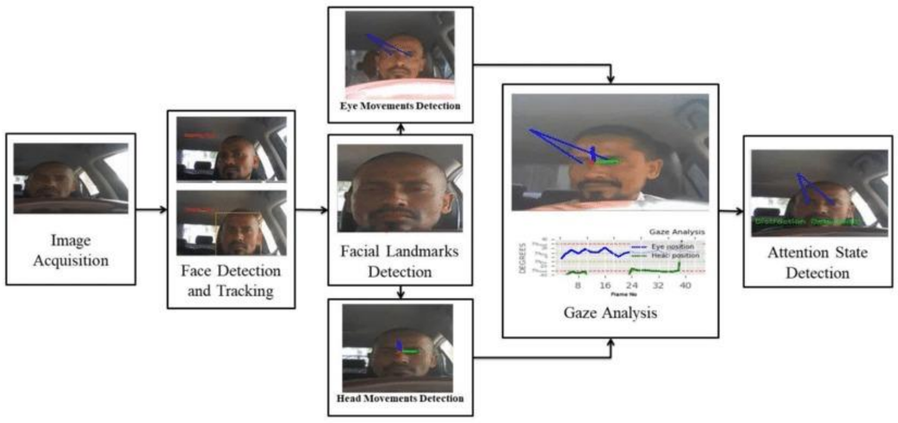


Image Acquisition
Face Detection and Tracking
Eye Movements Detection
Facial Landmarks Detection
Head Movements Detection
Gaze Analysis
Attention State Detection

Figure 10: Block diagram

Chapter 3 Summary

In Chapter 3 we went over the problem from a technical standpoint. It provided a briefing into Driver detection, the dataset, the detection of at-risk-state, the use of Digital image processing, the output, how those decisions were made in the background, The GUI and several diagrams distinctively expressing how the system works as a complete unit as well as designating roles of each actor. Now that we have a knowledge of the technicalities let’s bring in the code.

Chapter 4: Code Analysis and Evaluation

It is very important for any software that the code not only meets the written standards but also carries out the intended purposes with the least number of complications, the least number of lines of code, and in matters such as these where accidents can happen in a split second, it must be efficient time-consumption wise. In the following discourse we'll investigate the in-depth analysis of the code.

4.1 Code Analysis

The analysis module uses a recurrent and convolutional neural network to estimate the drowsiness and distraction level of the driver. The CNN is based on the VGG16 architecture, which presents a lightweight model that is highly precise for distraction detection. The model is very specific for image processing techniques as this project uses image processing to monitor the driver. Since the differences in accuracy are not significant in our domain when upgrading to a superior model, we consider VGG16 to be the most adequate model for this case, where the model needs to quickly obtain a prediction. Using previously trained weights that perform exceptionally well at identifying objects on photos from the needed dataset, we can transfer learning to this model in this way. Due to its extremely similar architecture, VGGNet-16, which has 16 convolutional layers, is also well-liked. It has several filters but only 3x3 convolutions, like AlexNet. On 4 GPUs, it may be trained in two to three weeks. The public currently favours this option the most for removing characteristics from photographs. The VGGNet weight configuration is openly accessible and used as a fundamental feature key in numerous different applications and challenges.

However, managing the 138 million parameters in VGGNet can be a little difficult. Transmitting Reading will result in VGG.

Once the model has been pre-trained in the database, the parameters have been updated for greater accuracy, and the parameter values can be used.

VGG16 has 16 layers.

1. 64 filters are used in convolution.
2. For high integration, convolution employs 64 or more filters.
3. Convolution makes use of 128 filters.
4. Convolution employs 128+ sophisticated filters.
5. The 256 filters used in convolution
6. Convolution makes use of 256 filters.
7. Using 256+ filters during conversion for maximum integration
8. Convolution makes use of 512 filters.
9. The 512 filters used in convolution
10. For sophisticated integration, convolution employs 512+ filters.
11. 512 filters are used in convolution
12. 512 filters are used in convolution.
13. For sophisticated integration, convolution employs 512+ filters.
14. Completely linked with 4096 nodes
15. Completely linked with 4096 nodes
16. 1000 notes in the Softmax output layer.

