

# **Crash Free- A Smart System to Detect, Prevent Drowsiness and Locate Accidents**

**N. A. Sharaaf  
2021**



# **Crash Free- A Smart System to Detect, Prevent Drowsiness and Locate Accidents**

**A dissertation submitted for the Degree of Master of  
Computer Science**

**N. A Sharaaf  
University of Colombo School of Computing  
2021**






## DECLARATION

I hereby declare that the thesis is my original work, and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis. This thesis has also not been submitted for any degree in any university previously.

Student Name: N. Ahamed Sharaaf

Registration Number: 2018/MCS/003

Index Number: 18440032


 29-11-2021

Signature of the Student & Date

This is to certify that this thesis is based on the work of Mr. /Ms. N.A. Sharaaf under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by,

Supervisor Name: Dr. Hiran Ekanayake

 30.11.2021

Signature of the Supervisor & Date

I would like to dedicate this thesis to my family.

## **ACKNOWLEDGEMENTS**

To commence with this section, it is a grate and kind pleasure to show my gratitude to the institution – University of Colombo School of Computing (UCSC) which gave an opportunity to conduct a study and my supervisor Dr. H E M H B Ekanayake, Senior Lecturer, UCSC. The progressive advice, scholarly knowledge and uplifting words and endless motivation of him is the main reason for competing this study.

I am also thankful to Dr. B H R Pushpananda, Coordinator of the Research Project of Master of Computer Science degree program for his timely suggestions and endless guidance throughout the stages of the project. Also, I thank all the lecturers at the institute.

I also like to thank all open-source communities for extending their support by providing information and guidance whenever need. A grate salute to the participants who volunteered during the research.

Finally, I have the responsibility to thank my family, friends, and teachers for giving continuous encouragement. I take this opportunity to thank everyone who support me to complete this research project successfully.

## ABSTRACT

Due to increased number of vehicles in the present era, there are significant number of problems faced by the drivers. Drowsiness is a major problem among them. This causes many accidents, and those accidents are very dangerous which will affect the economic development of a country. There is a necessity to keep drivers safe from drowsiness while they are driving. Also, it is recommended to keep a tracking mechanism of any driver to locate them easily if they met with an accident. Thus, the proposed system CrashFree focus on both drowsiness prediction and accident detection. This project uses behavioral based drowsiness prediction techniques to identify the state of a driver. Numerous approaches have been used under this technique previously. But this study uses a technique where it has three prediction models which will predict the state of face, eye, and mouth separately. The prediction models are trained with public and private datasets using Convolutional Neural Network (CNN) combining Support Vector Machines (SVM). Initially the video feed of a driver will be captured by a camera and regions of face, eye, and mouth will be extracted separately. Then it will be passed to their own prediction models and each state will be obtained. The final decision whether a driver is drowsy or not will be occupied by combining each separate states. The system will give a continuous alert via mobile application once a drowsy state is identified. On the other hand, the system uses a two-sensor based Internet of Things (IoT) platform to suspect an accident on a vehicle. The vibration sensor and accelerometer sensor will be used along with Arduino UNO. The sensor values will be transferred to the server on real-time. The values will be compared, and an accident suspect prompt alert will be sent to the driver if an accident is suspected. If the driver failed to acknowledge to the alert an automatic notification will be sent to the driver's friends circle, family circles, etc. The drowsiness prediction module was tested with some public and private datasets, and it work well with numerous conditions. Prediction models of eye state, mouth state and face state gave an average accuracy of 96%, 97% and 85% respectively. The overall experiment of the module gave an average accuracy of 93% where the drowsiness prediction lacks in low light environment. The accident detection module cannot be evaluated in the real-world. Therefore, the prototype has been tested based on some conditions. The proposed system CrashFree will make an impact on the respective research area, and it provides a solution to the above-mentioned issues.

***Drowsiness Prediction, Accident Detection, Convolutional Neural Network (CNN), Support Vector Machines (SVM), Internet of Things (IoT), Arduino UNO, Vibration Sensor, Accelerometer***

# TABLE OF CONTENTS

DECLARATION.....	i
TABLE OF CONTENTS .....	v
LIST OF FIGURES .....	viii
LIST OF TABLES .....	x
ABBREVIATIONS .....	xi
CHAPTER 1 - INTRODUCTION .....	1
1.1 Prolegomena .....	1
1.2 Background and Motivation .....	1
1.3 Problem Statement .....	2
1.4 Aim and Objectives .....	3
1.4.1 Aim.....	3
1.4.2 Objectives.....	3
1.5 Scope of the Research.....	3
1.6 Structure of the Dissertation .....	4
1.7 Summary .....	4
CHAPTER 2 - LITERATURE REVIEW .....	5
2.1 Introduction.....	5
2.2 Image Processing and Computer Vision.....	5
2.3 Visual Object Tracking and Detection.....	5
2.4 Facial Landmarks Recognition .....	7
2.5 Internet Of Things (IoT) .....	8
2.6 Drowsiness Detection Methods .....	9
2.7 IoT and Accident Detection .....	13
2.8 Summary .....	15
CHAPTER 3 - METHODOLOGY .....	16
3.1 Introduction.....	16
3.2 Problem Analysis.....	16



3.3 Research Approach of CrashFree .....	17
3.4 Research Design .....	18
3.5 Drowsiness Prediction and Alert .....	19
3.6 Pre-Process and Extraction Module.....	20
3.7 Build Dataset Module .....	22
3.8 Train Model Module .....	23
3.8.1 Define Training Model.....	23
3.8.2 Keras Model Layers .....	24
3.8.3 CrashFree CNN Model.....	26
3.8.4 Model Compilation .....	27
3.8.5 Model Fitting.....	28
3.9 Capture Image Module .....	28
3.10 Classification Module .....	28
3.11 Alert Module.....	30
3.12 Accident Detection and Notification .....	31
3.13 Accident Detection Module .....	31
3.14 Notification Module.....	36
3.15 Integration of Modules.....	36
3.16 Summary .....	37
CHAPTER 4 - EVALUATION AND RESULTS.....	38
4.1 Introduction.....	38
4.2 Evaluation of Drowsiness Detection and Alert.....	38
4.2.1 Datasets .....	38
4.2.2 Evaluation Methods.....	39
4.2.3 Results and Evaluations .....	40
4.3 Evaluation of Accident Detection and Notification.....	52
4.3.1 Experiments and Results .....	53
4.4 Summary .....	57

CHAPTER 5 - CONCLUSION AND FUTURE WORK .....	58
5.1 Introduction.....	58
5.2 Contributions .....	58
5.3 Achievements of Objectives .....	58
5.4 Problems Encountered .....	59
5.5 Limitations .....	60
5.6 Future works .....	60
5.7 Summary .....	61
APPENDICES .....	I
Appendix A: User Interfaces .....	I
Appendix B: Image Extraction Code Snippet.....	VI
Appendix C: Drowsiness Prediction Results .....	VIII
Appendix D: Arduino Platform .....	XII
Appendix E: Send Notification Code Snippet .....	XIII
Appendix F: Participant Information Sheet & Consent Form .....	XIV
Appendix G: Arduino Code Snippet.....	XVI
REFERENCES .....	XXI

## LIST OF FIGURES

Figure 2.1. Overall Architecture of CNN .....	7
Figure 2.2. Example overview of an IoT system.....	9
Figure 2.3. Drowsiness Detection Methods .....	10
Figure 3.1. CrashFree Overall High-level Diagram .....	18
Figure 3.2. Drowsiness Prediction and Alert Module Overview .....	19
Figure 3.3. Dataset Structure and Preprocessing .....	21
Figure 3.4. Final Dataset Structure.....	22
Figure 3.5. CrashFree CNN Architecture.....	26
Figure 3.6. Adding SVM Support to CNN Model .....	27
Figure 3.7. Hinge Loss Function .....	27
Figure 3.8. Flowchart for Classification Module .....	28
Figure 3.9. Classification Module’s Drowsiness Prediction Process .....	29
Figure 3.10. Algorithm for Drowsiness Prediction from an Image.....	30
Figure 3.11. Accident Detection and Notification Module Overview .....	31
Figure 3.12. Arduino Platform High-level Overview.....	32
Figure 3.13. Example of Accelerometer sensor axis.....	32
Figure 3.14. Accelerometer Outputs Respect to Several Movements .....	33
Figure 3.15. Algorithm for Accident Detection and Notification .....	34
Figure 3.16. Algorithm for Suspicious Detection Function .....	35
Figure 3.17. Arduino Platform of Accident Detection Module.....	36
Figure 4.1. Eye State Model Accuracy and Loss .....	40
Figure 4.2. Eye State Model Confusion Matrix.....	41
Figure 4.3. Mouth State Accuracy and Loss .....	42
Figure 4.4. Mouth State Model Confusion Matrix .....	43
Figure 4.5. Face State Accuracy and Loss.....	44
Figure 4.6. Face State Model Confusion Matrix .....	45
Figure 4.7. Average Evaluation Results of Models.....	46
Figure 4.8. Frame of Alert State – 1 .....	47
Figure 4.9. Frame of Alert State - 2.....	47
Figure 4.10. Frame of Alert State with Eyeglasses .....	48

Figure 4.11. Frame of Drowsy State - 1 .....	48
Figure 4.12. Frame of Drowsy State - 2 .....	49
Figure 4.13. Frame of Drowsy State with Eyeglasses .....	49
Figure 4.14. Frame of Alert State in a Low Light Environment .....	50
Figure 4.15. Frame of Alert State with Eyeglasses in a Low Light Environment.....	50
Figure 4.16. Driver Drowsiness Alert via Mobile App .....	51
Figure 4.17. Readings From Sensors .....	53
Figure 4.18. Accident Suspect Alarm to the Driver .....	56
Figure 4.19. Accident Notification to a Close Circle .....	57

## LIST OF TABLES

Table 4.1. Summary of Eye State Model.....	41
Table 4.2. Summary of Mouth State Model .....	43
Table 4.3. Summary of Face State Model .....	45
Table 4.4. Experiment results of drowsiness prediction and alert module.....	52
Table 4.5. Accident Suspect Alarm - Experiment 1 .....	54
Table 4.6. Accident Suspect Alarm - Experiment 2 .....	54
Table 4.7. Accident Suspect Alarm - Experiment 3 .....	55
Table 4.8. Accident Notification - Experiment 1 .....	55
Table 4.9. Accident Notification - Experiment 2 .....	56
Table 4.10. Accident Notification - Experiment 3 .....	56

## ABBREVIATIONS

<b>Abbreviation</b>	<b>Meaning</b>
UCSC	University of Colombo School of Computing
CNN	Convolution Neural Network
SVM	Support Vector Machine
IoT	Internet of Things
MOSSE	Minimum Output Sum of Squared Error
KCF	Kernel Correlation Filter
HOG	Histogram of Oriented Gradient
ANN	Artificial Neural Network
DCNN	Deep Convolutional Neural Network
TCNN	Tweaked Convolutional Neural Networks
GMM	Gaussian Mixture Model
DAN	Deep Alignment Network
MTI	Massachusetts Institute of Technology
IDC	International Data Corporation
ECG	Electrocardiography
EEG	Electroencephalography
MLNN	Multilayer neural network
ANF	Adaptive Neuro Fuzzy
HRV	Heart Rate Variability
ROC	Receiver Operation Curve
PPG	Photoplethysmogram
GPS	Global Positioning System
SVMPPM	SVM based Posterior Probabilistic Model
SWA	Steering Wheel Angles
NB	Naive Bayes
KNN	K-Nearest Neighbors
FEC	Frequency of Eye Closing
ALBP	Advanced Local Binary Pattern
SSIM	Structural Similarity Measure
MCCNN	Multi-task Cascaded Convolutional Neural Networks
KCF	Kernel Correlation Filter
MLP	Multilayer Perceptron Classifier
ROI	Region of Interest
ITS	Intelligent Transportation Systems
ICT	Information and Communication Technology
MEMS	Micro Electromechanical System
GSM	Global System for Mobile
HE	Histogram Equalization
IDE	Integrated Development Environment
FPS	Frames Per Second
GPRS	General Packet Radio Service
TP, TN, FP, FN	True Positives, True Negatives, False Positives, False Negatives
PC	Personal Computer
GPU	Graphics Processing Unit
VSV	Vibration Sensor Value
AST	Accelerometer Sensor Threshold
ASNV	Accelerometer Sensor Normal Value

<b>Abbreviation</b>	<b>Meaning</b>
ASCV	Accelerometer Sensor Current Value
AS	Accident Suspected
ASA	Accident Suspect Alarm
AAT	Accident Alarm Timer
DA	Driver Acknowledgement
AD	Accident Detected
NTU	Notification to User
NST	Notification Sent Time

# CHAPTER 1 - INTRODUCTION

## 1.1 Prolegomena

This final report mostly communicates information regarding the project's functionality and performance. Furthermore, this will clearly indicate the project's development. The project was intended to be a prediction and alarm system for driver drowsiness and accident detection and notification. Further, this chapter will explain the background and motivation of the research project, problem statement, aim, and objectives of the project, scope of the project, and the dissertation structure.

## 1.2 Background and Motivation

People are making more money and purchasing more automobiles in today's environment. Every day, the number of automobiles on the road grows. While this reflects an accurate degree of growth, it also indicates an increased risk of road accidents. Accidents on the road are frequently a disaster and very dangerous.

It is well recognized that tiredness and driving are not a good mix. To be a successful driver, you must be attentive, alert, and focused on the work at hand, and combining these attributes with drowsiness is challenging. Distraction is another trait that does not mix well with driving. Both these disturbances will affect the driver and cause a problem. That problem might be even worse than death.

According to several studies and data, drowsiness is one of the leading causes of serious road accidents in our everyday lives. According to the AAA Foundation for Traffic Safety (Tefft, 2014), 328,000 drowsy driving collisions occur each year. That is more than three times the number stated by the police. According to the same research, 109,000 drowsy driving collisions resulted in injuries, with around 6,400 fatalities. According to the study, the number of sleepy driving deaths is more than 350 percent more than previously recorded (Tefft, 2014). According to the National Highway Traffic Safety Administration<sup>1</sup>, 795 people died in car accidents involving sleepy driving in 2017. There was a total of 4,111 deaths caused by sleepy driving between 2013 and 2017. In 2017, 91,000 accidents involving drowsy drivers were reported to the police. Approximately 50,000 persons were injured because of these collisions. In 2013, the

---

<sup>1</sup> <https://www.nhtsa.gov/risky-driving/drowsy-driving>



World Health Organization published a worldwide status report on road safety, which detailed the prevalence of auto accidents in 182 countries and revealed that automobile accidents cause 1.24 million lives per year. (World Health Organization, 2013)

According to the AAA Foundation's 2019 traffic safety index (AAA Foundation, 2019), 96% of drivers consider drowsy driving to be very or extremely risky. Only 29% believe that sleepy drivers are at risk of being arrested by the police. Drowsy driving is frowned upon by almost 97% of drivers. Despite the high rates of perceived risk and social stigma associated with sleepy driving, over a quarter of drivers says that they are unable to keep their eyes open while driving.

Sri Lanka, a middle-income country, is facing the burden of road traffic accidents (RTCs), and associated injuries and death, due to an exponential growth in motorization on a static road system. The traffic police statistics of Sri Lanka show an increase of RTCs reported during the recent past. During the 2009-2015 period, the total number of RTCs increased by 15% while deaths increased by 17% ( World Health Organization, 2013)

Therefore, considering all the above-mentioned statistics and factors it is important to provide a solution using computer science concepts to keep drivers active and reduce accidents. Therefore, this project emphasizes having a solution to address the driver drowsiness problem.

### **1.3 Problem Statement**

The problem is to detect the drowsiness of the driver in the vehicle. If the driver is noticeably inattentive and drowsy there is a huge chance that he/she might face an accident. To solve this problem, there are mainly three methods available, and this project is to propose a drowsiness detection system for drivers by using a behavioral-based drowsiness detection method. It is known that a driver is under drowsiness influences by looking at the eyelid and the yawning patterns. If the driver shows signs of drowsiness, a warning or alert should be given to the driver. Part of the challenge is to come up with something better than the low-cost solutions that are already available. Further, in addition to the problem, there can be some scenarios where accidents can happen due to some other factors. In case of that this project will give a solution to locate the location of the vehicle and notify the relevant parties such as relatives, friend circles and other services etc.

## **1.4 Aim and Objectives**

### **1.4.1 Aim**

The goal of this research project is to develop a new system that will aid drivers which detects drowsiness using facial behaviors and alert them before they get into trouble including accident alert to other external necessary parties such as relatives, friend circles etc. using Image processing, Machine learning and Internet of Things (IoT) techniques.

### **1.4.2 Objectives**

- To select a suitable image classification model which can be used for drowsiness prediction.
- To select a suitable IoT technique to detect accidents on a vehicle.
- To develop a smart system that will monitor the facial expressions of the driver and notify an alert to him if the system detects that driver feels drowsy.
- To develop a tracking system to track the vehicle location when an accident occurs due to drowsiness.
- To develop a mobile application that can send and receive notifications related to an accident.
- To write a thesis that includes information about the background, research methods and design, findings, and evaluation procedure.

## **1.5 Scope of the Research**

Scope of this project is to model a system which can be used by vehicle drivers for the prediction of drowsiness using their facial expression. Because there many factors and methods which will be an influence to identify the driver's drowsiness. Those methods and factors will be studied only to gather some knowledge related to the research area. But this research design and methodologies will be limited to detect drowsiness using facial expressions such as eye closing, yawning, and behaviors. In addition to that, accident detection methods using IoT also will be considered to make the system notify the external parties such as families of drivers, friend circles, etc. These will be considered as an in-scope functionalities and other functions related to performance improvements, real world implementations, etc. will be considered as an out-of-scope functionalities.

## **1.6 Structure of the Dissertation**

The thesis is structured as follows. Chapter one gives a brief introduction to the research background and motivations, problem statement, highlighting the aim, objectives along with the research scope. Chapter two gives describe the existing approaches and studies related to drowsiness prediction and accident detection. Chapter three gives a detailed explanation about the methodology used to full fil the objectives. It also includes an illustration of the modules that have been included in the design. Chapter four represents the results and evaluations obtained by the proposed method along with a summary of new findings. Finally, chapter five summarizes the work, discusses its findings and contributions, points out limitations of the current work, and outlines directions for future research.

## **1.7 Summary**

This chapter presents the introduction of the problem that is going to be addressed and a brief description of the solution that is going to be given and the background for the problem domain. The next chapter describes the existing approaches and technologies that have invented to address the above-mentioned problem.

## **CHAPTER 2 - LITERATURE REVIEW**

### **2.1 Introduction**

The previous chapter provides a brief introduction about problem and the background information related to the problem domain. In this chapter research done by other researchers that relates to our chosen problem will be discussed. The similarities and differences between our system and the other systems that have been identified will be discussed in this chapter.

### **2.2 Image Processing and Computer Vision**

The concept and techniques of image processing on computers were initially developed in the 1960s, at Bell Laboratories (Lipkin, 1970). Image processing is one of the most used latest computer vision technology in many fields across the globe. The image processing techniques play a vital role in Image Acquisition, Image Pre-processing, Clustering, Segmentation and, Classification techniques with different kinds of images.

### **2.3 Visual Object Tracking and Detection**

In machine vision, object tracking, and detection is a critical problem. Human-computer interaction, behavior recognition, automation, and security are only a few of the areas where it can be used. Provided the original status of the target in the previous frame, visual object monitoring predicts the target location in each frame of the image series.

Authors (Lucas and Kanade, 1981) introduced that the detection of a moving object can be accomplished using the relationship of pixel between adjacent frames of a video series and pixel displacement shifts, according to the proposed method. The algorithm, on the other hand, can only identify a medium-sized object that moves between two frames. Later with the recent advances and the improvements of the correlation filter in computer vision many new techniques were introduced including Minimum Output Sum of Squared Error (MOSSE) filter (Bolme et al., 2010). When it starts with a single frame, this creates stable correlation filters. This model's idea is simple, it works quickly, and it does a superior job of separating the target from the backdrop. As a result, several follow-up studies using this strategy have had positive results. Although its high computational power, the MOSSE algorithm's precision is poor, and it can only process gray color information.

Henriques et al. (2012) developed the Circulant Structure of a tracking-by-detection with Kernel (CSK) method based on the MOSSE filter; in this method, ridge regression was used as the loss function. CSK calculated the closed-form solution of the correlation filter after optimizing the training goal. Furthermore, by utilizing the circulant matrix's features, it was able to significantly minimize the matrix multiplication operation in the Fourier domain while maintaining MOSSE's performance advantage and gaining a better tracking effect.

Again Henriques et al. (2015) offered a way for converting CSK's single-channel filter to a multidimensional filter, and a unique kernel correlation filter (KCF) methodology was offered to replace CSK's one-dimensional grayscale feature with a multidimensional Histogram of Oriented Gradient (HOG) (McConnell, 1986) feature. Through experiments, this study discovered that KCF significantly improved tracker performance.

Based on the correlation filter framework, Li and Zhu (2015) proposed a highly attractive tracker. This study proposes a scale adaptive technique to handle the issue of the set template size of the kernel correlation filter tracker.

Furthermore, various object detection frameworks have been introduced. Haar feature-based cascade classifiers are very intelligent in object identification. This approach was developed by Viola and Jones (2001) in their research work, "Rapid Object Detection Using a Boosted Cascade of Simple Features". A cascade function is taught using many positive and negative pictures which is useful in machine learning. Later it was used to identify objects in images. This framework is already included in the OpenCV<sup>2</sup> image processing and computer vision library.