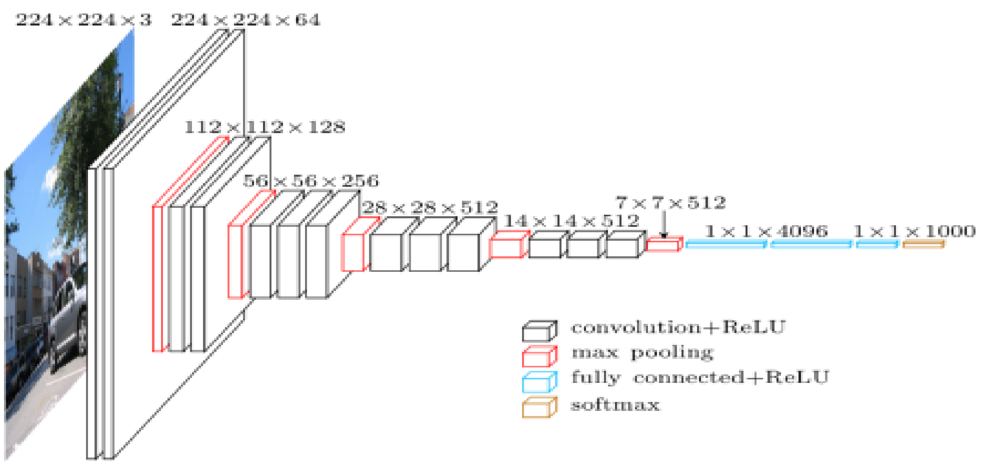
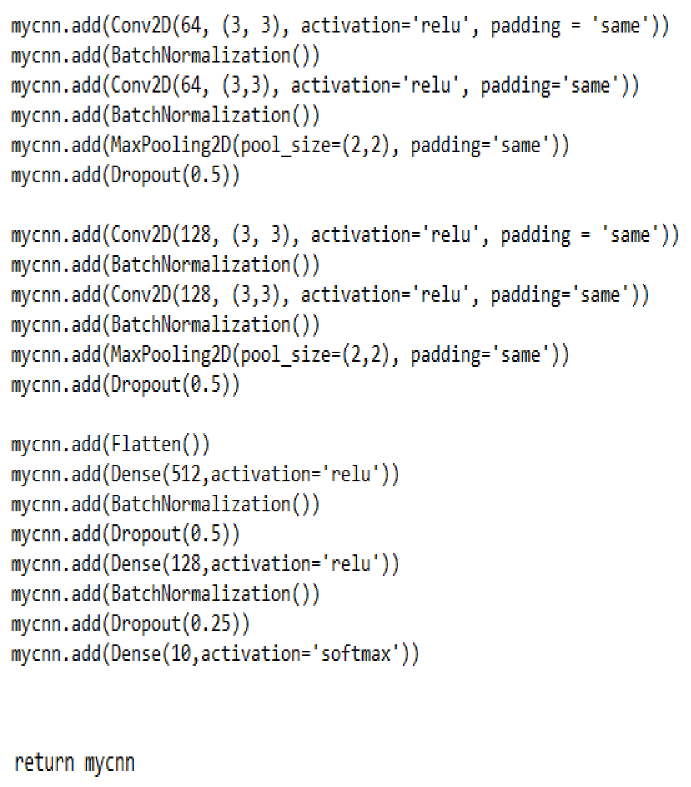