HOG features were implemented by Dalal and Triggs (2005) in their paper, "Histograms of oriented gradients for human detection". They showed how the HOG image descriptor and a Linear SVM might be utilized to train extremely accurate object classifiers. In their work they focused on human detectors. The HOG is a function descriptor that is largely utilized in image processing and computer vision for object detection. A function descriptor is a representation of an image or an image patch that removes significant information from the image to make it simpler. McConnell (1986) was the first to introduce the notion.

---

<sup>2</sup> <https://opencv.org/>

On the other hand, due to state-of-the-art findings gained in the domains of image classification, object identification, and natural language processing, deep learning technology has become a buzzword nowadays. Deep learning is a type of Artificial Neural Network (ANN) that has multiple layers of processing. It is a type of machine learning technique that uses data to learn characteristics. CNNs are a sort of deep, feed-forward artificial neural network that has been shown to perform well in picture categorization and detection applications. CNNs are a sort of neural network with more layers than a traditional neural network. Through a nonlinear activation, it contains weights, biases, and outputs. The following Figure 2.1 below shows the overall architecture of a CNN model.

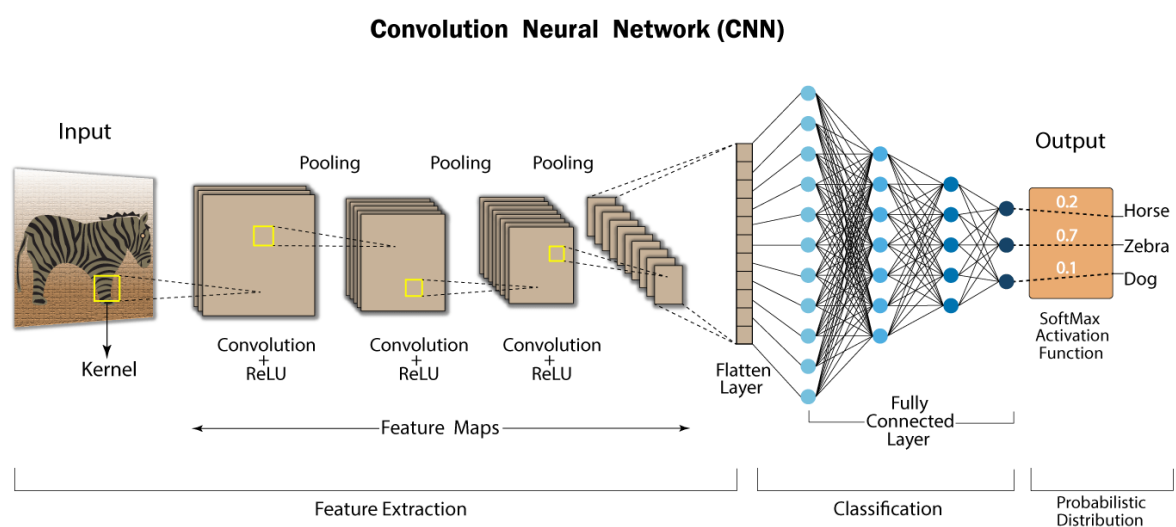


Figure 2.1. Overall Architecture of CNN<sup>3</sup>

## 2.4 Facial Landmarks Recognition

The goal of facial key-points recognition is to obtain vital information regarding the placement of the eyes, brows, nose, and lips on the face. With the improvements of deep learning, this became easy comparing the past. For the very first time Sun et al. (2013) proposed Deep Convolutional Neural Network (DCNN) based on CNN to recognize human face key points for the first time in their research article, "Deep Convolutional Network Cascade for Facial Point Detection". This algorithm only detects five facial key points: two for each eye, one for the nose, and two for the mouth.

<sup>3</sup> [https://i0.wp.com/developersbreach.com/wp-content/uploads/2020/08/cnn\\_banner.png?fit=1400%2C658&ssl=1](https://i0.wp.com/developersbreach.com/wp-content/uploads/2020/08/cnn_banner.png?fit=1400%2C658&ssl=1)

There are two advantages to use DCNN in facial landmark localization: 1) Geometric restrictions among face points are used implicitly; 2) a large quantity of training data may be used. However, in the job of comprehensive facial landmark localization, more than 50 face landmark points must be found in a unified system. It presents a significant challenge in the structure design and training process of typical convolutional networks. To achieve greater precision for facial key point identification, Zhou et al. (2013) used FACE++, which optimizes DCNN and can detect 68 facial key points, although this approach involves too many models and its operation is highly difficult.

Tweaked Convolutional Neural Networks (TCNN) based on the Gaussian Mixture Model (GMM) were proposed by Wu et al. (2016) to enhance different layers of CNN. However, the robustness of TCNN is too dependent on data. Then, Kowalski et al. (2017) presented Deep Alignment Network (DAN) to distinguish face important points, which outperforms previous methods. Unfortunately, DAN necessitates extensive models and calculations based on complex functions.

To achieve the need for real-time speed, CrashFree employs Dlib<sup>4</sup> (King, n.d.) to detect facial key points. The Dlib library's facial landmark detector is an implementation of the research paper "One millisecond face alignment with an ensemble of regression trees" by Kazemi and Sullivan (2014). This is also the same approach as Face++ which can be used to identify 68 facial key points in a particular image that contains a face.

## **2.5 Internet Of Things (IoT)**

IoT is a network of digital devices, and human with unique identifiers (UIDs). It has the ability to send the data from human to computer or human to human. The idea of a network of smart devices was explored as early as 1982, when a modified Coke machine at Carnegie Mellon University became the first Internet-connected appliance, reporting its inventory and whether newly loaded beverages were cold<sup>5</sup>. Kevin Ashton, the co-founder of the Auto-ID Center at the Massachusetts Institute of Technology (MIT), originally referenced the internet of things in a 1999 presentation to Procter & Gamble (P&G) (Ashton, 2016). An IoT ecosystem is a combination of internet enabled digital devices which use embedded systems. These systems

---

<sup>4</sup> <http://dlib.net/>

<sup>5</sup> [https://www.cs.cmu.edu/~coke/history\\_long.txt](https://www.cs.cmu.edu/~coke/history_long.txt)

can share and gather the data from their environment. IoT devices can share the sensor data with another IoT platform or any other devices. It is also possible to send the data to a cloud platform. These IoT devices may communicate with each other occasionally. Even though humans are involved to configure the devices, most of the work is done without a human interaction. The Figure 2.2 following shows an example overview of an IoT system and its components.

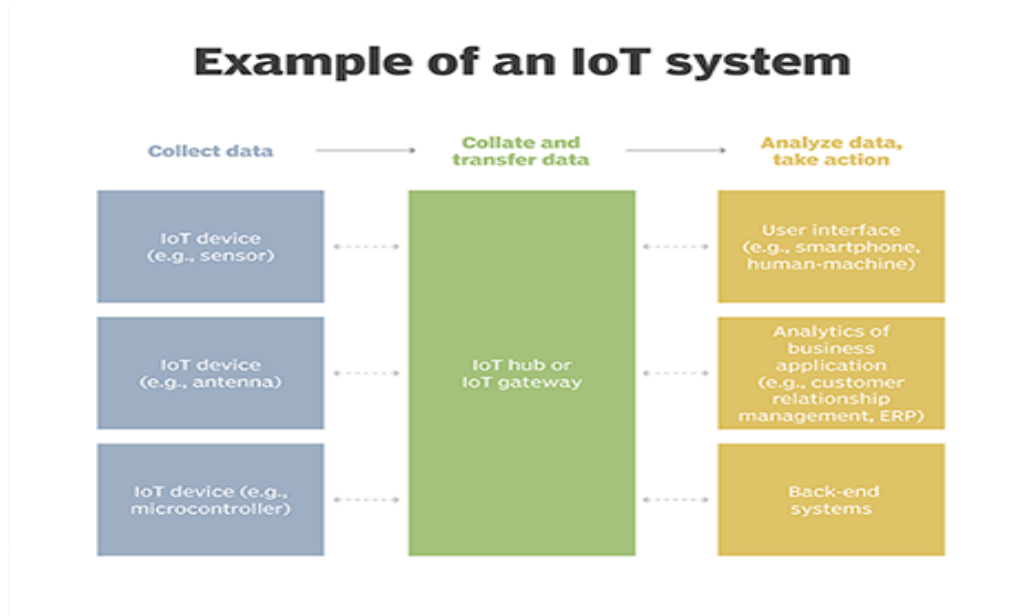


Figure 2.2. Example overview of an IoT system<sup>6</sup>

According to International Data Corporation (IDC), there will be 55.9 billion connected devices worldwide by 2025, with 75 percent of them connected to an IoT platform. Data created by linked IoT devices would amount to 79.4 ZB by 2025, up from 13.6 ZB in 2019<sup>7</sup>.

## 2.6 Drowsiness Detection Methods

Drowsiness is one of the major factors that influence accidents as discussed before. The gigantic expansion in road accidents around the globe has prompted genuine examination among researchers. A wide range of studies has been presented to improve driver safety. However, most existing drowsiness detection algorithms have a few drawbacks. Many research papers including Saini and Saini (2014) and Ramzan et al. (2019) states that drowsiness detection methods can be classified as shown in Figure 2.3 below.

<sup>6</sup> [https://cdn.ttgtmedia.com/rms/onlineimages/iota-iot\\_system\\_mobile.png](https://cdn.ttgtmedia.com/rms/onlineimages/iota-iot_system_mobile.png)

<sup>7</sup> <https://www.idc.com/getdoc.jsp?containerId=prAP46737220>



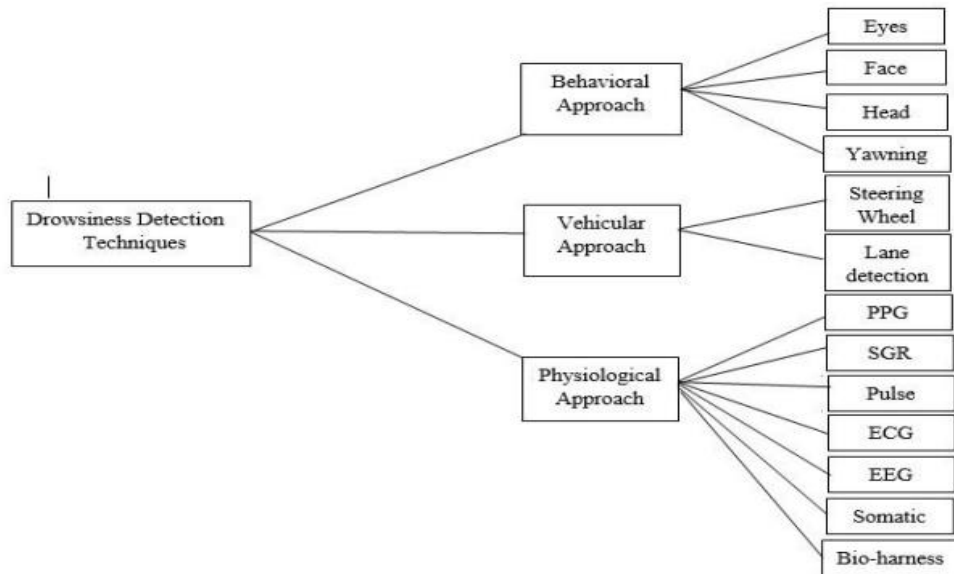


Figure 2.3. Drowsiness Detection Methods

As mentioned in the above Figure 2.3, Psychophysiological detection approaches identify drowsiness based on the physical parameters of the driver, such as pulse rate, breathing rate, heart rate, respiratory rate, and body temperature, among other things. These biological criteria are more effective and precise in detecting drowsiness since they are concerned about what is going on physically with the driver. In the paper, “A Hybrid Approach to Detect Driver Drowsiness Utilizing Psychophysiological Signals to Improve System Performance and Wearability”, the researchers (Awais et al., 2017) have been used a method based on electrocardiography (ECG) and electroencephalography (EEG) signals which will be a useful feature to measure heart rate. A simulated environment has been used to record the EEG and ECG values. Authors (Mohamed et al., 2018) used only EEG signals. Multilayer neural network (MLNN) and Adaptive neuro fuzzy (ANF) algorithms have been used to train a model which can predict drowsiness.

Authors (Li and Chung, 2013) proposed a technique to predict drowsiness based on Heart Rate Variability (HRV) analysis. For feature selection and classification, they employed Receiver Operation Curve (ROC) analysis and a Support vector machine SVM classifier, respectively. HRV was identified by using Photoplethysmogram (PPG) sensors. Arunasalam et al. (2020) suggested a system that employs a heartbeat sensor and an eye blink sensor to detect the driver's heart rate and eye blink rates. The method also incorporates a Global Positioning System (GPS) that can track the vehicle's present location.

Li et al. (2015) introduced a SVM based Posterior Probabilistic Model (SVMPPM) for drowsiness detection, with the goal of translating the sleepiness level to any value between 0 and 1 instead of discrete labels. To assess the suggested model in real-time, a fully wearable EEG system comprised of an EEG headgear with Bluetooth facility and a wristwatch was employed.

Techniques for detecting driver fatigue that employs vehicular factors include vehicle speed variability, steering wheel grip force, frequent lane shifting patterns, steering wheel tilt, and so on. These procedures need the use of sensors on vehicle features such as the accelerator, steering wheel, brake pedal, and so on. The signals generated by these sensors are utilized to determine driver drowsiness. The primary goal of these tactics is to examine driving patterns and detect a reduction in the performance of driving due to fatigue or tiredness. Katyal et al. (2014) presented a drowsiness detection system to solve concerns such as reckless driving, inexperience, disregarding signboards, and signal jumping, among others. The Hough transform is used by the system to recognize lanes. Second, it detects drowsiness in the driver's eyes. Then Zhenhai et al. (2017) proposed another method for drowsiness detection using time series analysis of steering wheel angular velocity. In the research paper "Online Detection of Driver Fatigue Using Steering Wheel Angles for Real Driving Conditions" Li et al. (2017) presented an internet application for sleepiness that uses Steering Wheel Angles (SWA) to assess drivers' tired levels in real-world situations. SWA data is obtained using sensors mounted on the steering wheel.

Non-invasive approaches to detecting drowsiness are behavioral parameters. To detect driver weariness, these solutions employ behavioral indicators such as eye closing ratio, eye blinking, head position, facial expressions, and yawning. A mounted camera is commonly employed to monitor the facial behaviors of drivers. Therefore, by adopting the standard protocol for eliminating face characteristics from video streams, several experts, and researchers from all over the world have made major contributions to this field. Other methods, in addition to feature extraction, are used for evaluating the degree of drowsiness; commonly, machine learning algorithms such as the Naive Bayes (NB) classifier, K-Nearest Neighbors (KNN), SVM, and ANN etc. These algorithms are learned by combining features and named outputs to create models that can predict drowsiness.

Another driver fatigue detection system based on the mouth and yawning data were proposed by Sarada Devi and Bajaj (2008). To begin, the gadget locates and monitors a driver's mouth using a cascade of classifier training and mouth detection from the input photos. SVM is then used to train the representations of the mouth and yawning. Further, SVM based classifier is used to classify the features and predict results whether drowsy or not (Jumana Waleed et al., 2020).

D'Orazio et al. (2007) introduced a method that uses eyelid movement information to monitor driver fatigue, as well as new drowsiness parameters such as frequency of eye closing (FEC) and eye-closure time. Their key achievement was the creation of a consistent eye-detection method that does not place any restrictions on the driver and does not require any preprocessing to segment eye regions. They showed that their system's efficiency is equal to that of systems that use psychophysiological signals.

Based on machine vision and the Adaboost algorithm (Viola and Jones, 2001), Kong et al. (2015) suggested an improved method and practical system for detecting driving weariness. The suggested technique first detects faces effectively using front face and deflected face classifiers. The potential eye area is then selected based on the geometric distribution of face organs. Finally, trained classifiers of open and closed eyes are employed to locate eyeballs swiftly and correctly in the candidate region. developed a detection technique for detecting eye movements to know whether it is closed or open. The eye region is detected using the Viola-Jones algorithm (Viola and Jones, 2001). Advanced Local Binary Pattern (ALBP) (Ma et al., 2008) is used to extract eye and eye region-related features. These features are collected, and it was trained and tested using NB and SVM. The results show that the system had a accuracy of high rate when compared to other current approaches, with the accuracy rate of NB and SVM utilizing an eye detection data set with training 90% and testing 10% being 96 and 97 percent, respectively.

Another model for real-time driver monitoring was described by Pinto et al. (2019), in which the method of HOG (McConnell, 1986) is used for detecting the driver's face from the acquired video frames. The ensemble of regression trees is then used to generate eye bounding boxes. The bounding boxes are then expanded and input into a CNN to detect the driver's eye condition.

Omidyeganeh et al. (n.d.) proposed a method that offers a reliable and knowledgeable scheme for detecting driver drowsiness that combines eye closing and yawning identification approaches. A fusion approach is utilized to improve tiredness detection. Structural Similarity Measure (SSIM) is used to extract the eyes from the face area. SSIM is a novel approach for calculating the similarity of two pictures. (Wang et al., 2004). Deng and Wu (2019) present an MC-KCF method based on deep learning and the correlation filter. This approach employs CNN and Multi-task Cascaded Convolutional Neural Networks (MTCNN) to compensate for the limitations of the Kernel Correlation Filter (KCF), which is used to track objects. Using their method, the algorithm can follow the driver's face in real-time.

On the other hand, there are other research available for drowsiness detection based on deep learning models like CNN, Multilayer Perceptron Classifier (MLP) etc. Research paper “A Deep Learning Approach to Detect Drowsy Drivers in Real Time” by Pinto et al. (2019) and another paper by Hashemi et al. (2020) suggested an algorithm that tracks the driver's eyes and feeds them into a pre-trained CNN model that predicts the condition of the eye. Once the forecast is acquired, it will be able to determine whether or not the driver is drowsy. Jabbar et al. (2018) developed a method for predicting drowsiness using an MLP trained model.

Zhao et al. (2020) proposed a paper to state the drowsiness by using the combination of several techniques. In the proposed algorithm, the MTCNN architecture is used in face identification and feature point localization, and feature points are used to extract the Region of Interest (ROI). This ROI images was used to detect the states of the eyes and mouth with the help of a CNN named EM-CNN.

Among the existing approaches, this study will concentrate on behavioral drowsiness detection using deep learning models since other methods are intrusive and not advisable at the current pandemic situation. Therefore, it was identified that CNN models work well when it comes to image processing and classifications. Further, this study will be focusing on those models.

## **2.7 IoT and Accident Detection**

The IoT has advanced in recent years, with breakthroughs in a few diverse applications in the military, navy, transportation, intelligence, and many other sectors relating to human safety and comfort. Despite these facts, the IoT for Intelligent Transportation Systems (ITS) generates a

lot of attention compared to traditional Information and Communication Technology (ICT). However, accidents are rather regular and are rising day by day.

Many researchers have done their contribution to make use of IoT along with this problem. There are several methods and strategies available to handle safety of passengers, communication between vehicles, and post-accident freeing tasks. Those are categorized into two major categories as hardware-based and software-based systems (Bhatti et al., 2019).

Ki and Lee (2007) propose an automated method for detecting, recording, and reporting incidents at intersections. Cameras at the intersections recognize the vehicle and its data, such as speed, direction, velocity, and area. The system makes its choice based on the features it has retrieved. We may learn about the causes of an accident as well as the characteristics of junctions that have an influence. We may learn about the causes of an accident including the characteristics of junctions that affect safety. This technology only works at junctions, and it will not detect accidents at non-intersection locations.

The authors (Tushara and P.A., 2016) suggest an accident detection system in which all operations are controlled by a microcontroller. The alert messages will be delivered to a pre-determined phone number. The system's performance review revealed that accidents were falsely reported. The technology in place is simply for accident detection and has nothing to do with a rescue system. Chaturvedi and Srivastava (2014) introduced an accident identification and reporting system that uses a single sensor to identify accidents. Where Routh et al. (2019) uses Micro Electro Mechanical System (MEMS) sensor to detect accidents. When an accident is detected, Prabha et al. (2014) and Routh et al. (2019) uses Global System for Mobile (GSM) modules to send a message to the nearest places and relatives. Authors Prabha et al. (2014) and Routh et al. (2019) use a GPS tracking system to locate the vehicle when an accident occurs. Liang (2015) offers a technique that identifies accidents using SVM and IoT. This technique is used for both accident detection and traffic prediction.

Aloul et al. (2015) and Lahn et al. (2015) used the accelerometer sensor as the main sensor in a smartphone to identify an accident. The accelerometer data is continually received by this system, which is used to estimate the severity of an accident. It sends information about the accident to the medical care provider and alerts them of the location of the accident. Zhao et al. (2020) demonstrated a mobile device-based crash notification system that identifies accidents

using accelerometer sensor and GPS data. This system takes a long time to deliver an accident message. Those messages contain relevant data of the vehicle location.

## **2.8 Summary**

This chapter discussed numerous ways that others have tried and evaluated in the field of driver drowsiness prediction and accident detection. It offers specifics on various tactics and the technology employed to achieve their objectives. This chapter also outlines the shortcomings of the methodologies used previously. When considering the approaches above, it is clearly visible that they are only focused on some scenarios. Most of the research related to driver drowsiness detection is based on recorded videos and based on some parameters. But this research will focus on giving a solution to driver drowsiness in real-time using behavioral-based methods and machine learning models. Addition to that it is identified accident detection is based on one sensor which may ended up with false notifications. But this study will focus on providing a solution which can minimize the accident false notification.

## **CHAPTER 3 - METHODOLOGY**

### **3.1 Introduction**

The preceding chapter contains a comprehensive discussion of related systems and research. This chapter outlines the suggested design approach to the problem as well as the techniques employed and design considerations. Several design concerns were highlighted as a result of the background study's critical examination. The system architecture was created with the design constraints and defined requirements in mind. This chapter explains the system architecture, which is made up of numerous modules. The detailed descriptions of problem analysis, research approach and solution are explained under separate sections.

### **3.2 Problem Analysis**

Every successful system undergoes a well-planned development lifecycle. It is necessary to understand the problem and analyze the problem to give a successful output from the research. As stated in Section 1 above, drowsiness and driving are not a good mix to enjoy. Surveys and records all over the world prove that drowsiness and inattention is the key factor for many accidents that we face. Therefore, this problem is to be addressed in a scientific manner by giving a solution using computer science tools and techniques. As a result, from literature review, it was highlighted that drowsiness plays a major role in accidents. Therefore, the above-mentioned problem should be sorted out by giving a solution.

The major goal of this project was to create a solution for detecting the drowsiness of the driver in a vehicle and preventing accidents if the driver is drowsy. If the driver is visibly inattentive and drowsy, there is a huge risk he or she may have an accident. To solve this problem, there are mainly three methods available as discussed above in Section 2. It is known that a driver is under drowsiness influences by looking at the eyelid and the yawning patterns. If the driver shows signs of drowsiness, a warning or alert should be given to the driver. Aside from that, part of the solution is to design something better than the present low-cost solutions. Further, in addition to the problem, there can be some scenarios where accidents can happen due to some other factors. In case of that this project will give a solution to locate the location of the vehicle and notify the relevant parties including the location of the vehicle.

### **3.3 Research Approach of CrashFree**

A broad analysis of the behaviors of drivers who are drowsy paved the path to study in this context. The investigation began with a focus on ways and techniques that have been utilized to identify the driver's drowsiness while driving and progressed to each way to identify an appropriate way to go with. However, finally behavioral-based drowsiness detection technique among available techniques is used in this research since other methods are not advisable during this epidemic situation and intrusive as mentioned in the literature review. The behavioral-based methods work on detecting behavioral evidence given by drivers when they are in a state of drowsiness. Typically, these methods concentrate on facial expressions that could indicate attributes like frequent yawning, rapid and constant blinking, or head swinging, these expressions might refer that the driver is awake or drowsy.

Thus, this study will specifically be focused on eye state and yawning. Detailed information about behavioral-based drowsiness methods including the image and video of drivers in both condition of awake and alert was first gathered, and a solid dataset was derived. Then the dataset has been trained using machine learning models and finally, generalized conclusions were produced based on the image or video that we provide to the system. Furthermore, this study will use scientific approach as the research methodology.

It is possible to emphasize that the drowsiness detection method was applied as this study will be very helpful to detect driver's drowsiness and alert them before they get into a struggle. However, some of the prior research's conclusions are also verified in this study. As a result, a behavioral-based approach was chosen to develop the system. In addition to that an IoT based solution using Arduino based platform will be developed to address the accident detection and notification. The accident detection module will use two sensor-based approach to suspect an accident from a vehicle. The suggested system CrashFree's high-level design is depicted in Figure 3.1. It shows a simplified form of the proposed system and its components.



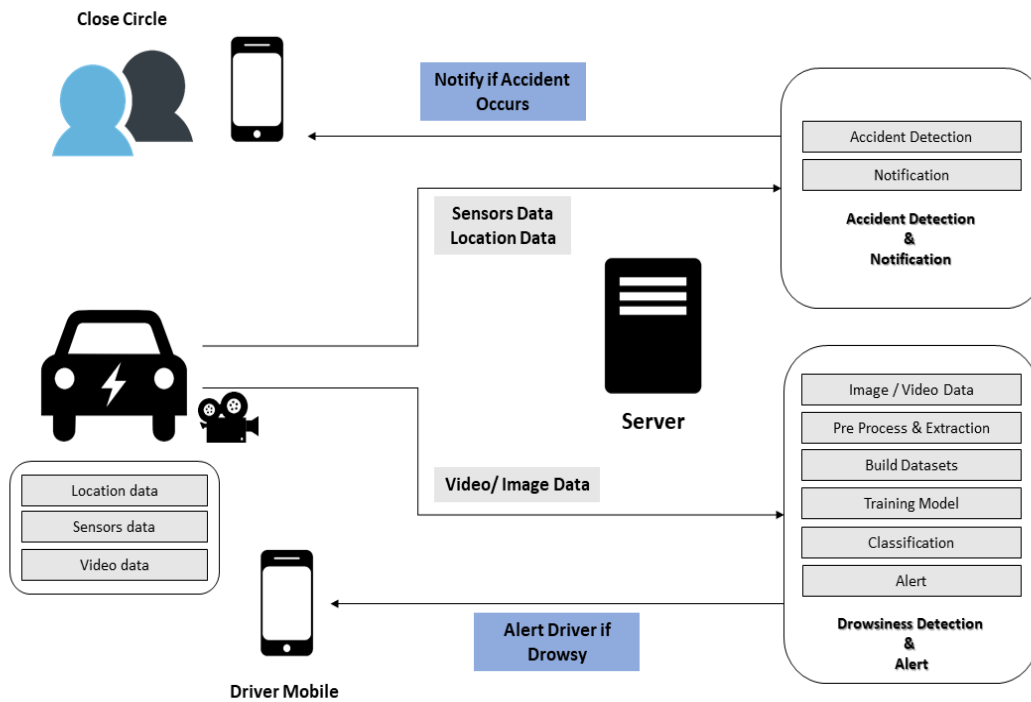


Figure 3.1. CrashFree Overall High-level Diagram

### 3.4 Research Design

A well-designed system is adaptable, secure, and efficient. More work put in during the design phase makes implementation easier. It has been worked diligently on numerous design methodologies to ensure that the end criteria decided from the analysis phase were met as precisely as feasible.

The study's research design served as a roadmap for addressing the study's aim and objectives. The research design consists of two major modules such as:

1. Drowsiness prediction and alert
2. Accident detection and notification.

The research design starts with the data collection process where data is collected through various sources. The public image data were extracted from publicly available data sources like Kaggle<sup>8</sup>. Whereas image data which indicates both alert and drowsy state was gathered from 10 participants who volunteered in the research process. The extraction of the public video data

<sup>8</sup> <https://www.kaggle.com/>

led to gain the knowledge of image categorization regarding drowsiness which helped to categorize the image data of participants easily.

### 3.5 Drowsiness Prediction and Alert

Drowsiness prediction and the alert module are one of the major components of CrashFree. It is designed with several sub modules which will be an essential part of the major module. It consists of the following submodules such as,

1. Pre-Process and Extraction Module
2. Build Dataset Module
3. Train Model Module
4. Image Capture Module
5. Classification Module
6. Alert Module

The following Figure 3.2 shows the detailed explanation of the Drowsiness prediction and alert module.

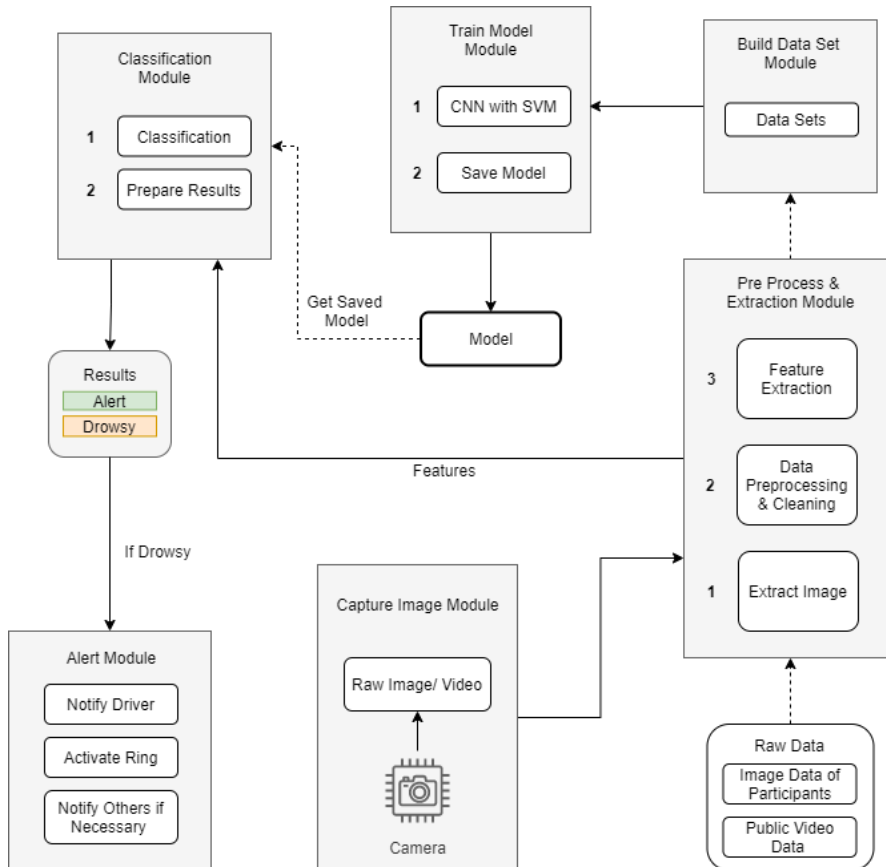


Figure 3.2. Drowsiness Prediction and Alert Module Overview

Even though the design has above six modules connected, the first three modules will be used only once at a time throughout the life cycle of the system. And other three modules are designed to execute repeatedly.

### **3.6 Pre-Process and Extraction Module**

The first stage in building a machine learning model is data preparation, and it is the starting point of the process. Real-world data is frequently inadequate, untrustworthy, deceptive (owing to errors or outliers), and deficient inaccurate attribute values and trends. Therefore, it is necessary to do some data preprocessing and preparation. The process cleans, prepares, and organizes raw data to make working with machine learning models simpler.

This module is designed to collect the raw data from the available sources and extract images from video sources if necessary. Then a collection of image sets will be pass-through data preprocess and cleaning sub module where it goes through each image and adjust every image by resizing and applying illumination techniques etc. In order to do more enhancements on a particular image, Kamel and Guan (1989) introduced an enhancement technique call Histogram Equalization (HE). Because of its great effectiveness and simplicity, histogram equalization is one of the most often used methods for picture contrast enhancement. It is accomplished by normalizing the intensity distribution using its cumulative distribution function, resulting in a picture with a uniform intensity distribution.

In this study, HE was used for the enhancement of the image. It is a well-known contrast enhancement due to its performance on almost all types of images. Spyder Anaconda<sup>9</sup>, Google Colab<sup>10</sup> allowed to create, share, explore, and interact with scripts in real time. Python scripting language<sup>11</sup> and OpenCV was used in writing codes as there are many key features available in OpenCV and it supports Python. During data processing and cleaning, as the initial stage OpenCV was imported into the Integrated Development Environment (IDE), thereafter the raw image was enhanced using HE.

---

<sup>9</sup> <https://www.anaconda.com/>

<sup>10</sup> <https://colab.research.google.com/notebooks/intro.ipynb>

<sup>11</sup> <https://www.python.org/>

Initial data images contain the images of participants, and a public image dataset that has a closed eye, open eye images, yawn and not yawning images. The following Figure 3.3 shows the structure of available data images.

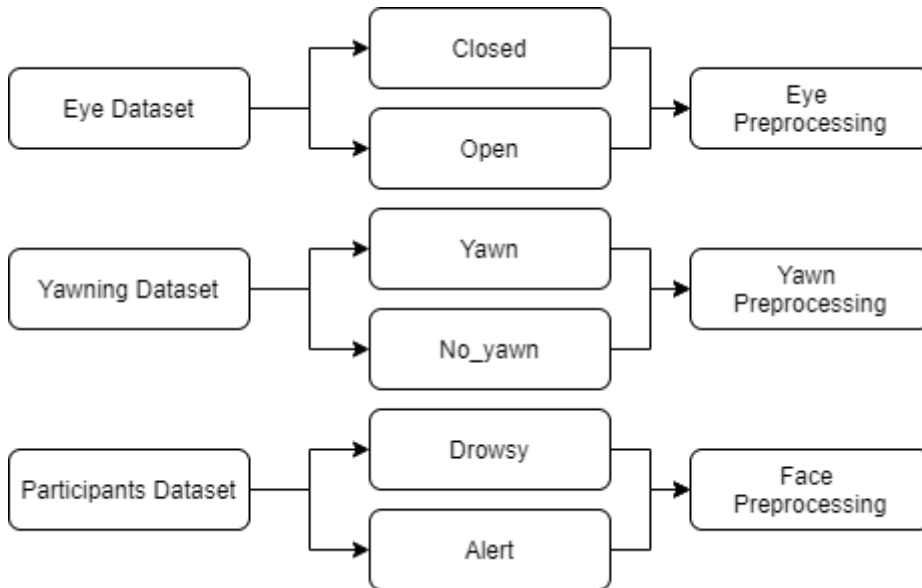


Figure 3.3. Dataset Structure and Preprocessing

Yawn preprocessing take care of identifying the face regions and mouth regions. And cropping those face and mouth images. Eye preprocessing, Mouth preprocessing and Face preprocessing do further pre-process such as histogram equalization, resizing etc. Resizing will reshape the image to (224, 224) to maintain equal shape for each image.

Haar Cascade (Viola and Jones, 2001) is one of the most used object detection methods in image processing. Edge or line detection features are used by the algorithm. On the other hand, another effective approach is the HOG. It is a feature descriptor used in computer vision and image processing for object recognition. The concept was initially introduced by McConnell (1986). Based on the experiments done in the previous studies, Authors (Rahmad et al., 2020) have proved that HOG behaves better than Haar Cascades. Therefore, this project uses HOG algorithm to detect faces and other features from the image.

After successful completion of image pre-processing, the processed image will be move to the next stage to extract separate images. Image extraction is used to grab separate images for each category. There are numerous ways and pre-trained models available to detect face and other features in an image.

Dlib has been used with Python to archive this process. Dlib itself has an inbuilt face detector, uses HOG which can detect face and face landmarks in each image. The Dlib library's facial landmark detector is an implementation of the "One millisecond face alignment with an ensemble of regression trees" a research paper by Kazemi and Sullivan (2014). Using this is easier to extract other features as well. Cropped face, cropped eye, cropped mouth images will be moved to the next phase to build the datasets, respectively. Figure 3.4 shows the process and final dataset structure.

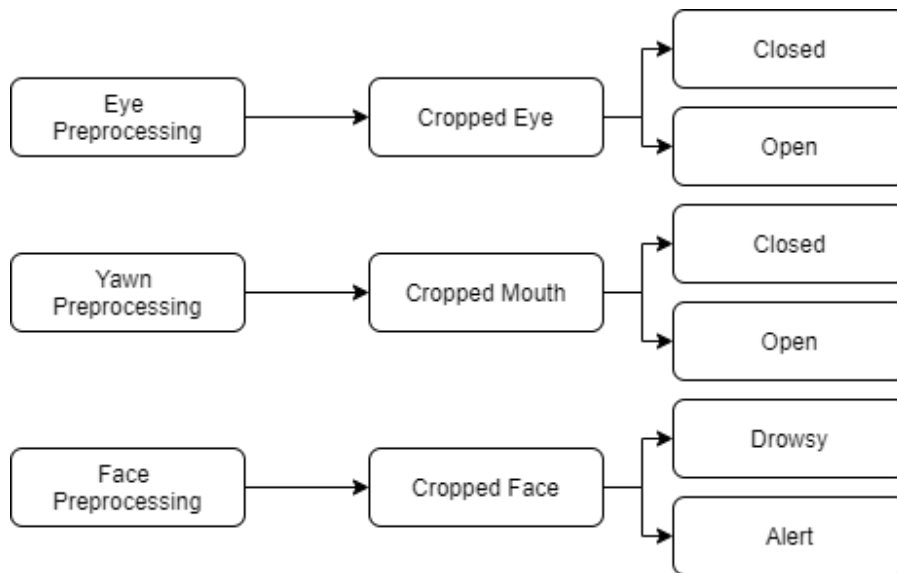


Figure 3.4. Final Dataset Structure

### 3.7 Build Dataset Module

Datasets are very important when it comes to machine learning. Machine learning models use datasets to make predictions. Datasets are nothing but a collection of large data which will be useful in the machine learning process. Building dataset module is used to capture all the images that have been produced by Preprocess module and build a dataset on top of those images. All the images that the dataset module receive will be stored as separate images for later usage. The dataset will include data of the categories such as the following, respectively.

- Opened\_eye – This folder contains the cropped images of open eyes.
- Closed\_eye – This folder contains the cropped images of closed eyes.
- Drowsy\_face – This folder is available for containing drowsy faces.
- Alert\_face – This folder is responsible for containing alert faces/ non-drowsy faces.
- Opened\_mouth – This folder contains images of an open mouth.
- Closed\_mouth – This folder has images of a closed mouth.

## 3.8 Train Model Module

After building the datasets successfully, datasets will be used to train the prediction model. Datasets will be divided into two parts: train and test sets. Train sets will be used to train the model, and test sets will be used to test the model and check the accuracy of the model. Testing of the dataset will be done right after the completion of training. Each train set and the test set will be containing the images categorized as Open\_eye, Closed\_eye, Drowsy\_face, Alert\_face, Open\_mouth, and Closed\_mouth as mentioned in the Build dataset module.

Each category from a particular image will be trained using a CNN algorithm with the combination of support for SVM which was developed by Cortes and Vapnik (1995). The CNN is a deep learning neural network that belongs to the convolutional neural network family. CNNs are a significant advancement in image recognition technology. They are regularly utilized behind the scenes in picture categorization and are most usually utilized to examine visual imagery.

For CNN implementation, the Keras<sup>12</sup> sequential model was employed, which is a notion suitable for modeling a feed-forward network. Keras is an open-source library which is highly used for ANNs. Keras serves as the TensorFlow<sup>13</sup> interface. The network definition is made up of layers. The concept of the layer in the Keras sequential model does not perfectly transfer into the previously given notion of the layer from a topological standpoint. Keras layers are finer granular, and numerous Keras layers are required to build an equal topological layer.

### 3.8.1 Define Training Model

To define a model, it is necessary to use the constructor Sequential. This is where the model definition starts.

```
model = Sequential()
```

After defining the model each layer is added by calling the add function on the object as shown below. It is necessary to pass the definition of the layer as a parameter.

```
model.add(layer)
```

---

<sup>12</sup> <https://keras.io/>

<sup>13</sup> <https://www.tensorflow.org/>

### 3.8.2 Keras Model Layers

All models were built by combining the layers listed below.

#### *Convolutional*

From the input data, convolutional filters are utilized to generate an activation map. The architecture's convolutional layer had the following structure.

```
Conv2D(filters=n, strides=(s, s), padding='same', kernel_size=(z,z),  
input_shape=shape, activation= activation_function)
```

Here the number of filters is denoted by n, number of pixels in stride is given by s, kernel size is z, and input matrix size is provided as input\_shape.

#### *Activation / Rectified Linear Unit Layer (ReLU)*

Filters out negative numbers to deliver only positive values, resulting in a substantially shorter training time. Activation function can be directly added to the convolutional layer by specifying the activation parameter.

#### **Activation(function\_of\_activation)**

function\_of\_activation can be either 'softmax' or 'relu'. Because Keras utilizes a linear activation function for each layer, these specifications are equal. After the completion of model training, the model will be saved in a predefined place for later use.

#### *Pooling*

Nonlinear down-sampling is used to reduce the number of parameters for a simpler result. The pooling layer is defined as below.

#### **MaxPooling2D(pool\_size=(z, z), strides=(s, s))**

Here pool\_size indicates the size of the pooling kernel and strides indicates the number of pixels traveled in the x and y directions between applications of different pools.

## ***Fully Connected***

The class probability scores are computed by producing a vector of  $C$  dimensions.  $C$  denotes the number of classes. This layer connects all neurons. The fully connected layer is created by

**Dense(number\_of\_units)**

Where number\_of\_units are the number of neurons in one layer that are completely linked.

## ***Dropout***

Like the activation function dropout on a layer should be added after it as another layer.

**Dropout(p)**

Where  $p$  denotes the possibility that any unit will be discarded as well as the coefficient by which the outputs are multiplied during forwarding assessment.

Layers for feature extraction are multidimensional. Convolutional and Pooling has the structure of two-dimensional layers. Fully linked layers produce one-dimensional classification layers. It is important to develop a mapping between the two to connect them. It is important to utilize the following layer for this purpose.

## ***Other***

**Flatten()**

This is responsible for handling the appropriate connection of a layer.



### 3.8.3 CrashFree CNN Model

CrashFree uses three models to predict the state of eye, mouth, and face. It contains most of the above-mentioned layers to make a better CNN model. The following Figure 3.5 shows the architecture of the proposed CNN model. The architecture is same for each model, but some parameters are different for each.