Figure 11: VGG16 architecture

```
def myCNNArchitecture(weights=None):
    mycnn = Sequential()

    mycnn.add(Conv2D(32, (3, 3), activation='relu', padding = 'same', input_shape=(64,64,1)))
    mycnn.add(BatchNormalization())
    mycnn.add(Conv2D(32, (3,3), activation='relu', padding='same'))
    mycnn.add(BatchNormalization())
    mycnn.add(MaxPooling2D(pool_size=(2,2), padding='same'))
    mycnn.add(Dropout(0.5))

    mycnn.add(Conv2D(64, (3, 3), activation='relu', padding = 'same'))
    mycnn.add(BatchNormalization())
    mycnn.add(Conv2D(64, (3,3), activation='relu', padding='same'))
    mycnn.add(BatchNormalization())
    mycnn.add(MaxPooling2D(pool_size=(2,2), padding='same'))
    mycnn.add(Dropout(0.5))

    mycnn.add(Conv2D(128, (3, 3), activation='relu', padding = 'same'))
    mycnn.add(BatchNormalization())
    mycnn.add(Conv2D(128, (3,3), activation='relu', padding='same'))
    mycnn.add(BatchNormalization())
```



```
    mycnn.add(Conv2D(64, (3, 3), activation='relu', padding = 'same'))
    mycnn.add(BatchNormalization())
    mycnn.add(Conv2D(64, (3,3), activation='relu', padding='same'))
    mycnn.add(BatchNormalization())
    mycnn.add(MaxPooling2D(pool_size=(2,2), padding='same'))
    mycnn.add(Dropout(0.5))

    mycnn.add(Conv2D(128, (3, 3), activation='relu', padding = 'same'))
    mycnn.add(BatchNormalization())
    mycnn.add(Conv2D(128, (3,3), activation='relu', padding='same'))
    mycnn.add(BatchNormalization())
    mycnn.add(MaxPooling2D(pool_size=(2,2), padding='same'))
    mycnn.add(Dropout(0.5))

    mycnn.add(Flatten())
    mycnn.add(Dense(512,activation='relu'))
    mycnn.add(BatchNormalization())
    mycnn.add(Dropout(0.5))
    mycnn.add(Dense(128,activation='relu'))
    mycnn.add(BatchNormalization())
    mycnn.add(Dropout(0.25))
    mycnn.add(Dense(10,activation='softmax'))


    return mycnn
```

Figure 12: CNN model

For drowsiness we use haarcascade and shape predictor 68 landmark. The shape predictor is a tool that replaces an image that contains an object and pulls out a set of points that define the shape of the object. We then passed the outline and the size of the obtained rectangle. The identified landmarks are then displayed on a frame using cv2 circle.

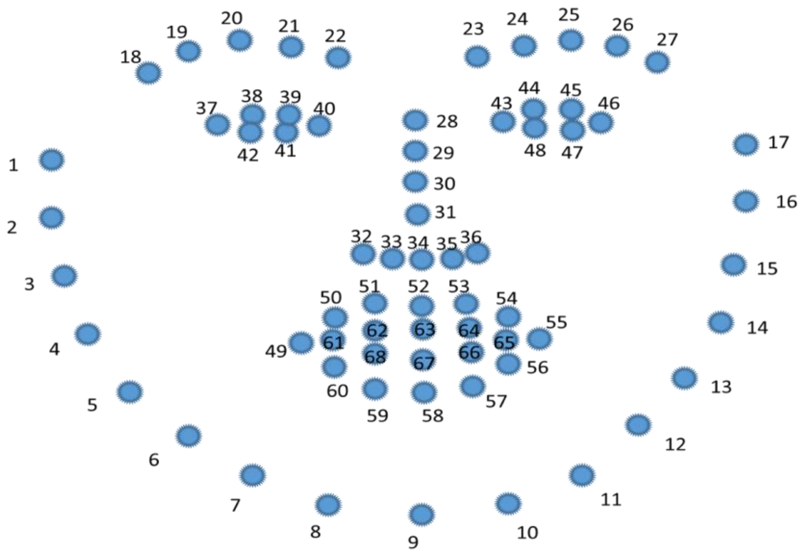


Figure 13: shape_predictor_68_face_landmarks

4.1.1 Software details:

Software	**Description**
Operating System	Windows 10 Windows 11 Windows 10 and 11 for its best support, user-friendliness and compatibility with python 3. As for Linux, Ubuntu can be used as well.
Tools and Libraries	

	Python3: Python 3 is more popular and has a type system, hence the most recent Python interpreter will be utilised. The print function in Python 2 employs an antiquated syntax. Python 3 is the current standard, while Python 2 is still used in DevOps for configuration management. Tkinter: Tkinter is Python's default GUI library. GUI applications may be made quickly and simply using Python and Tkinter. An effective object-oriented interface for the Tk GUI toolkit is provided by Tkinter. NumPy: NumPy is a library for the Python programming language that adds support for big, multidimensional arrays and matrices as well as a huge selection of high-level mathematical functions to work on these arrays. SciPy: SciPy is a Python library that is available for free and open source and is used for technical and scientific computing. SciPy includes modules for a

	variety of common tasks in science and engineering, including optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, and ODE solvers. Pygame: A cross-platform collection of Python modules created specifically for creating video games. It provides music and graphics libraries created specifically for use with the Python programming language. In essence, sound will be played for audio feedback. For various situations/alerts, multiple sounds can be employed. Dlib: Written in the computer language C++, Dlib is a general-purpose, cross-platform software library. Its design is significantly influenced by concepts from component-based software engineering and design by contracts. Consequently, it is first and mainly a collection of separate software components. Imutils: A collection of useful functions for OpenCV and Python that make it simpler to perform common image processing tasks including translation, rotation, scaling, skeletonization, and presenting Matplotlib pictures.

	OpenCV Real-time computer vision is the primary focus of OpenCV, a library of programming tools. It was initially created by Intel and then backed by Willow Garage and Itseez.

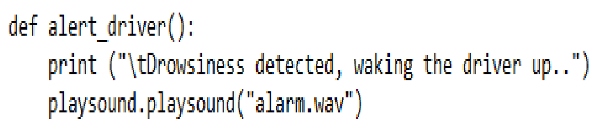
4.2 Alarm Activation

The following conditions must be met to alert the driver:

- The driver is considered distracted if he will be according to dataset distraction classes.

  The driver is considered to be drowsy at a specific moment when the output of the analysis module is greater than a threshold, called drowsiness threshold.

- The driver has to be considered drowsy for multiple instants of the last 60 s to raise an alarm. This is determined by the variable, which represents how many seconds the driver has to be drowsy before alerting him or her. This value ranges between 0 and 60. This way, when the driver is considered drowsy for at least the established minimum time, an alarm is raised. If the condition to raise an alarm continues after alerting the driver, a second alarm is not raised. Instead, another alarm is raised only if the conditions for alerting the driver are no longer met, and after that, drowsiness is detected again.



```
def alert_driver():
    print ("\tDrowsiness detected, waking the driver up..")
    playsound.playsound("alarm.wav")
```

Figure 14: Alarm activation

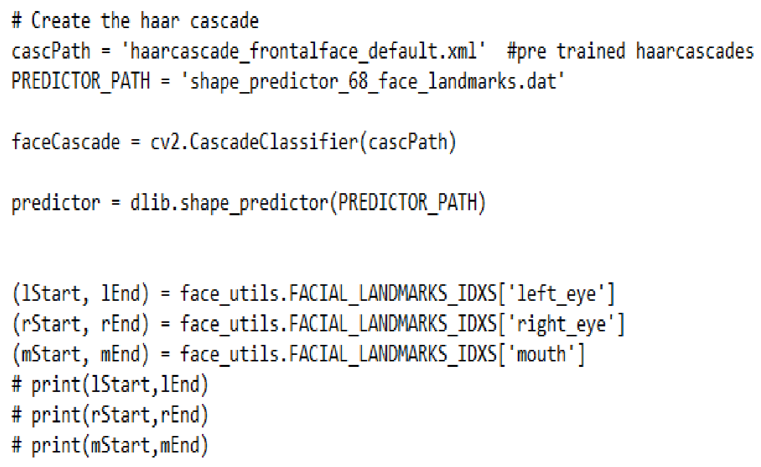
4.3 Preprocessing

The parameters that we are interested in discovering from the driver's image are the following:

1. Eye aspect ratio.
2. Mouth aspect ratio.
3. Threshold.
4. Number of yawns (yawns).
5. Time spent yawning (avg_yawn).
6. Learnig rate.
7. Bath size.

Finding the driver's face is the initial step in calculating these characteristics. DLIB's face detector is once more employed for this. The shape predictor uses an ensemble of regression trees in this instance, but we also use the landmark detector that DLIB offers.