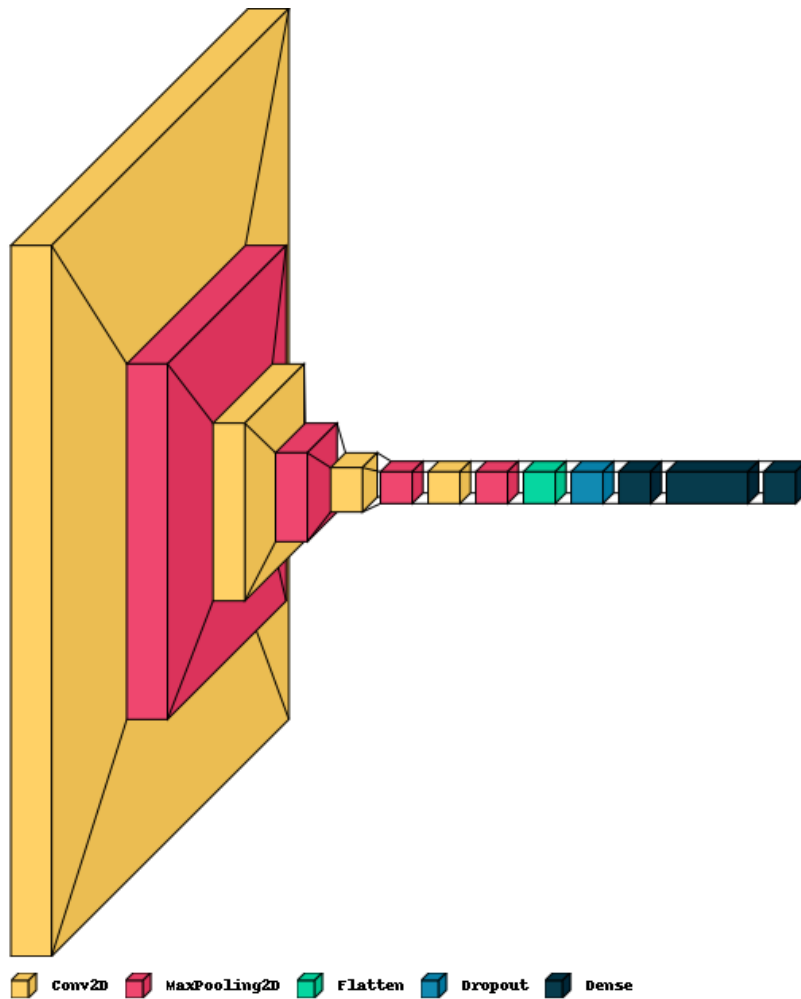


Figure 3.5. CrashFree CNN Architecture

To add SVM with the CNN model, the model needs to be modified with some parameters. The following Figure 3.6 shows the distinct approach of CrashFree that enables SVM support at the last layer of the CNN model. The last layer's weight should be regularized. So that the last layer of the model will automatically act as an SVM classifier.<sup>14 15</sup>

<sup>14</sup> <https://cs231n.github.io/linear-classify/>

<sup>15</sup> [https://github.com/nfmcclure/tensorflow\\_cookbook#ch-4-support-vector-machines](https://github.com/nfmcclure/tensorflow_cookbook#ch-4-support-vector-machines)

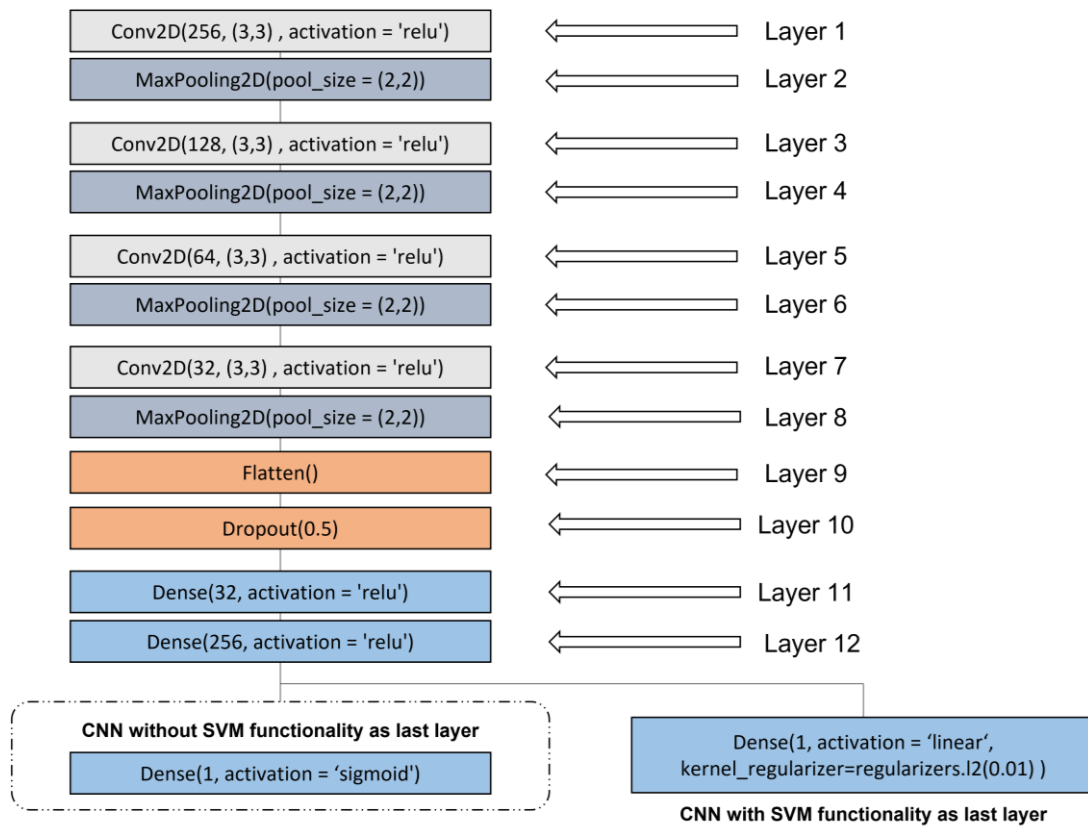


Figure 3.6. Adding SVM Support to CNN Model

### 3.8.4 Model Compilation

Before the model can be trained, the structure must be provided, as well as the cost function, optimization technique, and metrics. This is accomplished by invoking the model's build function. There should be a parameter change here as well to make the CNN model compile with an SVM loss function. The hinge loss with regularization term forms the complete SVM loss function for binary classifications. Following Figure 3.7 shows an example for the scenario.

```
model.compile(optimizer = 'adam', loss = 'hinge', metrics = ['accuracy'])
```

Figure 3.7. Hinge Loss Function

The cost function is specified by loss, the optimization technique is specified by optimizer, and the model is measured by metrics.

### 3.8.5 Model Fitting

Model fitting is the Keras term for the model training process. It provides two methods such as fit and fit\_generator. The fit method loads the whole dataset at once and uses it to train the network. The second method, fit\_generator, makes advantage of a python programming language feature called a generator. In this approach first option was used.

### 3.9 Capture Image Module

The image capture module is the one that is responsible for capturing live videos of drivers. A web camera will be installed in the dashboard of the driver's vehicle, recording the driver's facial expressions and activities at the same time. Further, the module transmits the captured video lively to the Preprocess and extraction module beneficial to extract features.

### 3.10 Classification Module

Once the training of the model is successful, now the trained model is ready to predict the unknown data. The classification module gathers unknown data and uses the saved trained model in contemplation of predicting the verdict. The module further analyses the verdict and acts accordingly. The following Figure 3.8 shows how the classification module works shortly.

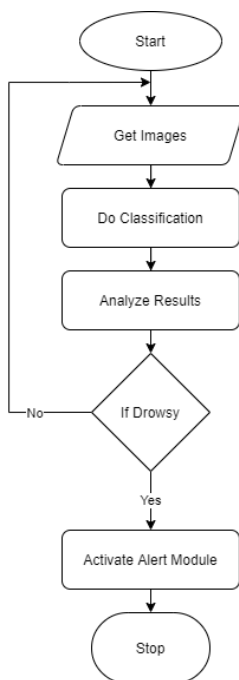


Figure 3.8. Flowchart for Classification Module

Do classification part will classify image categories that were mentioned earlier in the Build dataset module. Moreover, the Do classification part and Analyze results part can be explored in more detail as shown in the following Figure 3.9.

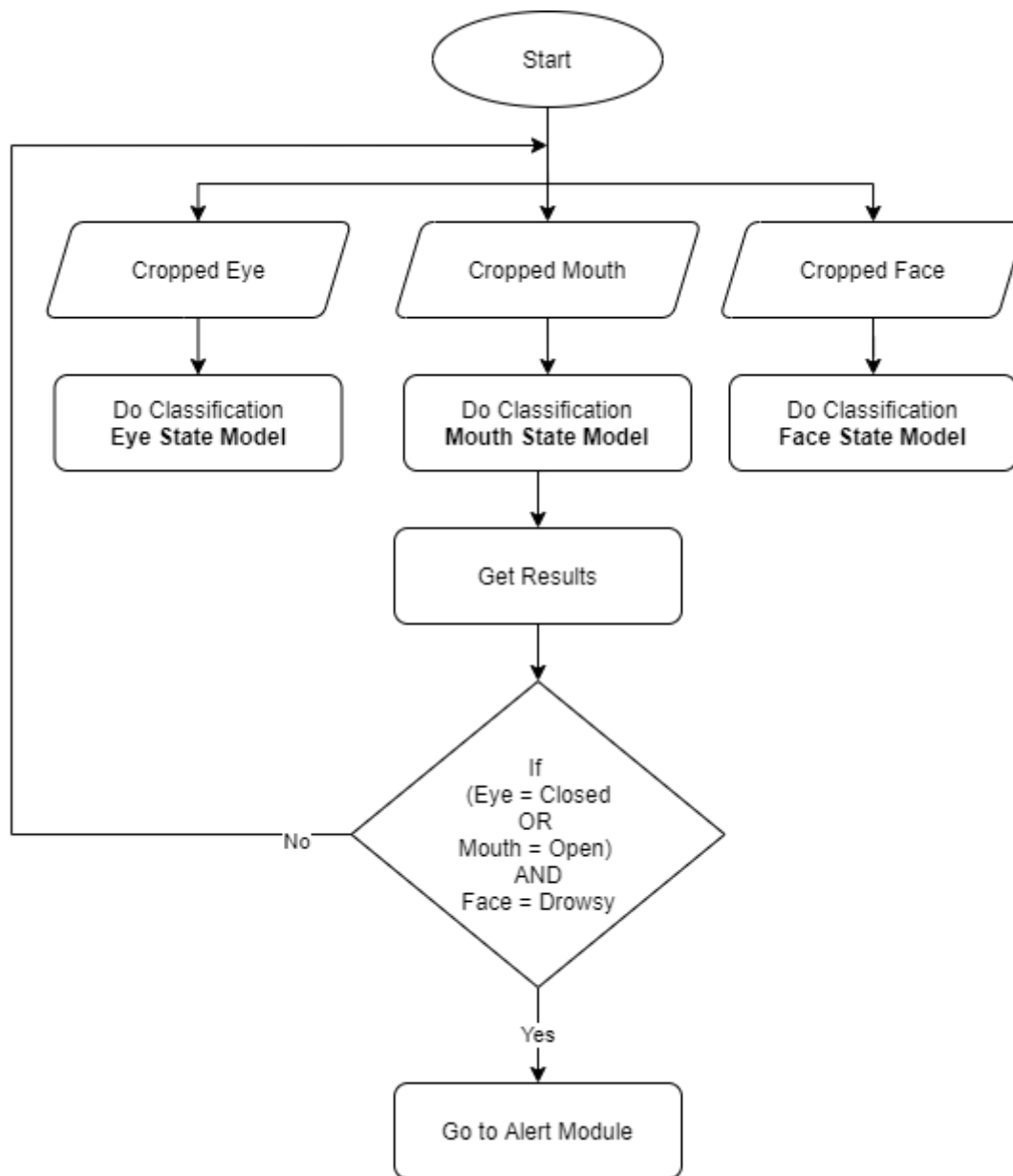


Figure 3.9. Classification Module's Drowsiness Prediction Process

Each cropped image such as Cropped\_eye, Cropped\_face, and Cropped\_mouth will be put into the trained models in order to identify their categories. Those categories will determine their respective outputs. Then the final output will be identified by using a calculation shown in below Figure 3.10.

---

**Algorithm 1:** Example code of drowsiness prediction for an image

---

```

Input: CroppedEye, CroppedMouth, CroppedFace
Output: Drowsy or Alert
1 Function EyeClassification(CroppedEye):
2   | Do classification           // Do classification and predict the eye state
3   | return faceState
4
5 Function MouthClassification(CroppedMouth):
6   | Do classification           // Do classification and predict the mouth state
7   | return mouthState
8
9 Function FaceClassification(CroppedFace):
10  | Do classification           // Do classification and predict the face state
11  | return faceState
12
13 if (EyeClassification(CroppedEye) == 'Closed' OR
      MouthClassification(CroppedMouth) == 'Open') AND
      FaceClassification(CroppedFace) == 'Drowsy' then
14  | return drowsy
15 else
16  | return alert

```

---

Figure 3.10. Algorithm for Drowsiness Prediction from an Image

The above Figure 3.10 can be considered when we are executing the classification module using a single image or single image frame. But in the practical scenario, the classification module will receive data every second. Therefore, there will be many more images according to the camera's frame per second (fps). For example, if a camera consists of 30fps, then there will be 30 images captured for a second. Therefore, only one frame for one second is considered. Once the verdict is known, then the module checks the verdict whether it is drowsy or not. If it indicates as drowsy the module attempts to activate the Alert module conducive to alert the driver.

### 3.11 Alert Module

Alert module is the last and final module under the Drowsiness Detection and Alert component where it takes care of notifying and alerting the driver once the driver feels drowsy. A mobile application specially developed for drivers will be taken care of that action by popping up a notification message and an alarm. Therefore, every driver uses this project needs to install the application on their mobile so that they can be kept updated. A successful alert will be popped up when the Classification module predicts the driver state as drowsy for consecutive two seconds.

### 3.12 Accident Detection and Notification

The accident detection and notification module are designed to be another major module of CrashFree. This module also combined with other sub-modules such accident detection module and a notification module. This module will use a combination of IoT and a smartphone platform to determine any accidents. The following diagram in Figure 3.11 shows the overall architecture of this major module.

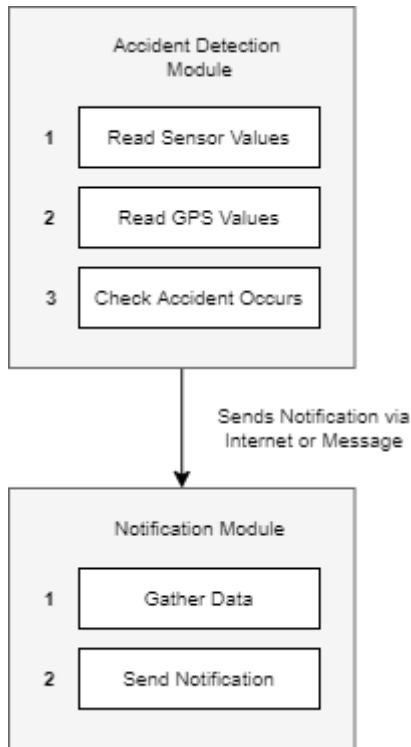


Figure 3.11. Accident Detection and Notification Module Overview

### 3.13 Accident Detection Module

Accident detection module is the initial component of this architecture. This module consists of several hardware components connected to an Arduino UNO<sup>16</sup> platform. The platform is enabled with an Accelerometer Sensor, Vibration Sensor, and General Packet Radio Service (GPRS). The following Figure 3.12 shows the proposed method of the connectivity of hardware modules with Arduino platform.

---

<sup>16</sup> <https://www.arduino.cc/>

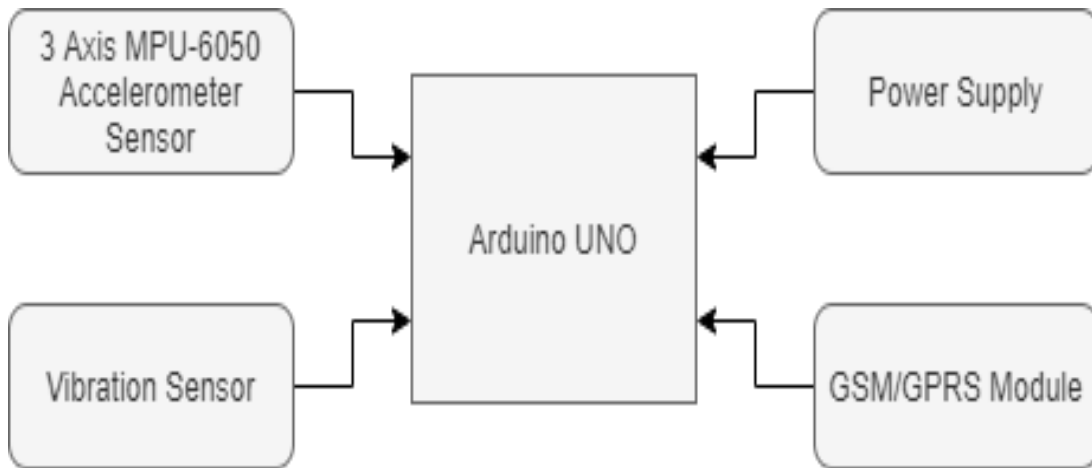


Figure 3.12. Arduino Platform High-level Overview

An accelerometer sensor is a sort of electrical sensor that measures the acceleration forces acting on an object. This is used to identify its location and track its movement. It will be responsible for monitoring the movements of the vehicle. An accelerometer is having three acceleration values such as  $X_{out}$ ,  $Y_{out}$ , and  $Z_{out}$ . These values are usually the value respect to a graph which contains values for X, Y and Z axis. The following Figure 3.13 shows that how the axis of acceleration can be found from the top view of the sensor.

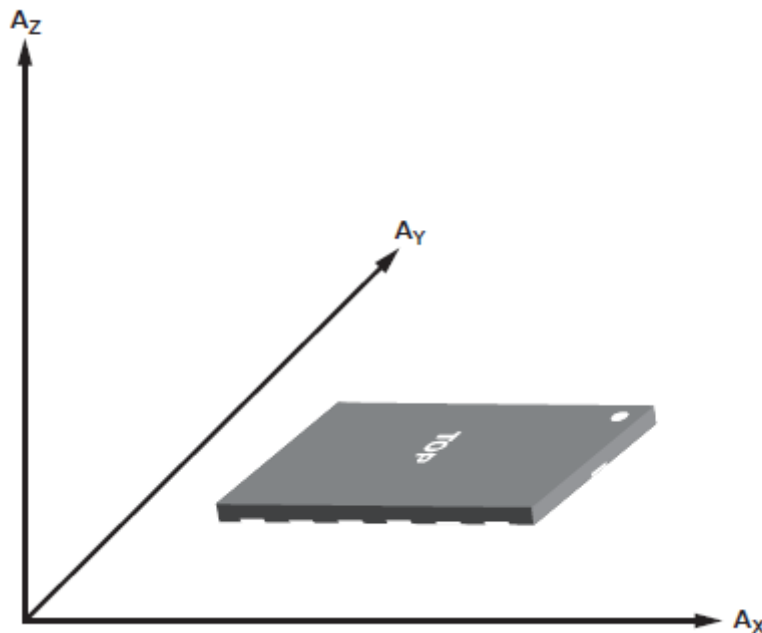


Figure 3.13. Example of Accelerometer sensor axis.<sup>17</sup>

<sup>17</sup> <https://www.engineersgarage.com/adx1345-accelerometer-sensor-how-to-use/>

The sensor is very sensitive where it can sense both static and dynamic accelerations. The following Figure 3.14 shows how the axis values will change according to some movements.

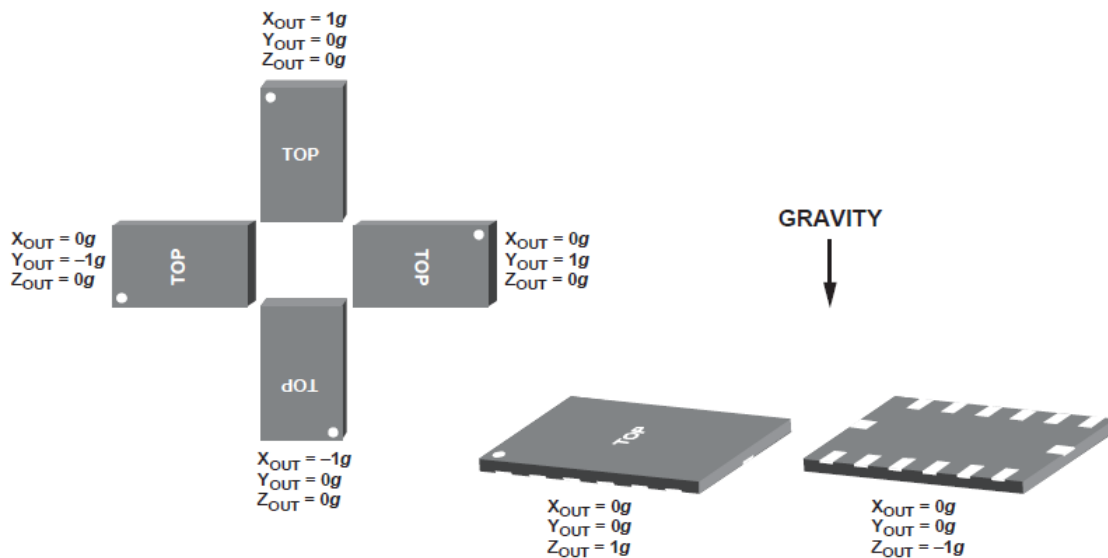


Figure 3.14. Accelerometer Outputs Respect to Several Movements <sup>18</sup>

Using the 3 axis values it is possible to calculate the Roll and Pitch values of the acceleration. Roll and Pitch values are the values of rotation around X-axis and rotation around Y-axis. These values will be used to determine an accident. The vibration sensor will be observing whether a vibration is available in the vehicle. And GPS will be tracking the location of the vehicle while moving. Once these values are available, they will be passed to the next module to determine whether an accident occurs or not. Each vehicle will contain average values for accelerometer sensor when there is a normal driving. But if anything, unwanted happens such as a crash or collision to the vehicle there will be a drastic change in the values of accelerometer sensor and the vibration sensor. That means there will be a drastic change on the roll values and pitch values of the accelerometer sensor. And the vibration sensor will indicate a high vibration as well. So once a drastic change is available, the average and current values will be compared with a pre-defined threshold value to suspect that an accident occurred. But there can be other situations such as a heavy sudden break where the vibration is available in the vehicle, or an accelerometer measures some values. Therefore, a confirmation prompt will be shown to the driver via the mobile application, and it will count for the next immediate 20 seconds. If there is no response from the driver the notification module will get triggered automatically.