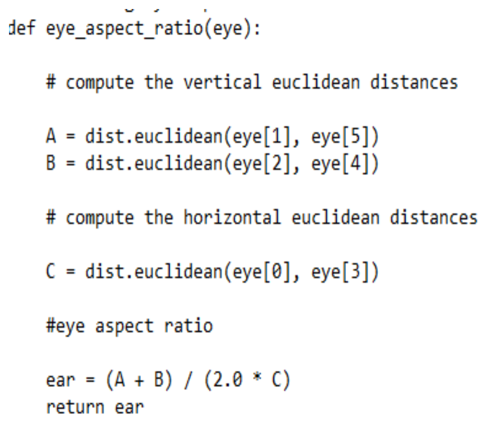
We can determine the location of the driver's eyes thanks to DLIB's landmark detector, which depicts facial features with a collection of 68 points, 12 of which are the eyes (6 points per eye). It is required to first determine how open or closed the driver's eyes are in order to calculate the factors linked to eyes and blinks.



```
# Create the haar cascade
cascPath = 'haarcascade_frontalface_default.xml'  #pre trained haarcascades
PREDICTOR_PATH = 'shape_predictor_68_face_landmarks.dat'

faceCascade = cv2.CascadeClassifier(cascPath)

predictor = dlib.shape_predictor(PREDICTOR_PATH)


(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS['left_eye']
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS['right_eye']
(mStart, mEnd) = face_utils.FACIAL_LANDMARKS_IDXS['mouth']
# print(lStart,lEnd)
# print(rStart,rEnd)
# print(mStart,mEnd)
```

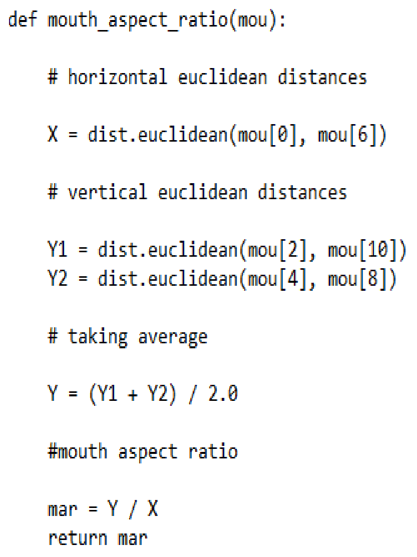
Figure 15: Haarcascade and shape predictor

To do this, we measure the distance between the top and bottom eyelids and divide that measurement by the eye width to acquire the eye aspect ratio (EAR), which gives us the openness values, which typically range from 0.16 to 0.36. Experimentally, a threshold of 0.20 has been established, meaning that whenever the measured EAR is less than 0.20, it is assumed that the driver has closed their eyes. The first three measurements are obtained by counting how frequently and how long the driver blinks. We then downgrade the quality of these photographs. Next, we want to utilise the driver's face to see if they are yawning. The driver's face is pre-processed and cropped in order to do this. The CNN we have trained to determine whether a person is yawning or not receives this image as an input.



```
def eye_aspect_ratio(eye):

    # compute the vertical euclidean distances

    A = dist.euclidean(eye[1], eye[5])
    B = dist.euclidean(eye[2], eye[4])

    # compute the horizontal euclidean distances

    C = dist.euclidean(eye[0], eye[3])

    #eye aspect ratio

    ear = (A + B) / (2.0 * C)
    return ear
```