<sup>18</sup> <https://www.engineersgarage.com/adx1345-accelerometer-sensor-how-to-use/>



The following algorithm in Figure 3.15 shows how the Accident detection module determines an accident.

---

**Algorithm 2:** Example code of accident detection

---

```

Data: Sensor Values
Output: Accident detection and notification
1 initialRollValue, initialPitchValue, timeSeconds = 0
2 suspicious, accidentDetected = FALSE
3 Function GetAccValues():
4 | return rollValue, pitchValue
5 Function GetVibrationValue():
6 | return vibrationValue
7 Function GetAccelerometerThreshold():
8 | return thresholdValue
9 Function SendUserPrompt():
10 | send a user prompt to determine the real state
11 Function GetUserResponse():
12 | send a user prompt to determine the real state
13 Function CalculateSuspicious(rollValue, pitchValue,
    initialRollValue, initialPitchValue):
14 | return suspicious
15
16 while timeSeconds ≤ 30 do
17 | initialRollValue, initialPitchValue = GetAccValues()
18 | timeSeconds ++
19 end
20 while TRUE do
21 | vibrationValue = GetVibrationValue()
22 | if vibrationValue == HIGH then
23 | | rollValue, pitchValue = GetAccValues()
24 | | suspicious = CalculateSuspicious(rollValue, pitchValue,
    initialRollValue, initialPitchValue)
25 | end
26 | else
27 | | suspicious = FALSE
28 | end
29 | if suspicious == TRUE then
30 | | START AlarmTimer
31 | | SET AlarmTimer = 20Seconds
32 | | SendUserPrompt()
33 | | while AlarmTimer > 0 do
34 | | | if GetUserResponse() == FALSE then
35 | | | | accidentDetected = FALSE
36 | | | end
37 | | | AlarmTimer = AlarmTimer - 1
38 | | | end
39 | | | if AlarmTimer == 0 then
40 | | | | accidentDetected = TRUE
41 | | | end
42 | | end
43 | | if accidentDetected == TRUE then
44 | | | Trigger notification module to send notification
45 | | end
46 end

```

---

Figure 3.15. Algorithm for Accident Detection and Notification

The following algorithm in Figure 3.16 shows the pseudocode segment of the *CalculateSuspicious* function.

---

**Algorithm 3:** Example code segment of accident suspicious function

---

```

1 Function GetThresholdValue():
2 |   Get the threshold value return thresholdValue
3
4 Function CalculateSuspicious(rollValue, pitchValue,
   initialRollValue, initialPitchValue):
5 |   rollTrue = FALSE
6 |   pitchTrue = FALSE
7 |   thresholdValue = GetThresholdValue()
8 |   if rollValue ≤ 0 then
9 |     | if rollValue < (initialRollValue - thresholdValue) then
10 |      | | rollTrue = TRUE;
11 |      | end
12 |     end
13 |   else
14 |     | if rollValue > (initialRollValue + thresholdValue) then
15 |      | | rollTrue = TRUE;
16 |      | end
17 |     end
18 |   if pitchValue ≤ 0 then
19 |     | if pitchValue < (initialPitchValue - thresholdValue) then
20 |      | | pitchTrue = TRUE;
21 |      | end
22 |     end
23 |   else
24 |     | if pitchValue > (initialPitchValue + thresholdValue) then
25 |      | | pitchTrue = TRUE;
26 |      | end
27 |     end
28 |   if pitchValue == TRUE OR rollTrue == TRUE then
29 |     | return TRUE
30 |   end
31 |   return FALSE
32

```

---

Figure 3.16. Algorithm for Suspicious Detection Function

Putting everything together the following Figure 3.17 shows the exact implementation of the IoT Arduino platform. Components such as GSM/GPRS module, Accelerometer and Vibration Sensor has been connected to the Arduino UNO board.

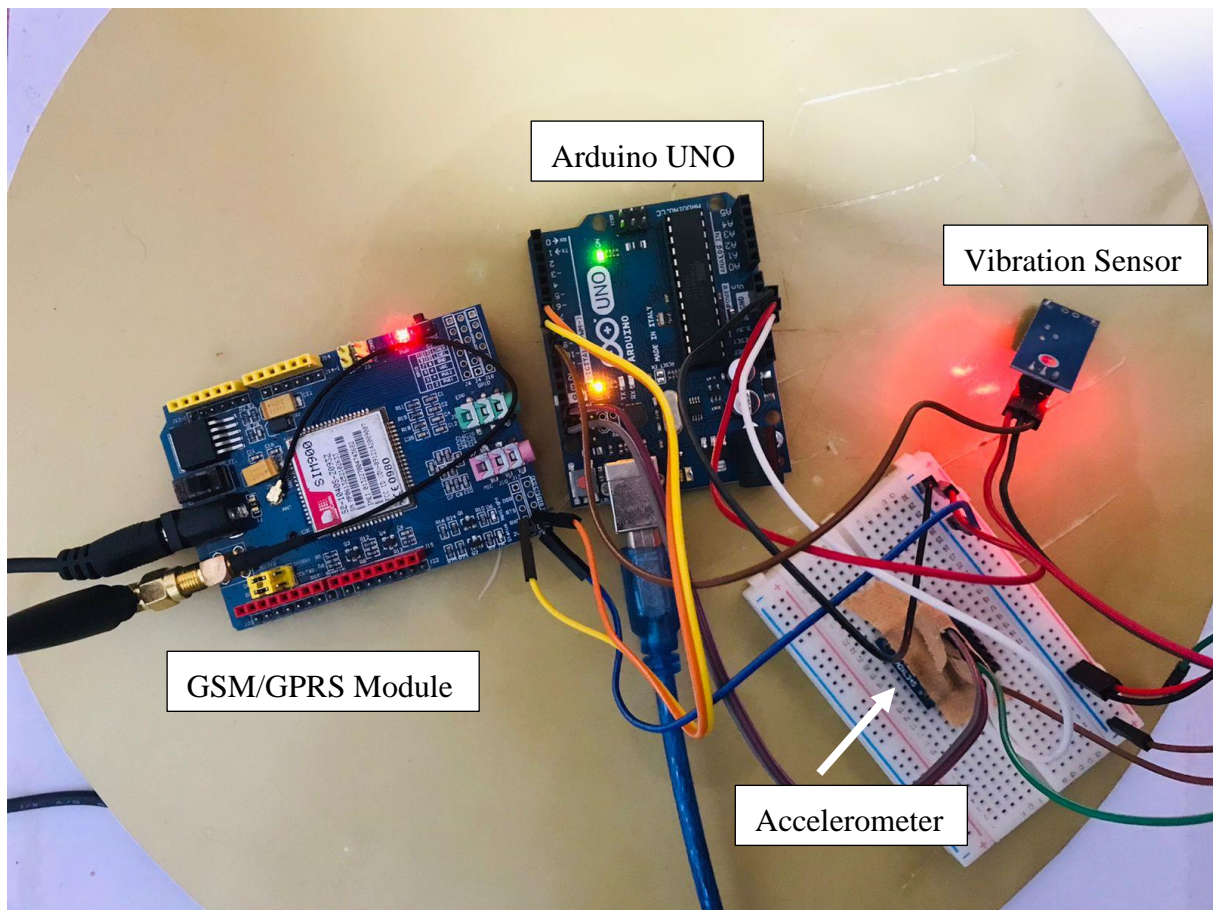


Figure 3.17. Arduino Platform of Accident Detection Module

### 3.14 Notification Module

Once an accident got identified the Notification module will gather the data including the GPS location of the vehicle and send a notification to close circles such as family members, relatives, friends of the driver, etc. Such notification can be a notification via the mobile app. The driver needs to add his contacts as a close circle via the mobile app. Then only the notification will work as expected.

### 3.15 Integration of Modules

Even though the CrashFree system has high level architecture with two major modules, these modules are interconnected with several components. All these interconnected components make CrashFree functionable. Drowsiness prediction and alert module is a separate module made available in the Google Colab environment. CrashFree system use a free tier version of the Colab environment. Image processing and related tasks needs more resources. Therefore, Colab was selected to host the drowsiness prediction module.

Accident detection and notification module incorporates many components as Arduino platform and backend services. Backend services of CrashFree is written on Node.js<sup>19</sup> using Express.js<sup>20</sup> framework and hosted in Heroku<sup>21</sup> platform. Both drowsiness prediction module and accident detection module have the connectivity of backend services. Firebase<sup>22</sup> has been used as the datastore and messaging service of CrashFree. The datastore has the direct link with the backend services. Finally, a mobile application is used to access all the services including drowsiness alert, accident alert etc. The backend service servers the necessary services to the mobile application. User can login to the mobile application and do the relevant things there. It is very important to click the start driving option from the mobile application to begin with all process. Once the driving is completed user can again go for stop driving option.

### **3.16 Summary**

This chapter discussed about the problem analysis and approach of the proposed solution. This chapter also explains and illustrates the top-level architecture of the proposed solution and the functions designed for each module. For each of the modules several tools and techniques were used. In addition to that needed requirements and constraints of the system are also mentioned in this chapter.

---

<sup>19</sup> <https://nodejs.org/en/>

<sup>20</sup> <https://expressjs.com/>

<sup>21</sup> <https://www.heroku.com/>

<sup>22</sup> <https://firebase.google.com/>

# CHAPTER 4 - EVALUATION AND RESULTS

## 4.1 Introduction

The previous chapter provides a detailed explanation of the methodology of the proposed solution. This chapter describes the evaluation and results of the system. Since the proposed solution contains two major modules, the evaluation of both modules will be conducted separately. The techniques of evaluation were chosen depending on the functioning and type of the system.

## 4.2 Evaluation of Drowsiness Detection and Alert

The evaluation process of this module begins with the evaluation of the prediction models. As mentioned above in the methodology section, CrashFree contains three prediction models to determine the state of eyes, mouth, and face. The developed module was evaluated using public and private video and image datasets.

### 4.2.1 Datasets

#### 1. UTA Real-Life Drowsiness Dataset

The University of Texas at Arlington Real-Life Drowsiness Dataset (Ghoddosian et al., 2019) It was created to detect drowsiness in multiple stages, addressing not only severe and clearly apparent situations, but also subtle cases where tiny expressions are the discriminative variables. Detection of these modest situations may be crucial for identifying sleepiness at an early stage, allowing drowsiness prevention systems to be activated. Because subtle micro-emotions of sleepiness have physiological and instinctual roots, actors who appear to be drowsy may struggle to authentically mimic such expressions. The RLDD collection is made up of about 30 hours of RGB footage from 60 healthy people. A total of 180 films were acquired for each participant, one for each of three separate classes: alertness, low vigilance, and drowsiness. Each footage has 10 minutes runtime and is classified as three classes: alert which is labeled as 0, low vigilance which is labeled as 5 and drowsiness which is labeled as 10. The labels were supplied by the participants depending on their main state at the time of filming each video. This dataset is widely used to test different kind of applications related to the drowsiness problem. Since CrashFree uses a behavioral approach to predict drowsiness this dataset will be useful when testing the efficiency of drowsiness prediction model.

## 2. Other participants dataset

This dataset is a combination of public datasets which is collected from Kaggle and private datasets which is created by 10 volunteers to make this research project useful. Each participants have two labeled categories such as alert state and drowsy state. Each category consists of 100 images including several conditions such as different face angles, face with spectacles and face without spectacles etc. There are 2000 images in total with several features. This dataset will be a used to evaluate the accuracy of the drowsiness prediction models.

### 4.2.2 Evaluation Methods

The following measurements were used to calculate the accuracy level of the three models and to calculate precision and recall. Accuracy is one of the benchmarks to evaluate the classification models. Accuracy refers to the percentage of correct predictions made by the model. The following equation is the formal definition of accuracy:

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

On the other hand, accuracy can also be calculated in terms of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN) as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision denotes the Fraction of retrieved instances that are relevant, and Recall denotes the Fraction of relevant instances that are retrieved.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

### 4.2.3 Results and Evaluations

The results of the testing performed for each of the models are described in the following sub sections. The equations used other than the described equations above are denoted in each of the sub section.

#### *Model 1 – Eye State Prediction*

The following Figure 4.1 describes about the accuracy and loss of the eye state prediction model while doing the training. After the training completion the model shows a validation accuracy of 98.9% and validation loss of 3%.

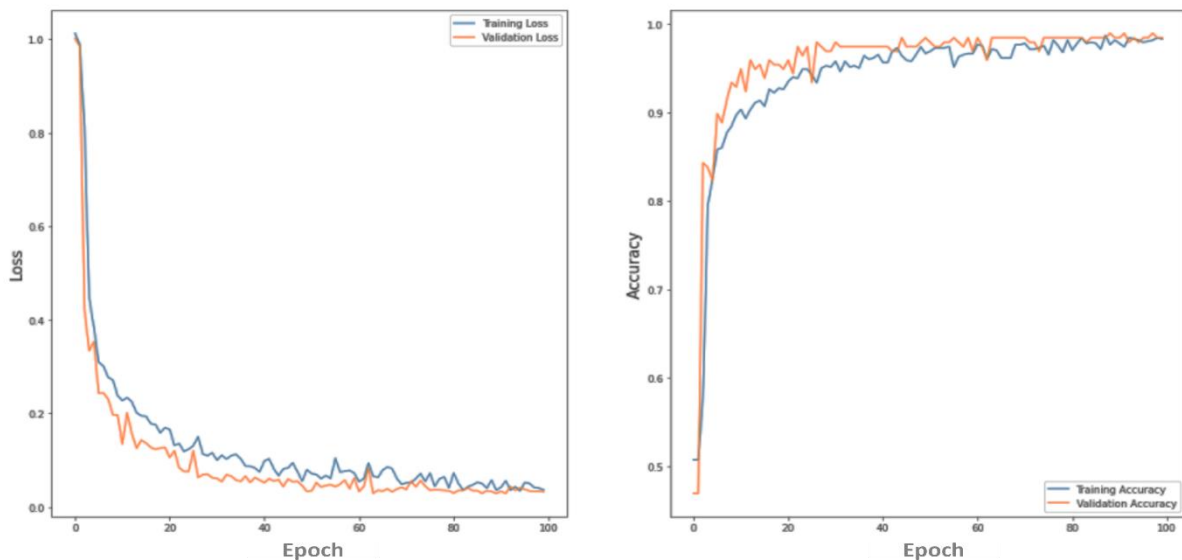


Figure 4.1. Eye State Model Accuracy and Loss

Furthermore, around ~200 sample frames were taken from the dataset to the testing of the eye state prediction model. Taking the total number of image samples input as 100% following percentages were calculated.

1. Correctly recognized eye state.
2. Incorrectly recognized eye state.

The number of correctly recognized eye states has taken as a measure of the accuracy. Therefore, the following equations were used to achieve that.

$$\text{Correctly recognized eye state} = \frac{\text{Number of correctly recognized eye state}}{\text{Total number of eye image frames}}$$

$$\text{Incorrectly recognized eye state} = \frac{\text{Number of incorrectly recognized eye state}}{\text{Total number of eye image frames}}$$

According to the eye state prediction model's testing the following results were obtained. The following Figure 4.2 shows the confusion matrix of the eye state model's prediction.

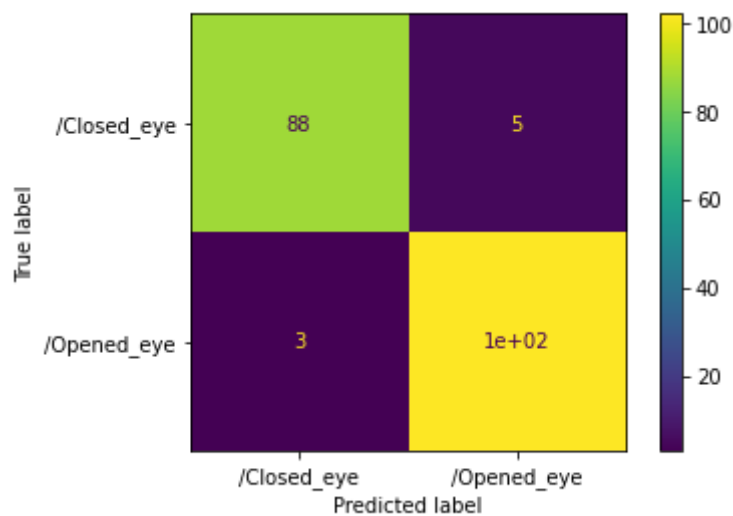


Figure 4.2. Eye State Model Confusion Matrix

According to the results the eye state prediction model predicts 88 frames as Closed eye and 105 frames as Opened eye which gives the average accuracy of 96%. The following Table 4.1 shows the summarized results of the model.

Table 4.1. Summary of Eye State Model

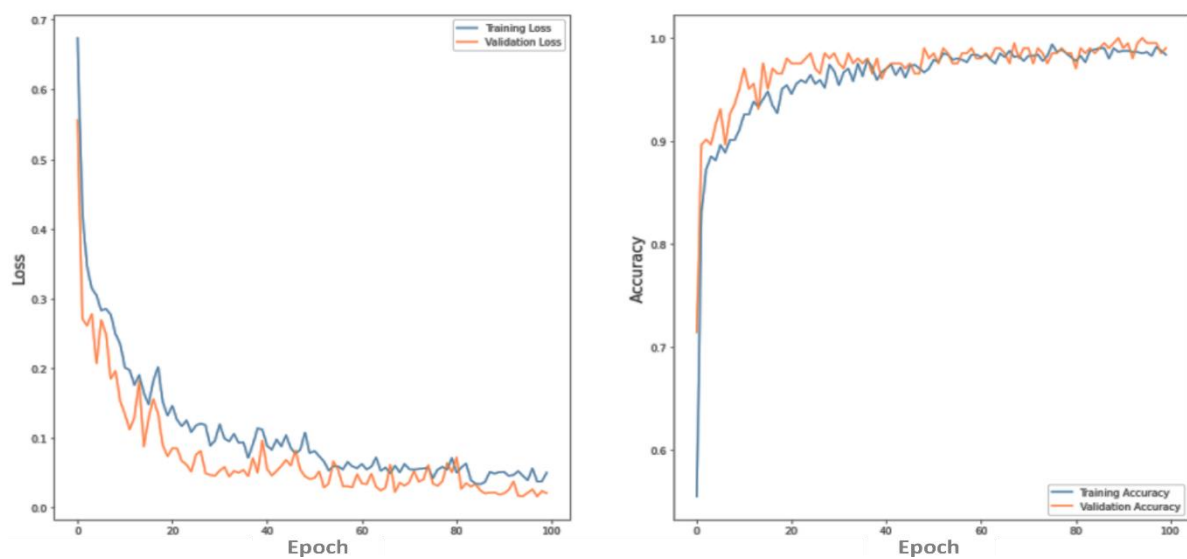
Total number of frames/images	198
Correctly recognized eye state	190
Incorrectly recognized eye state	8
Accuracy	96%
Precision	94.6%
Recall	96.7%



The eye state prediction model has the precision as 94.6%. That means when it predicts an eye state as closed, it is correct 94.6% of the time. On the other hand, the model has the recall as 96.7% as well. That means it correctly identifies 96.7% of all closed eyes.

### ***Model 2 – Mouth State Prediction***

The following Figure 4.3 describes about the accuracy and loss of the mouth state prediction model while doing the training. After the training completion the model shows a validation accuracy of 99.5% and validation loss of 4%.



*Figure 4.3. Mouth State Accuracy and Loss*

Another ~200 sample frames which contains opened mouth and closed mouth were taken from the dataset to the testing of the eye state prediction model. Taking the total number of image samples input as 100% following percentages were calculated.

1. Correctly recognized mouth state.
2. Incorrectly recognized mouth state.

The number of correctly recognized eye states has taken as a measure of the accuracy. Therefore, the following equations were used to achieve that.

$$\text{Correctly recognized mouth state} = \frac{\text{Number of correctly recognized mouth state}}{\text{Total number of mouth image frames}}$$

$$\text{Incorrectly recognized mouth state} = \frac{\text{Number of incorrectly recognized mouth state}}{\text{Total number of mouth image frames}}$$

According to the mouth state prediction model's testing the following results were obtained. The following Figure 4.4 shows the confusion matrix of the mouth state model's prediction.

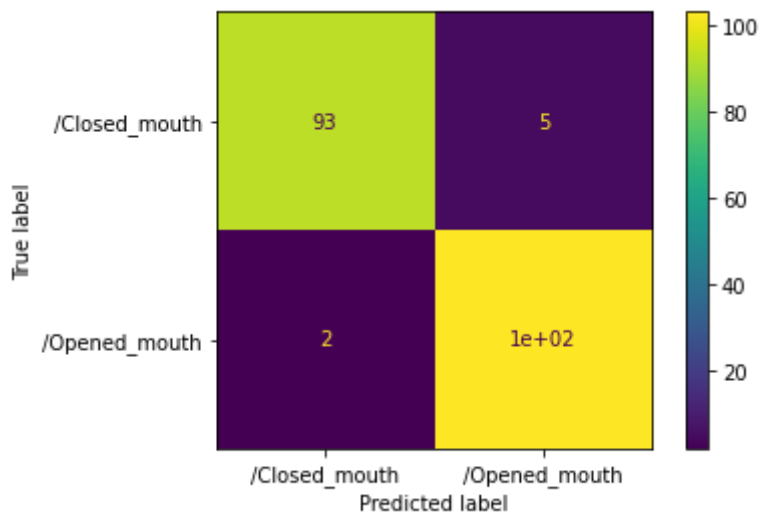


Figure 4.4. Mouth State Model Confusion Matrix

According to the results the mouth state model predicts 93 frames as Closed mouth and 103 frames as Opened mouth which gives the average accuracy of 97%. The following Table 4.2 shows the summarized results of the model.

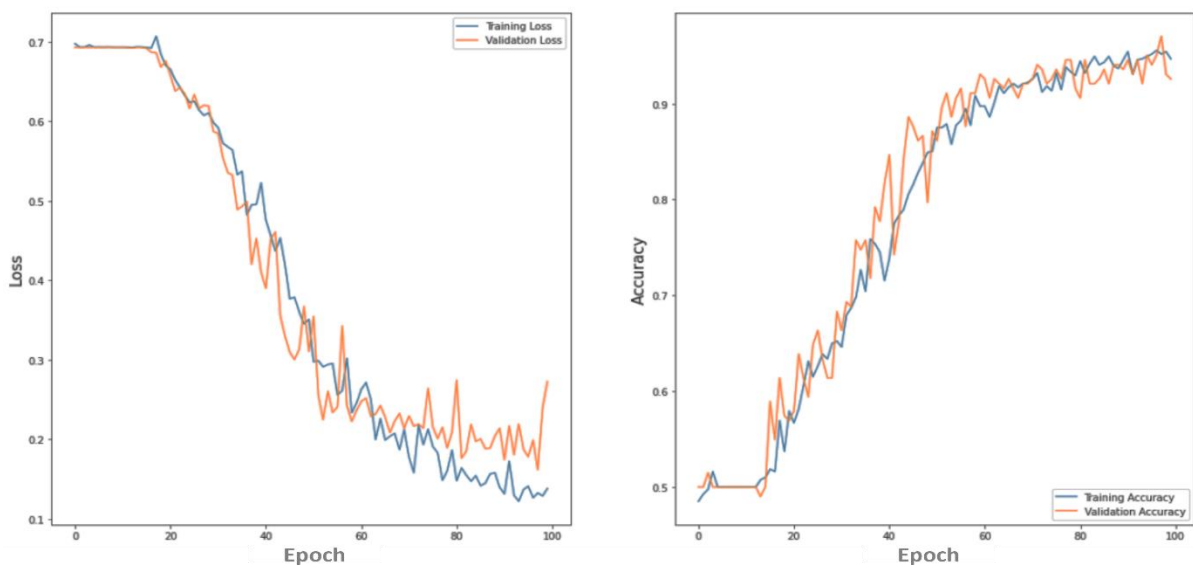
Table 4.2. Summary of Mouth State Model

Total number of frames/images	203
Correctly recognized mouth state	196
Incorrectly recognized mouth state	7
Accuracy	97%
Precision	94.8%
Recall	97.8%

The mouth state prediction model has the precision as 94.8%. That means when it predicts a mouth state as closed, it is correct 94.8% of the time. On the other hand, the model has the recall as 97.8% as well. That means it correctly identifies 97.8% of all closed mouths.

### ***Model 3 – Face State Prediction***

The following Figure 4.5 describes about the accuracy and loss of the face state prediction model while doing the training. After the training completion the model shows a validation accuracy of 93% and validation loss of 2%.



*Figure 4.5. Face State Accuracy and Loss*

Another ~200 sample frames which contains drowsy face and alert face were taken from the dataset to the testing of the face state prediction model. Taking the total number of image samples input as 100% following percentages were calculated.

3. Correctly recognized face state.
4. Incorrectly recognized face state.

The number of correctly recognized eye states has taken as a measure of the accuracy. Therefore, the following equations were used to achieve that.

$$\text{Correctly recognized face state} = \frac{\text{Number of correctly recognized face state}}{\text{Total number of face image frames}}$$

$$\text{Incorrectly recognized face state} = \frac{\text{Number of incorrectly recognized face state}}{\text{Total number of face image frames}}$$

The following results were obtained by testing the face state model's prediction. The following Figure 4.6 shows the confusion matrix of the face state model's prediction.

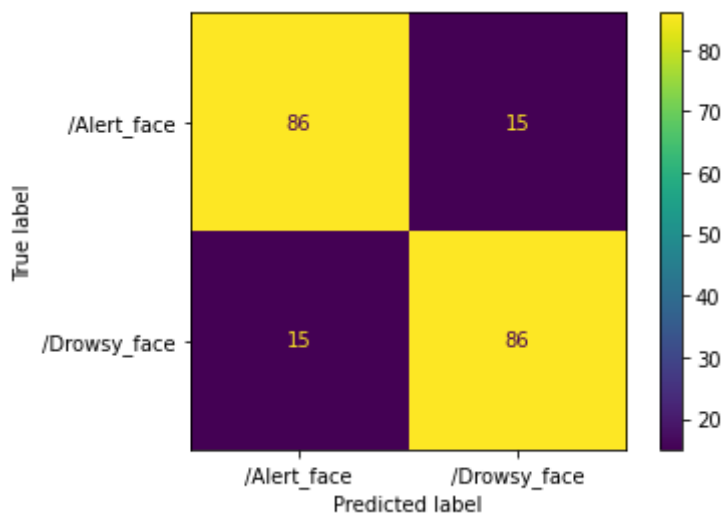


Figure 4.6. Face State Model Confusion Matrix

According to the results the mouth state model predicts 86 frames as Alert face and 86 frames as Drowsy face which gives the average accuracy of 85%. The following Table 4.3 shows the summarized results of the model.

Table 4.3. Summary of Face State Model

Total number of frames/images	202
Correctly recognized mouth state	172
Incorrectly recognized mouth state	30
Accuracy	85%
Precision	85.1%
Recall	85.1%

The mouth state prediction model has the precision as 85.1%. That means when it predicts a face state as alert, it is correct 85.1% of the time. On the other hand, the model has the recall as 85.1% as well. That means it correctly identifies 85.1% of all alert faces. The following Graph shown in Figure 4.7 summarizes the average evaluation results of three models.

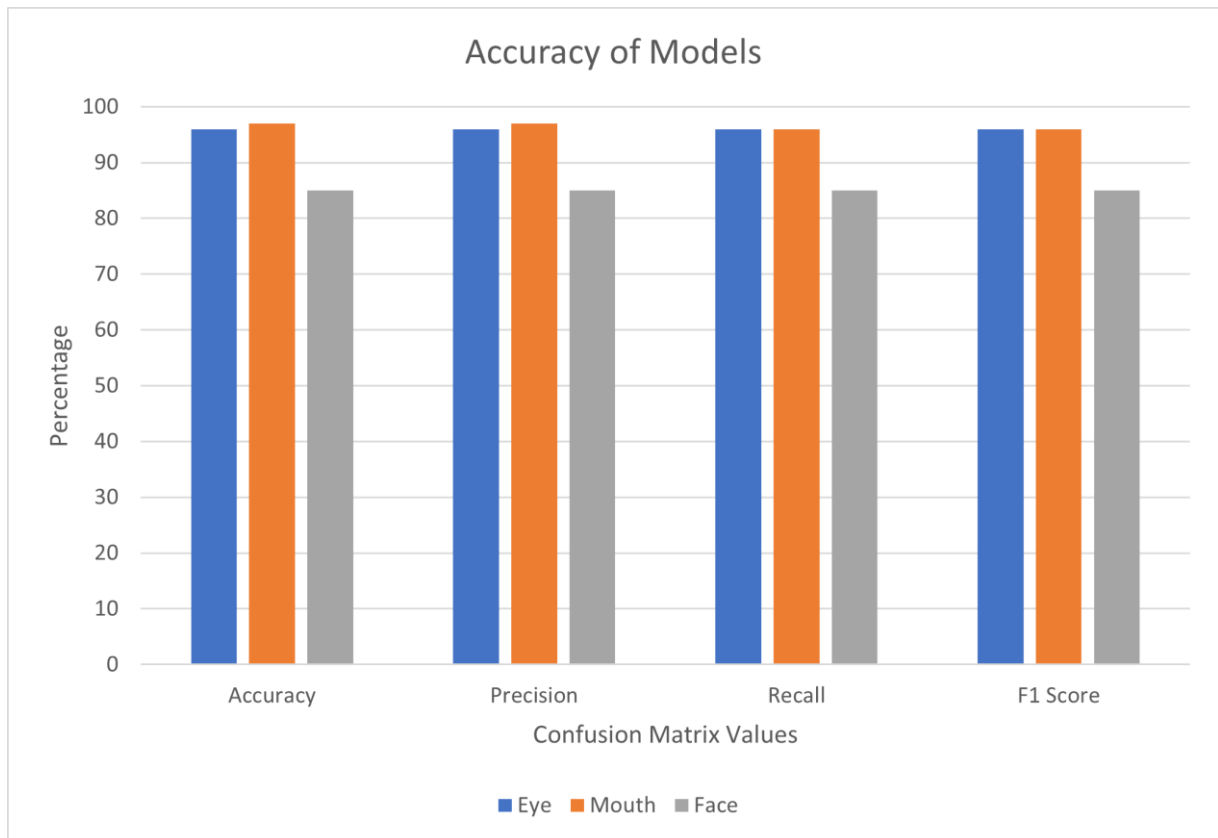


Figure 4.7. Average Evaluation Results of Models

Apart from the evaluation of the models it is necessary to evaluate whether the whole component works well with all the conditions. It is necessary to evaluate whether the cropped elements are being cropped correctly. The combination of closed eye and drowsy face or combination of opened mouth and drowsy face or combination of opened mouth, closed eye and drowsy face indicates that the driver is drowsy.

The following Figures 4.8 – 4.15 show the results of extraction of face, eye, and mouth from a frame. And it shows the respective states of face, eye, and mouth. Comparing each state, the final decision was taken.

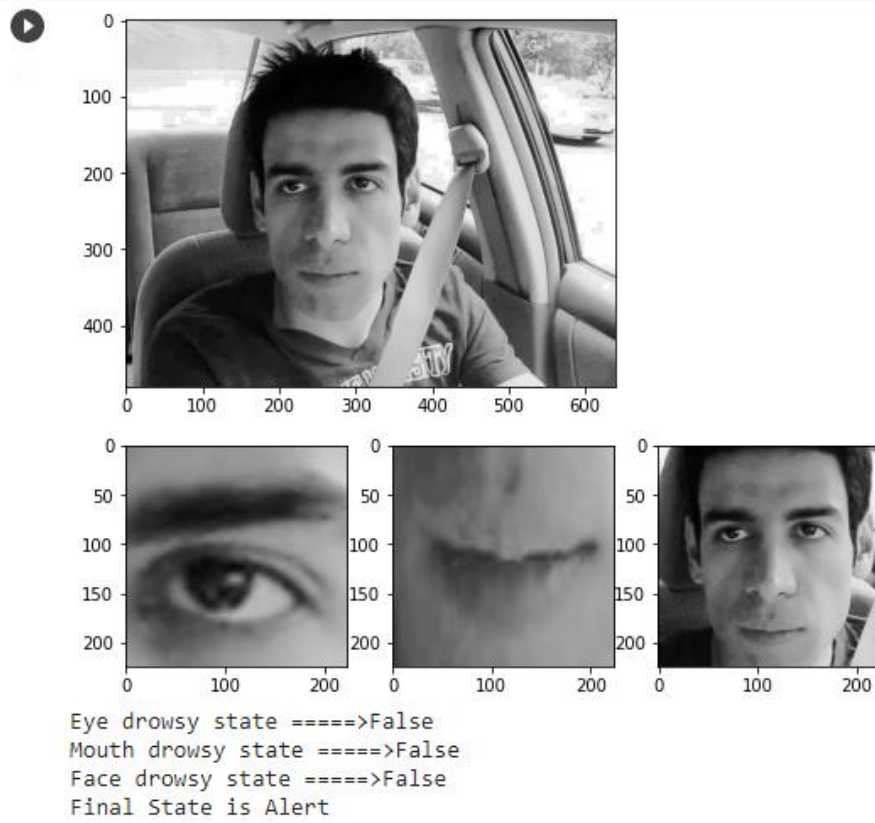


Figure 4.8. Frame of Alert State – 1

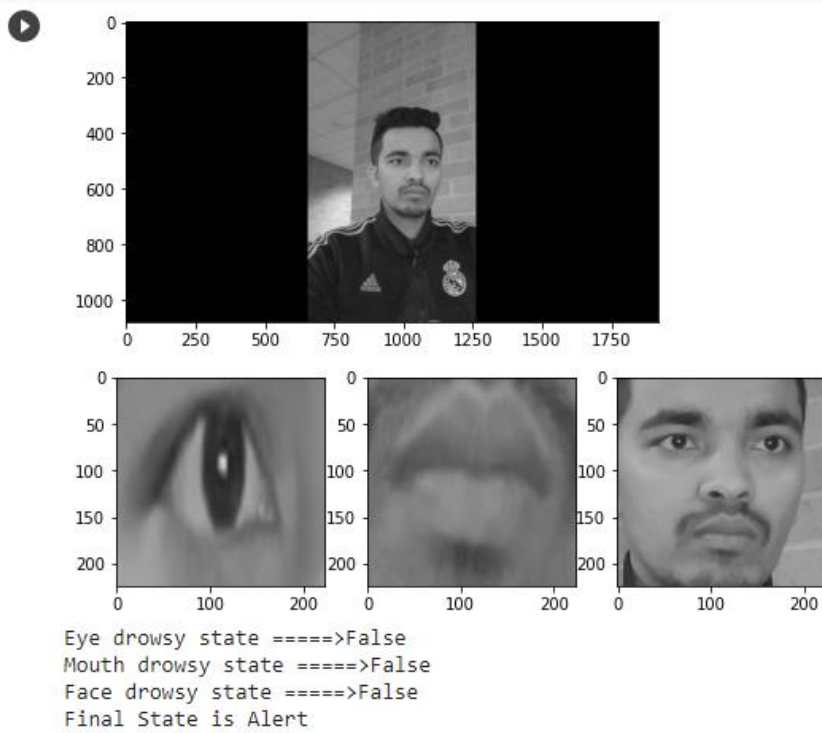


Figure 4.9. Frame of Alert State - 2



Figure 4.10. Frame of Alert State with Eyeglasses

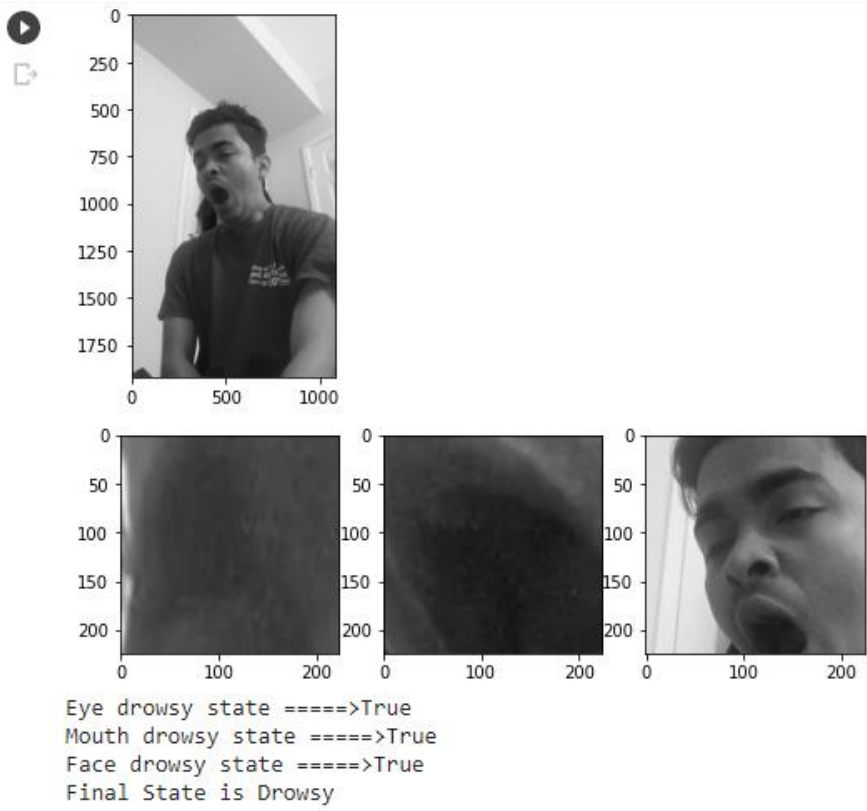


Figure 4.11. Frame of Drowsy State - 1

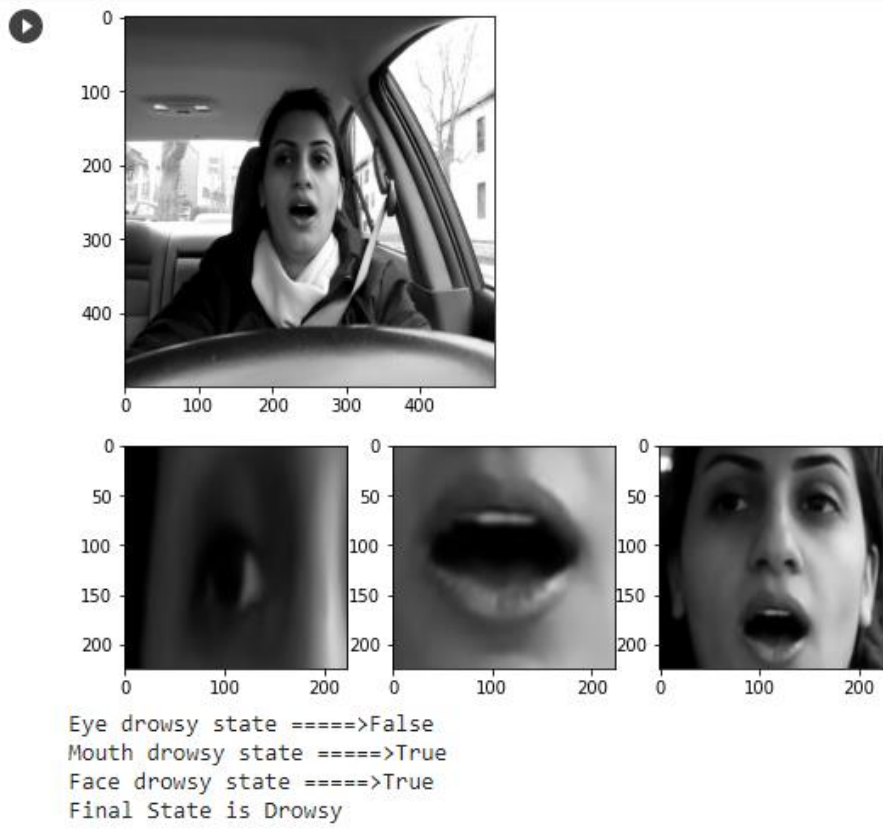


Figure 4.12. Frame of Drowsy State - 2

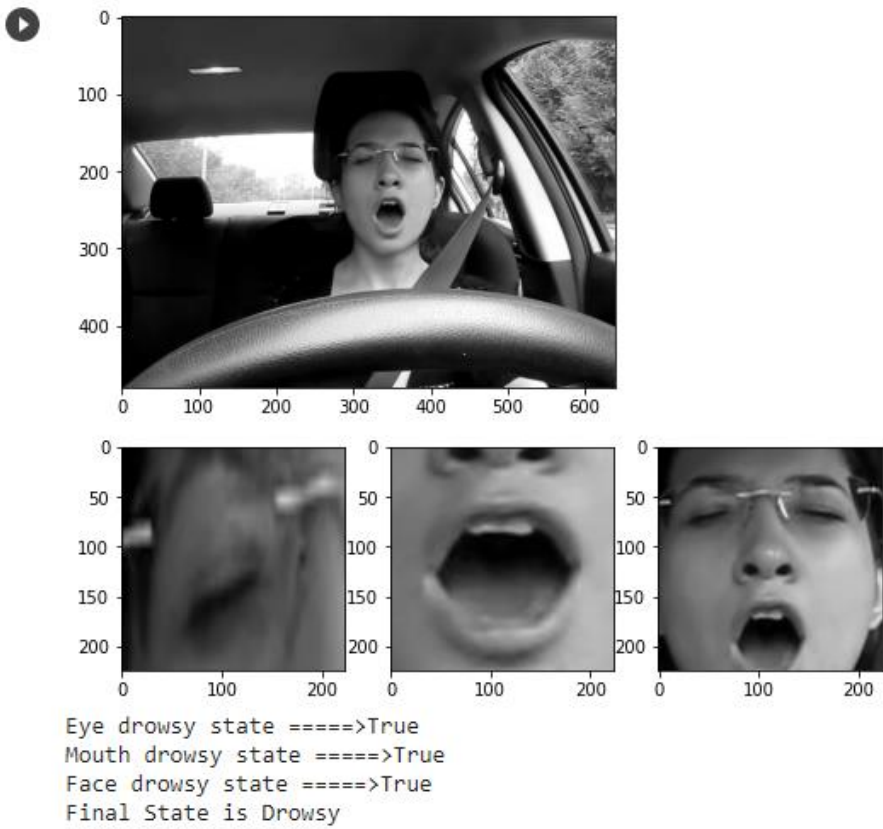


Figure 4.13. Frame of Drowsy State with Eyeglasses



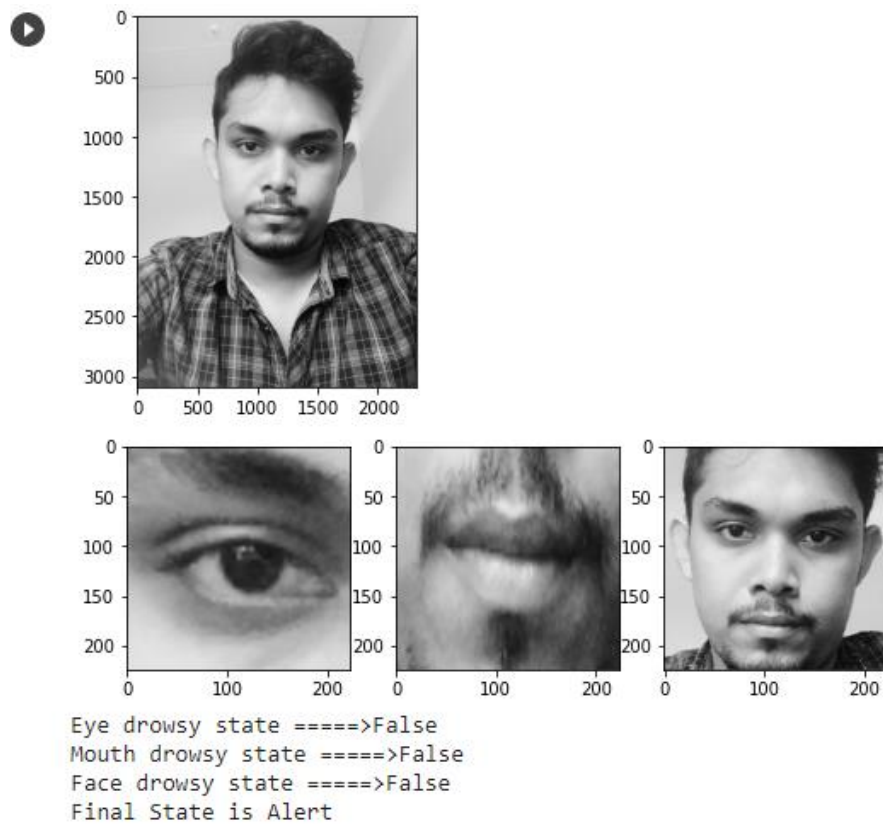


Figure 4.14. Frame of Alert State in a Low Light Environment

The prediction was wrong in certain scenarios like low light environment. The following Figure 4.15 shows an incorrect prediction in a low light environment. Eye state is predicted as drowsy where it is alert.