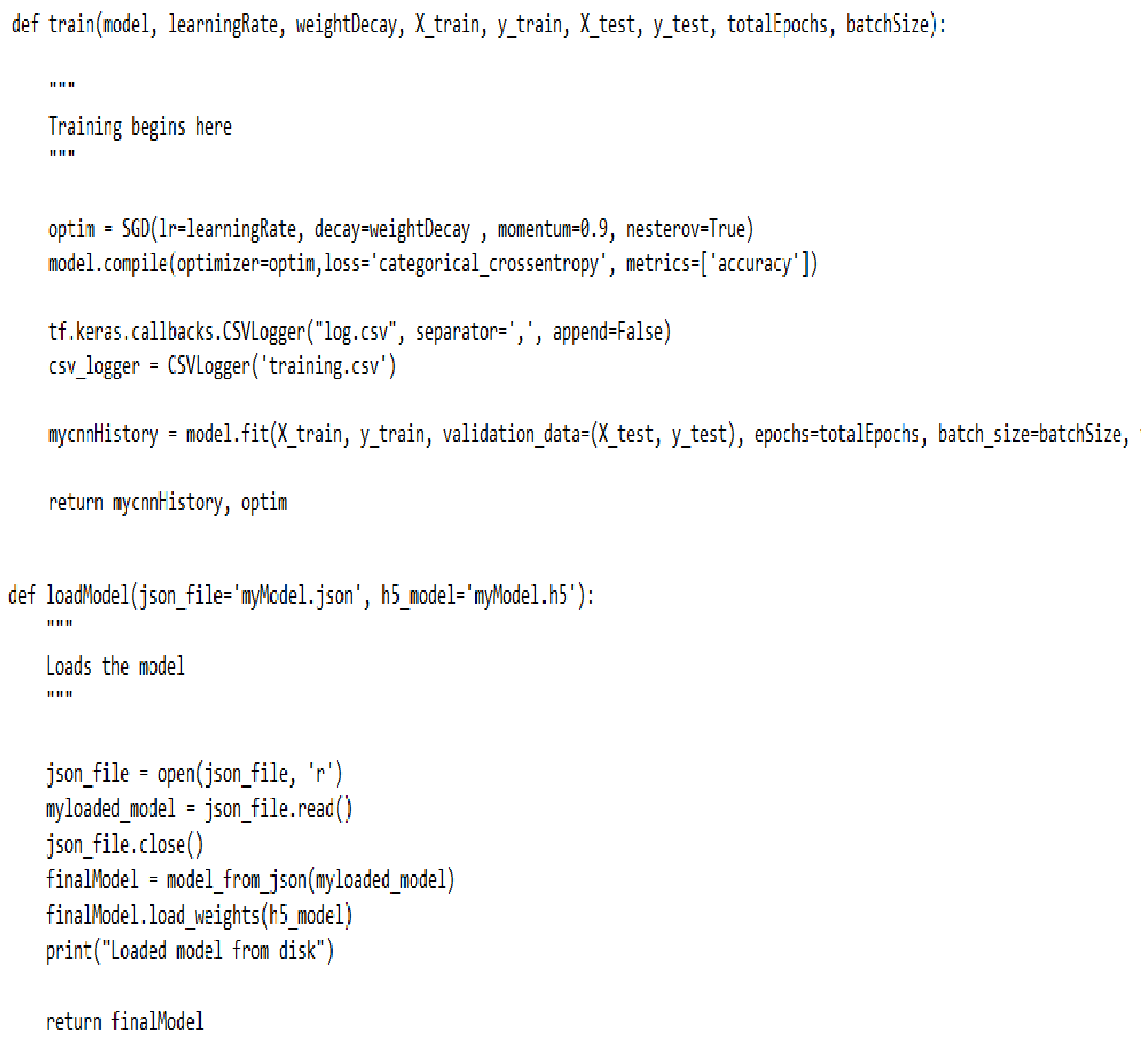
Figure 16: Calculating EAR



```
def mouth_aspect_ratio(mou):

    # horizontal euclidean distances

    X = dist.euclidean(mou[0], mou[6])

    # vertical euclidean distances

    Y1 = dist.euclidean(mou[2], mou[10])
    Y2 = dist.euclidean(mou[4], mou[8])

    # taking average

    Y = (Y1 + Y2) / 2.0

    #mouth aspect ratio

    mar = Y / X
    return mar
```