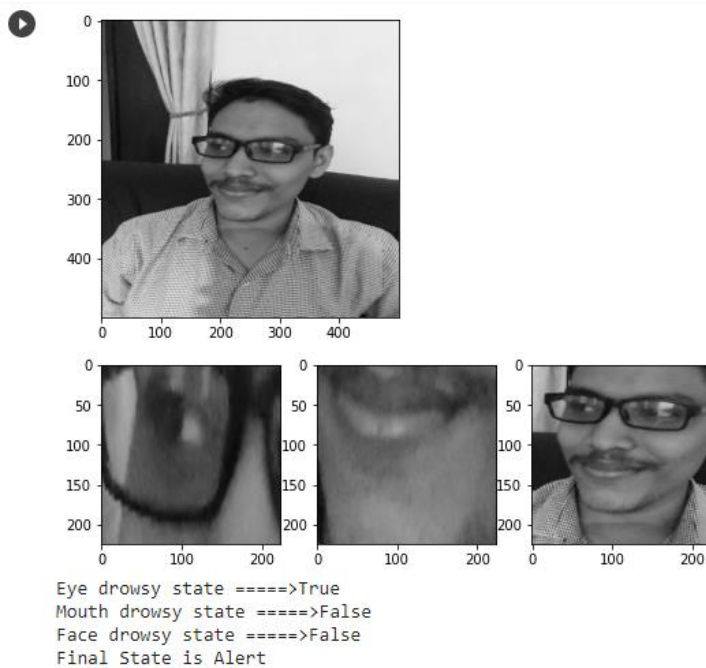


Figure 4.15. Frame of Alert State with Eyeglasses in a Low Light Environment

The following Figure 4.16 shows the actual alert which will be sent to the mobile application. This will be an alarm which will alert the user with a sound.

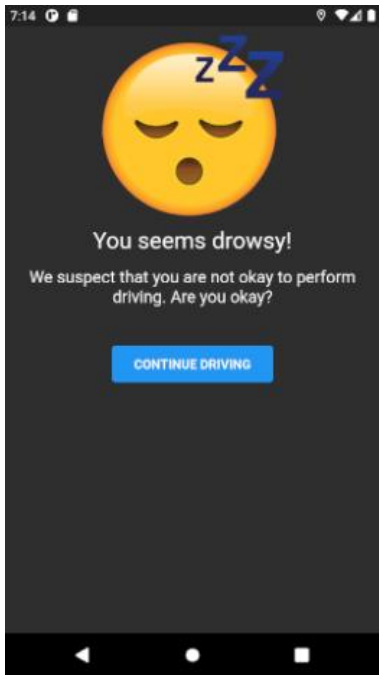


Figure 4.16. Driver Drowsiness Alert via Mobile App

After the results from the prediction model is identified, if the results states that driver is drowsy the alert module should be activated and give an alert to the user/driver via the CrashFree mobile application. Thus, the following results shown in the Table 4.4 was obtained based on the evaluation of whole drowsiness prediction module. Either face region and eye region or face region and mouth region should be identified properly to move forward with the process otherwise the results of the experiment will be labeled as “Fail”. If a drowsiness is identified, then the driver will be notified with an alarm via mobile application. The whole correct scenario was labeled as “Pass”.

It was planned to use Transfer Learning to improve the accuracy of the models after the evaluation. But it is not possible to go with. Because transfer learning models use three dimensional images where CrashFree uses only one-dimensional images. Based on the experiments the module performed with a good outcome where most of the test cases got passed under many circumstances. The final overall accuracy of the module was obtained as ~93% based on the results.

Table 4.4. Experiment results of drowsiness prediction and alert module

#	Face Cropped	Face State	Eye Cropped	Eye State	Mouth Cropped	Mouth State	Drowsiness	Alarm	Result
001	T	T	T	F	T	T	T	T	Pass
002	T	T	T	T	T	F	T	T	Pass
003	T	F	F	F	F	F	F	F	Fail
004	T	T	T	T	T	T	T	T	Pass
005	F	F	F	F	F	F	F	F	Fail
006	T	T	T	T	F	F	T	T	Pass
007	T	T	T	T	T	F	T	T	Pass
008	T	T	T	F	T	F	F	F	Pass
009	T	T	T	F	T	T	T	T	Pass
010	T	F	T	F	T	T	F	F	Pass
011	T	F	T	T	T	F	F	F	Pass
012	T	T	T	F	T	T	T	T	Pass
013	T	T	T	T	T	T	T	T	Pass
015	T	F	T	F	T	F	F	F	Pass
016	T	F	T	F	T	F	F	F	Pass
017	T	T	T	T	T	F	T	T	Pass
018	T	T	T	F	T	T	T	T	Pass
019	T	T	T	T	T	T	T	T	Pass
020	T	F	T	F	F	F	F	F	Pass
021	T	F	T	T	F	F	F	F	Pass
022	T	F	F	F	T	T	F	F	Pass
023	T	F	T	F	T	F	F	F	Pass
024	T	F	T	F	T	F	F	F	Pass
025	T	F	T	F	T	F	F	F	Pass
026	T	T	T	T	T	F	T	T	Pass
027	T	T	T	F	T	F	F	F	Pass
028	T	T	T	F	T	F	F	F	Pass
029	T	F	T	F	T	F	F	F	Pass
030	T	F	T	F	T	F	F	F	Pass
031	T	F	T	F	T	F	F	F	Pass
032	T	F	F	F	T	T	F	F	Pass
033	T	F	T	F	T	F	F	F	Pass
034	T	T	T	T	T	F	T	T	Pass
035	T	T	T	F	T	F	F	F	Pass

### 4.3 Evaluation of Accident Detection and Notification

Since this module is a prototype version, it is not possible to evaluate its performance in real time. But there are several theories and scenarios where we can say that this approach is a possible way to detect an accident on a vehicle. CrashFree uses two sensor approaches to determine an accident alert.

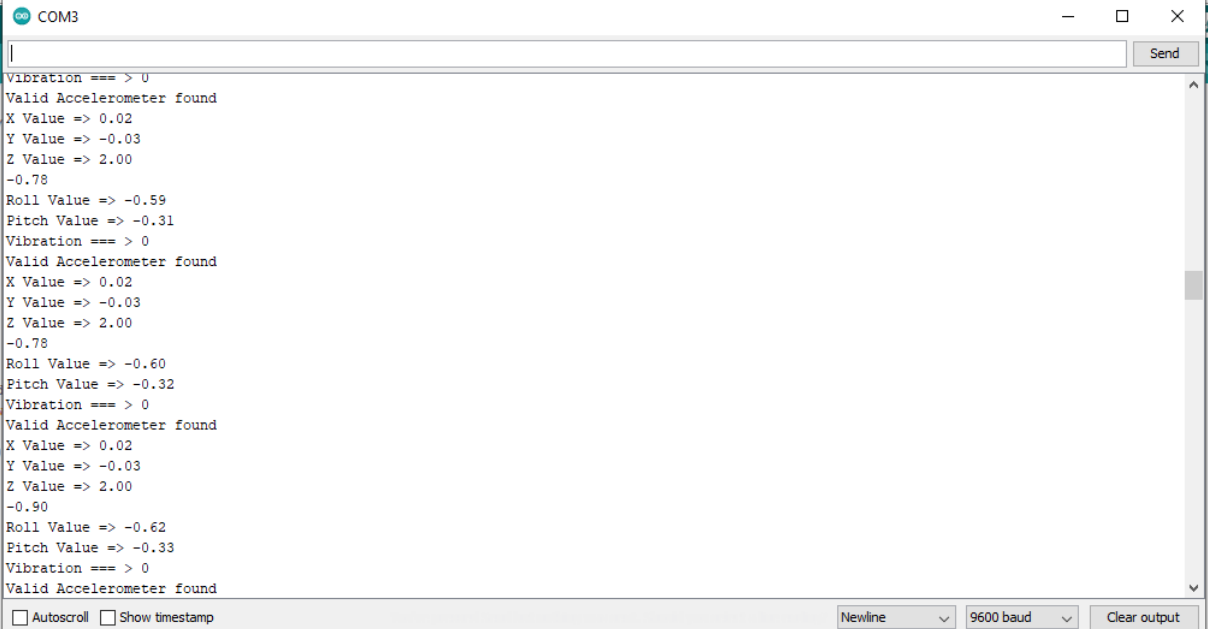
We can suspect that an accident occurred in the vehicle based on the following conditions.

1. Driver is driving the vehicle and running CrashFree mobile application.
2. Vibration sensors sense a high vibration in the vehicle.
3. Accelerometer indicates drastic change in the roll value or high pitch value compared to the average roll value and pitch values.

These three conditions should be full filled to suspect that an accident occurred. But this is not the final decision. Vehicle without drivers and vehicle which is parked somewhere will not be considered at this stage.

### 4.3.1 Experiments and Results

Different use cases have been tried out to evaluate the performance of this accident detection module. This module of the system was put to the test in two ways. Individual components were first subjected to unit testing. Then double-checked the components to see if there were any mistakes. Following then, the accident detection module of the system was put together using individual components. The module was then subjected to black box testing. The following Figure 4.17 shows how the sensor values are interpreted.



```
COM3
Vibration == > 0
Valid Accelerometer found
X Value => 0.02
Y Value => -0.03
Z Value => 2.00
-0.78
Roll Value => -0.59
Pitch Value => -0.31
Vibration == > 0
Valid Accelerometer found
X Value => 0.02
Y Value => -0.03
Z Value => 2.00
-0.78
Roll Value => -0.60
Pitch Value => -0.32
Vibration == > 0
Valid Accelerometer found
X Value => 0.02
Y Value => -0.03
Z Value => 2.00
-0.90
Roll Value => -0.62
Pitch Value => -0.33
Vibration == > 0
Valid Accelerometer found
```

Autoscroll  Show timestamp

Newline 9600 baud Clear output

Figure 4.17. Readings From Sensors

The following Tables 4.5, 4.6 and 4.7 show the experiments conducted to evaluate the accident detection module. It also shows the connectivity between different variables which is used during the experiments. The following values has been used to interpret the outputs.

- Vibration Sensor Value (VSV) – Readings from vibration sensor module.
- Accelerometer Sensor Threshold (AST) – A default value defined as a threshold.
- Accelerometer Sensor Normal Value (ASNV) – Readings from accelerometer when the vehicle is in the initial driving state. This has roll and pitch values which are constant.
- Accelerometer Sensor Current Value (ASCV) – Real time readings from the accelerometer when the vehicle is travelling. This also has roll and pitch values which will be getting updated time to time.
- Accident Suspected (AS) – A Boolean value which denotes whether an accident is suspected or not.
- Accident Suspect Alarm (ASA) – A Boolean value which denotes whether an accident suspect alarm initiated or not.

### Experiment 01 – Accident Suspect Alarm

Table 4.5. Accident Suspect Alarm - Experiment 1

	VSV	AST	ASNV		ASCV		AS	ASA
			Roll	Pitch	Roll	Pitch		
01	High	4	0.2	1.2	2.2	3.2	F	F
02	High	4	0.2	1.2	4.3	3.2	T	T
03	High	4	0.2	1.2	4.0	5.6	T	T
04	High	4	0.2	1.2	5.5	6.1	T	T

### Experiment 02 – Accident Suspect Alarm

Table 4.6. Accident Suspect Alarm - Experiment 2

	VSV	AST	ASNV		ASCV		AS	ASA
			Roll	Pitch	Roll	Pitch		
01	Low	4	0.2	1.2	2.3	3.4	F	F
02	Low	4	0.2	1.2	4.3	3.3	F	F
03	Low	4	0.2	1.2	4.5	5.6	F	F
04	Low	4	0.2	1.2	5.1	6.8	F	F

### Experiment 03 – Accident Suspect Alarm

Table 4.7. Accident Suspect Alarm - Experiment 3

	VSV	AST	ASNV		ASCV		AS	ASA
			Roll	Pitch	Roll	Pitch		
01	Low	5.5	0.5	0.2	2.3	6.8	T	T
02	Low	5.5	0.25	0.6	4.3	5.8	F	F
03	Low	5.5	0.3	0.8	6.0	4.2	T	T
04	Low	5.5	-0.5	0.12	10.2	7.1	T	T

Based on the above results, Further experiments have been conducted. The following experiment results in Table 4.8, 4.9 and 4.10 show how an accident is determined based on several accident suspect results. The interpretation of the outputs has the following values.

- Accident Alarm Timer (AAT) – A default timeout value which is used to show a popup alert to the driver. This value is normally set as 20 seconds. Thus, the alarm will be active for 20 seconds.
- Driver Acknowledgment (DA) – A Boolean value which denotes whether the driver acknowledged to the accident alert popup or not.
- Accident Detected (AD) – A Boolean value which interprets whether accident occurred or not. This value usually depends on the value of DA. If DA is FALSE, then this value will become TRUE.
- Notification to User (NTU) – A Boolean value which tells whether the notification is sent to the relevant parties or not.
- Notification Sent Time (NST) – A value which denotes the time taken to send notification to the relevant parties.

### Experiment 01 – Accident Notification

Table 4.8. Accident Notification - Experiment 1

	ASA	AAT	DA	AD	NTU	NST
01	F	20s	T	F	F	-
02	T	20s	F	T	T	5s
03	T	20s	F	T	T	4s
04	T	20s	T	F	F	-

## Experiment 02 – Accident Notification

Table 4.9. Accident Notification - Experiment 2

	ASA	AAT	DA	AD	NTU	NST
01	F	20s	F	F	F	-
02	F	20s	F	F	F	-
03	F	20s	F	F	F	-
04	F	20s	F	F	F	-

## Experiment 03 – Accident Notification

Table 4.10. Accident Notification - Experiment 3

	ASA	AAT	DA	AD	NTU	NST
01	T	20s	F	T	T	6s
02	F	20s	F	F	F	-
03	T	20s	F	T	T	3s
04	T	20s	T	F	F	-

The following Figure 4.18 shows the actual accident suspect alarm which will be sent to the driver. This timeout can be configured by the user according to their need.

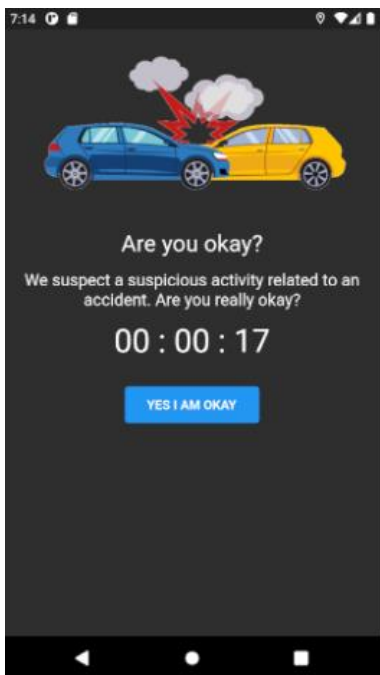


Figure 4.18. Accident Suspect Alarm to the Driver

If the driver fails to acknowledge then an automatic notification will send to the relevant parties. The following Figure 4.19 shows the notification which will be received by a relevant party.

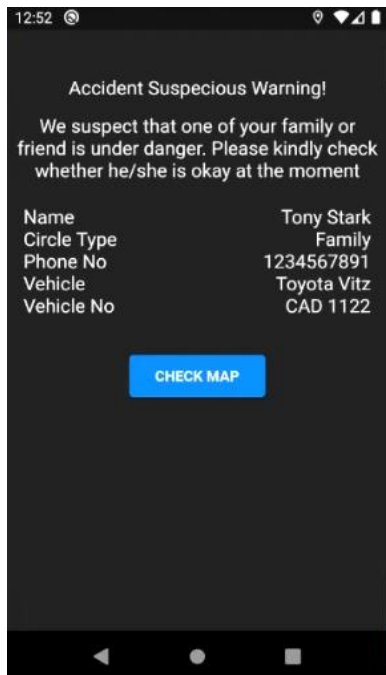


Figure 4.19. Accident Notification to a Close Circle

#### 4.4 Summary

This chapter discussed about the testing, performance evaluation and results of the proposed system. Different graphical illustrations and tables were used to describe the results. Component testing and system testing was used in the initial testing part. All the state detection models obtained a good number of accuracy while doing the evaluation. Eye state model, mouth state model, and face state model obtained an accuracy of 96%, 97% and 85% respectively. The accident detection module performed well under all necessary test cases. A black box testing has been performed to test the requirements of the system. The results also visually represented in this chapter.



# CHAPTER 5 - CONCLUSION AND FUTURE WORK

## 5.1 Introduction

The previous chapter evaluates the system with the results of testing performed. This chapter concludes the dissertation and states the further activities. First this chapter will discuss the contributions of the research. Then achievement of the objectives, whether it is up to the expected performance. Then the problems that were encountered will be discussed. As the next step, the limitations of the provided solution will be discussed with the help of the evaluation done in the previous chapter. Finally, further implementation of the system will be discussed.

## 5.2 Contributions

In a variety of ways, the study has aided the scientific community and the deep learning and IoT community. The major contribution of this study is the Drowsiness prediction system which is very helpful for many drivers. The findings of the research related to this domain is a CNN classifier with SVM can be used to predict the state of the driver to say whether driver is drowsy or not. In addition to that, three different models have been used to predict the state of eye, mouth, and face. Previously researchers have done classifications using either eye or mouth. But having a combination of eye, mouth and face states will give some better results. This also will enable future researchers to tryout some other combinations and prepare some another classification model which will be more accurate and efficient. In addition, validations for the findings made by previous researchers also provided. Therefore, it will be very useful carryout research by trying out some other different approach.

The next contribution is related to IoT and accident detection. The research has addressed an approach which can give an immediate solution when a driver face with an accident. Compared to the previous research, this study provides a two-sensor based approach to determine an accident. And adding a confirmation prompt will minimize the false rate of the system.

## 5.3 Achievements of Objectives

As an outcome all the objectives were achieved successfully and the aim of CrashFree is reflected by the implemented system. The first object was to select a suitable image classification method which can classify the facial states. The objective ended up with the selection of CNN classifier with SVM. Addition to that, suitable image processing techniques,

image enhancement techniques, and feature extraction technique was used. The evaluation results of drowsiness detection module state that the module achieved the aims with good amount of accuracy. Eye state model, mouth state model, and face state model obtained an accuracy of 96%, 97% and 85%. The model predicts every state of eyes, mouth, and face with a good amount of accuracy. Apart from that the final testing results of the whole module gave ~93% of success rate. But the prediction models had a lack of performance since it was hosted on Colab free tier environment. Also, the model's prediction was inefficient in some low light environments.

The next objective was to select a suitable IoT technique which can be used to detect accidents on a vehicle. Therefore, this objective came up with a solution to use two sensor-based Arduino platform. The evaluation of accident detection module shows that the approach is suitable to detect accidents on a vehicle. But, since this module is a prototype, it is not possible to evaluate this in a real environment. Based on the given inputs the component worked well. A real time monitoring system was developed to overcome the third objective. A perfect mobile application was developed to serve necessary functionalities such as drowsy alert, accident alert and notifications etc. In brief the CrashFree system contains components such as a drowsiness detection module which is available in Colab environment, accident detection module developed with Arduino platform, Backend server hosted in Heroku platform, Datastore available in Firebase and a mobile application. Moreover, the results sections shows that the selected approach and methods are sufficient to detect drowsiness and accident using the proposed solution.