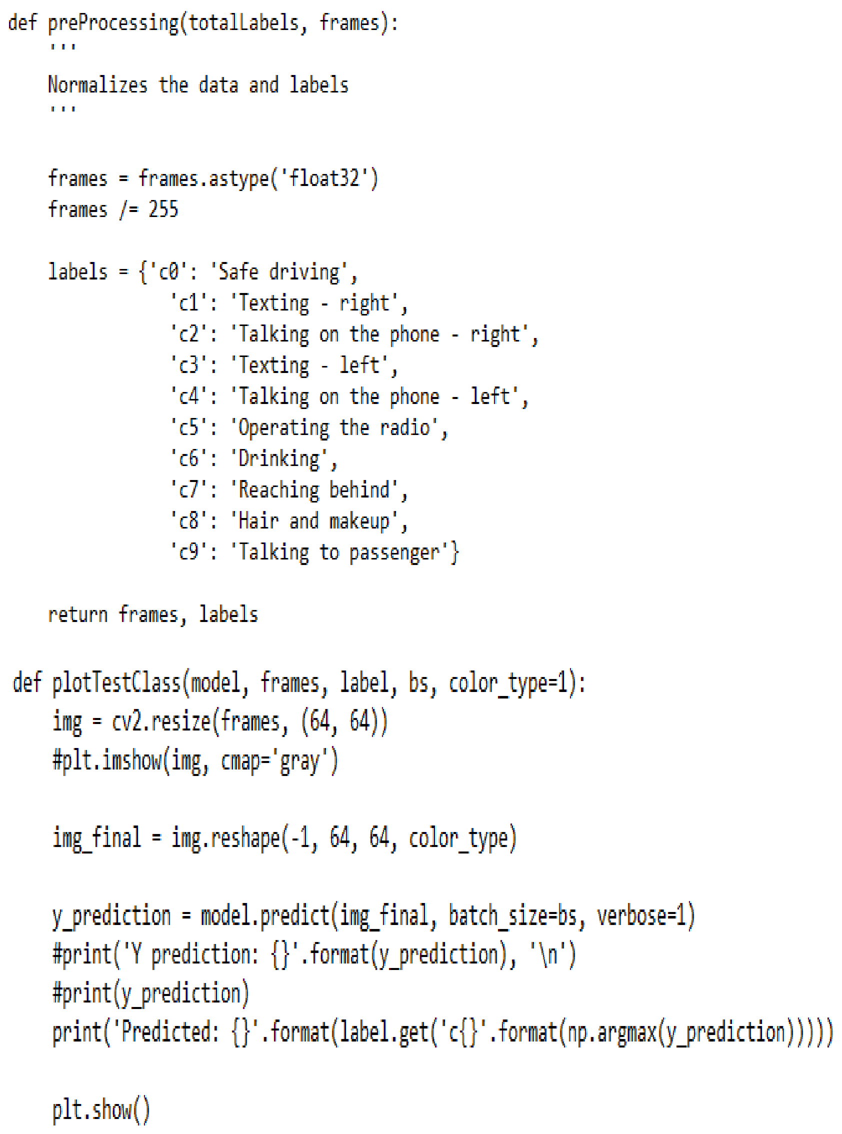
Figure 17: Calculating MAR

To train this neural network, the Distracted driver dataset is used, which provides images from 30 different people yawning and driving normally. In particular, we use the videos that were recorded by a camera placed on the dashboard, in front of the driver. Videos from 18 people are used at the training phase, while 6 are used for validation and the other 6 are used for testing. The accuracy when classifying the drivers' yawns reached a notable 88% accuracy on the testing data used. To avoid possible errors, to classify a frame, the system calculates the average class of the last 20 frames (2 s), and it returns the class that appears more times. Thanks to this deep learning model, we can calculate how many seconds the driver has spent yawning.



```
def train(model, learningRate, weightDecay, X_train, y_train, X_test, y_test, totalEpochs, batchSize):

    """
    Training begins here
    """

    optim = SGD(lr=learningRate, decay=weightDecay , momentum=0.9, nesterov=True)
    model.compile(optimizer=optim,loss='categorical_crossentropy', metrics=['accuracy'])

    tf.keras.callbacks.CSVLogger("log.csv", separator=',', append=False)
    csv_logger = CSVLogger('training.csv')

    mycnnHistory = model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=totalEpochs, batch_size=batchSize,

    return mycnnHistory, optim


def loadModel(json_file='myModel.json', h5_model='myModel.h5'):
    """
    Loads the model
    """

    json_file = open(json_file, 'r')
    myloaded_model = json_file.read()
    json_file.close()
    finalModel = model_from_json(myloaded_model)
    finalModel.load_weights(h5_model)
    print("Loaded model from disk")

    return finalModel
```

Figure 18: Training and load model



```
def preProcessing(totalLabels, frames):
    '''
    Normalizes the data and labels
    '''

    frames = frames.astype('float32')
    frames /= 255

    labels = {'c0': 'Safe driving',
              'c1': 'Texting - right',
              'c2': 'Talking on the phone - right',
              'c3': 'Texting - left',
              'c4': 'Talking on the phone - left',
              'c5': 'Operating the radio',
              'c6': 'Drinking',
              'c7': 'Reaching behind',
              'c8': 'Hair and makeup',
              'c9': 'Talking to passenger'}

    return frames, labels

def plotTestClass(model, frames, label, bs, color_type=1):
    img = cv2.resize(frames, (64, 64))
    #plt.imshow(img, cmap='gray')

    img_final = img.reshape(-1, 64, 64, color_type)

    y_prediction = model.predict(img_final, batch_size=bs, verbose=1)
    #print('Y prediction: {}'.format(y_prediction), '\n')
    #print(y_prediction)
    print('Predicted: {}'.format(label.get('c{}'.format(np.argmax(y_prediction)))))

    plt.show()
```

Figure 19: Preprocessing and plotting result

Chapter 4 Summary

We looked at a very detailed examination of the code, the triggers for the alarm (what registers as a danger situation for the system) and had a look at the pre-processing required to carry that out. Overall, chapter 4 ends all the discussion surrounding the project itself and wraps it up quite nicely, now we shall look at this project in terms of its impact and the heights we hope it reaches eventually.

Chapter 5: Conclusion

In this thesis, we discussed a road accident avoiding system which helps driver to be active in all proposed situations by monitoring to driver. Our system uses machine learning and image processing techniques to giver better solution to all problematic situations. The system monitors driver through camera and records the video. This video is used by image processing techniques and then fed to machine learning model to detect driver bod posture and alarmed driver in danger situation. The system monitors all the situations like talking to other person, seeing behind, using the radio, talking on phone and drowsiness. These are the main situations which causes road accidents. Our system handles them very well and alerts the driver. The techniques used by the system are very effective and easy to install dependencies of machine learning and image preprocessing. Our proposed system has good interface and easy to use. It can be used in all types of HV vehicles. Our system is less costly.

Chapter 6: Future Work

Future milestones that need to be achieved to commercialize this project are the following.

6.1 Driver alertness percentage:

Usage as driver alertness percentage.

The main objective of this project is to prevent road accident by monitoring driver. It can also be used as evaluation of activeness of driver by counting the driver alarming. Big automobiles company can use them for their driver to evaluate their performance. It is less costly and can be access its records of driver.

References

Facial-Expression Analysis for Predicting Unsafe Driving Behaviour - Maria E. Jabon, Jeremy N. Bailenson, Emmanuel (Manos) D. Pontikakis, Leila Takayama

https://www.researchgate.net/publication/220299623_Facial-Expression_Analysis_for_Predicting_Unsafe_Driving_Behavior

Emotion Recognition System with Facial Expression System for a Vehicle to Make the Journey Harmless - Haju Mohamed Mohamed Naleer and Narmadi Wathsala Dissanayake

http://www.seu.ac.lk/jisit/publication/v5n1/JISIT5114.pdf

Intelligent Driver Warning System using Deep Learning-based Facial Expression Recognition - S.Suchitra, S.Sathya Priya, R.J. Poovaraghan, B.Pavithra, J.Mercy Faustina

https://www.ijrte.org/wp-content/uploads/papers/v8i3/C4028098319.pdf

Vision-Based Road Rage Detection Framework in Automotive Safety Applications by Alessandro Leone, Andrea Caroppo, Andrea Manni, and Pietro Siciliano

https://www.mdpi.com/1424-8220/21/9/2942/htm