## **5.4 Problems Encountered**

When developing the CrashFree system several problems have been encountered. The major problem was finding the proper datasets related to drowsiness to train the models. Several online sources have been studied and finally some open-source datasets have been collected for training purpose. And when developing the system many issues were encountered. The major issue was the platform for the image processing part. It is not that much easy to do any image processing task in a Personal Computer (PC) with normal configuration. The image processing task requires Graphics processing unit (GPU) enabled environment. Therefore, after trying out several approaches Google Colab has been chosen to do the image processing tasks. Because Colab offers an integrated environment to do machine learning and image processing tasks.

Initially it was planned to implement the system using chosen technologies. But when trying to do it practically, some technologies cannot be used together. Therefore, there was a delay to find out the appropriate alternative technologies to implement the system.

## **5.5 Limitations**

The deployment of the CrashFree System comes with several limitations that should be considered. The limitations are as follows,

- SIM with a data plan is required.

To provide real time information from the vehicle to server, GSM/GPRS uses mobile sim data. As a result, in this situation, the data plan should be activated, and recharging is required to maintain an unbroken connection with the server.

- Sometimes, the system was unable to detect the drowsiness correctly in the low light environment. Therefore, a mechanism should be available in the vehicle to provide some light.
- The driver who uses the mobile applications are required to respond the accident suspicious alerts if they are not in a trouble. They system assumes that they will respond with in the given time.
- The drowsiness prediction module is only available in the free tier of Colab. Thus, the real-time functioning of drowsiness prediction might be slow. Because Colab only provides only a certain amount of time to perform image processing tasks.
- The accident detection module is not testable in the real-world environment.

## **5.6 Future works**

As the future work in the project, this project can be expanded in various ways. The drowsiness detection module only considers three states such as eye state, mouth state and face state. But there can be other possibilities as well to check the drowsiness from a driver. Head position, eye blink rate etc. can be used to determine the drowsiness level of the driver. Therefore, the system can be further improved by considering these facts. And the accident detection module was developed using a two-sensor based approach. But technically the further improvement of the system can consider using other possible ways including currently available sensors. The limitations shows that the prediction does not works well with a low light environment. This shows that there is a necessity to train the models with more datasets which has several features

including low light images. Also, the further works will address and provide the solutions to the limitations mentioned above. And further improvements related to the performance of the system will be considered in the future.

## **5.7 Summary**

This chapter summarizes the work that has been done up to now and described further work to be done. And the chapter includes the contributions and problems encountered. Limitations of the system have been discussed as well.

# APPENDICES

## Appendix A: User Interfaces

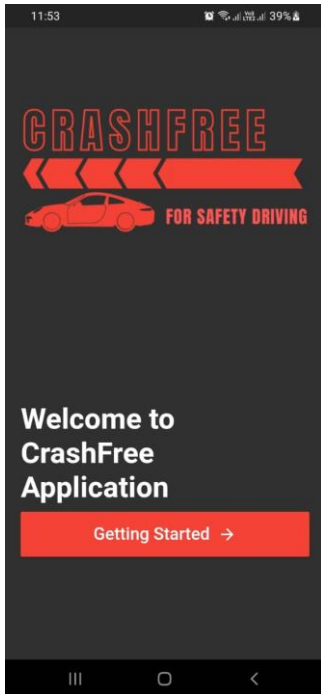


Figure A1. Starting Interface

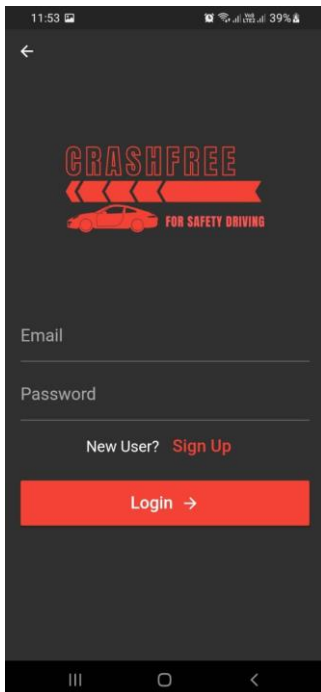


Figure A2. Login Interface

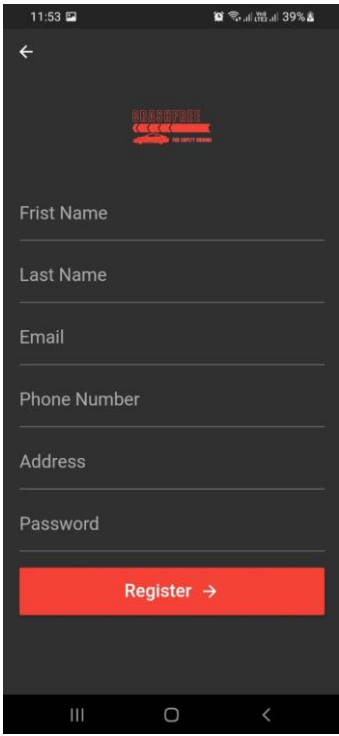


Figure A3. Register Interface

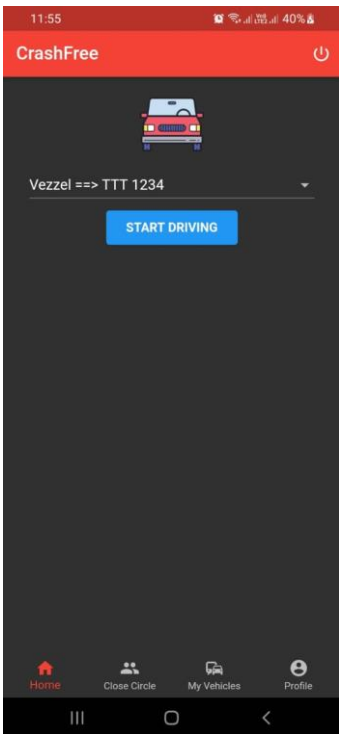


Figure A4. Home Interface – Before Driving

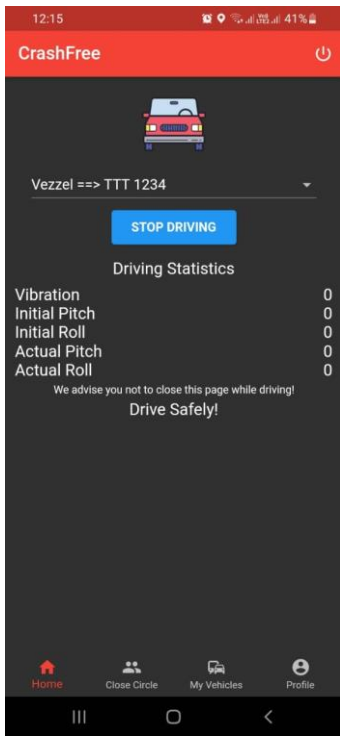


Figure A5. Home Interface - Started Driving

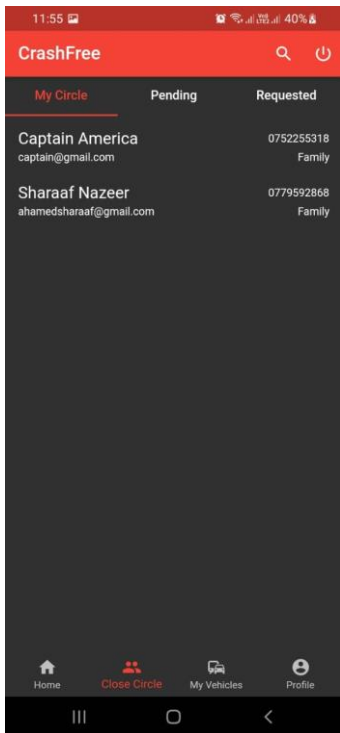


Figure A6. Close Circles Interface

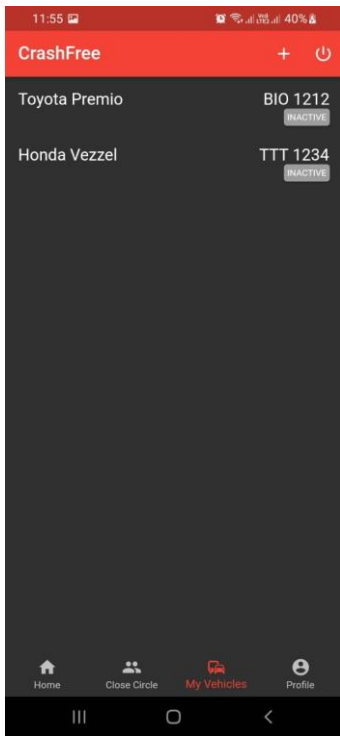


Figure A7. Added Vehicles Interface

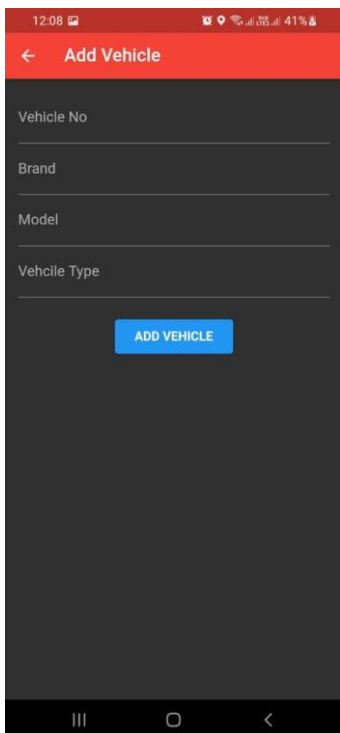


Figure A8. Adding Vehicles Interface



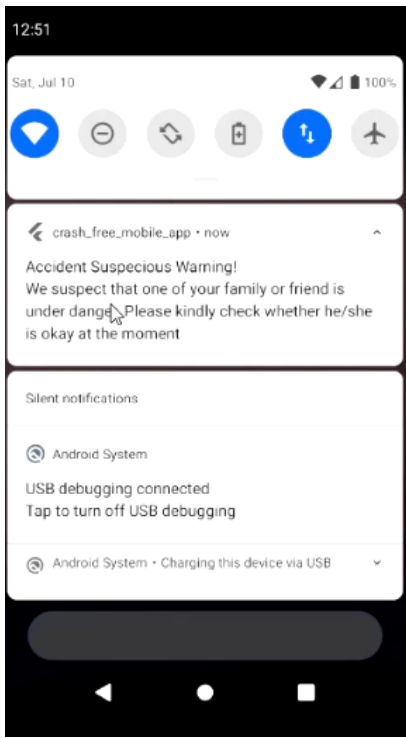


Figure A9. Notification Receiving Interface - Close Circle Perspective

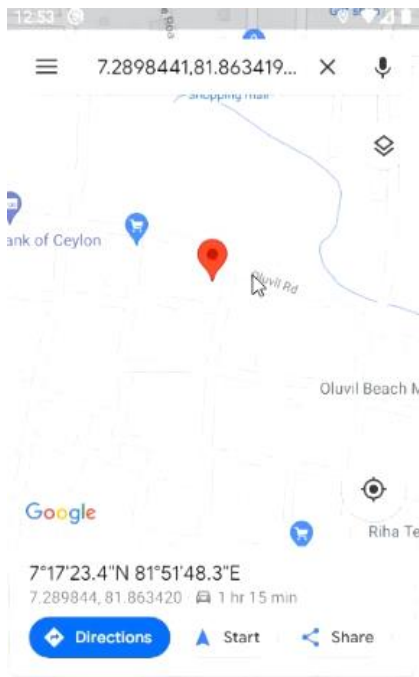


Figure A10. Locate Vehicle on Map Interface

## Appendix B: Image Extraction Code Snippet

```
22 def runDetector(image):
23     gray = image
24     rects = detector(gray, 0)
25     rect = None;
26     shape = None;
27     try:
28         rect = rects[0]
29         if rect is not None:
30             shape = predictor(gray,rect)
31     except IndexError:
32         pass
33     return shape, image, rect
34
35 def extractFace(faceRects, gray):
36     x = faceRects.left()
37     y = faceRects.top()
38     w = faceRects.right() - x
39     h = faceRects.bottom() - y
40
41     r = max(w, h) / 2
42     centerx = x + w / 2
43     centery = y + h / 2
44     nx = int(centerx - r)
45     ny = int(centery - r)
46     nr = int(r * 2)
47
48     face = gray[ny:ny + nr, nx:nx + nr]
49     if face.shape[0] > 0 and face.shape[1] > 0:
50         faceImage = cv2.resize(face, (image_size, image_size))
51     return faceImage
```

Figure B1. Extraction Code Segment 1

```

53 def extractEye(shape, gray):
54     x1Eye=shape.part(36).x
55     x2Eye=shape.part(39).x
56     y1Eye=shape.part(42).y
57     y2Eye=shape.part(45).y
58     eye=gray[y2Eye-18:y1Eye+18,x1Eye-30:x2Eye+30]
59     eyeImage = cv2.resize(eye, (image_size, image_size))
60     return eyeImage
61
62
63 def extractMouth(shape, gray):
64     xmouthpoints = [shape.part(x).x for x in range(48,67)]
65     ymouthpoints = [shape.part(x).y for x in range(48,67)]
66     maxx = max(xmouthpoints)
67     minx = min(xmouthpoints)
68     maxy = max(ymouthpoints)
69     miny = min(ymouthpoints)
70
71     mouth = gray[miny-8:maxy+10,minx-8:maxx+10]
72     mouthImage = cv2.resize(mouth, (image_size, image_size))
73     return mouthImage
74

```

Figure B2. Extraction Code Segment 2

## Appendix C: Drowsiness Prediction Results

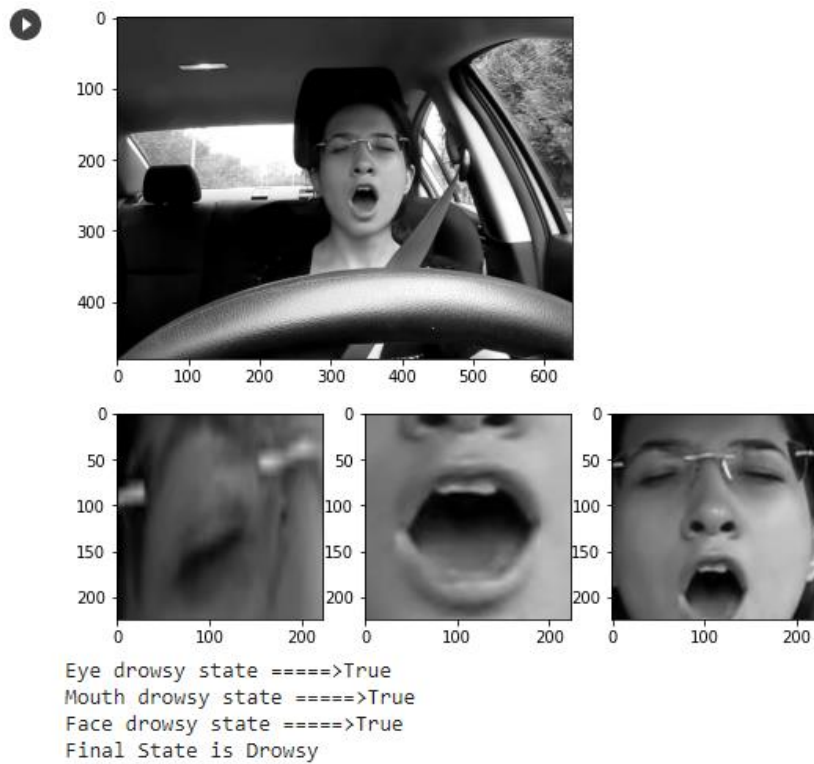


Figure C1. Drowsiness Prediction Results of Drowsy Face with Eyeglasses

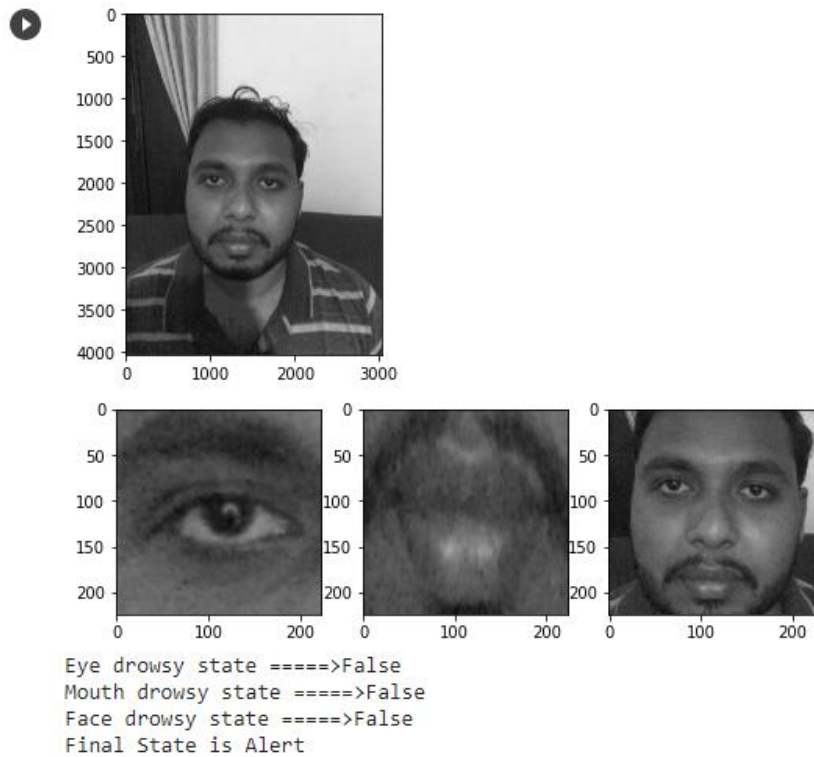


Figure C2. Drowsiness Prediction Results of Alert Face

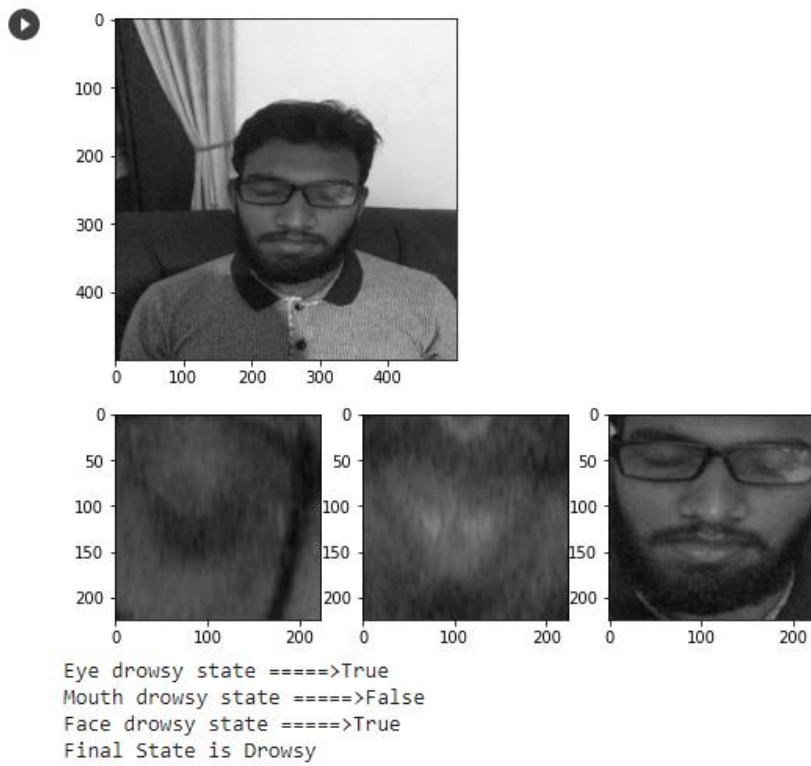


Figure C3. Drowsiness Prediction Results of Drowsy Face with Eyeglasses

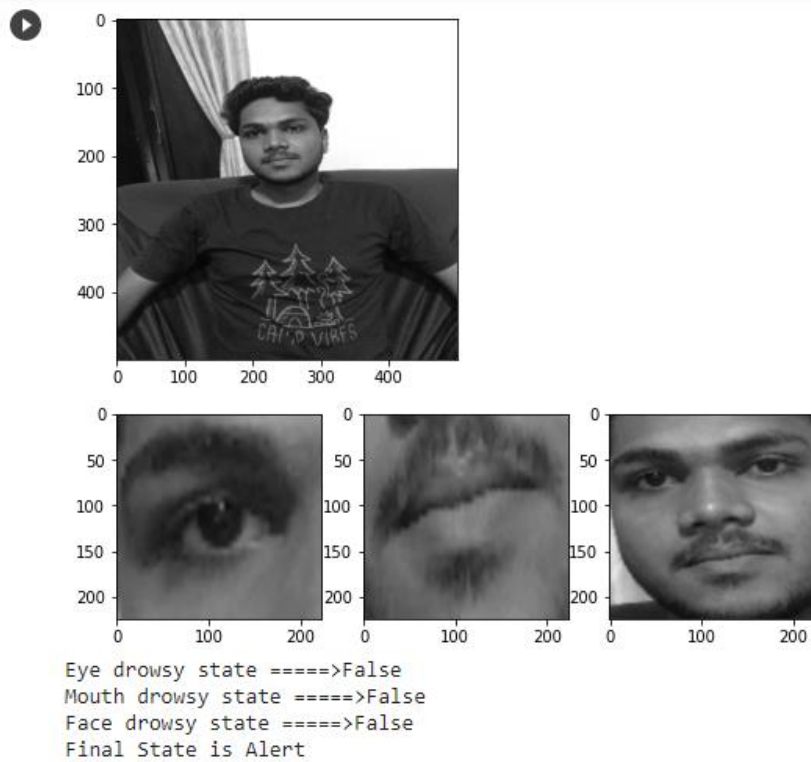


Figure C4. Drowsiness Prediction Results of Alert Face

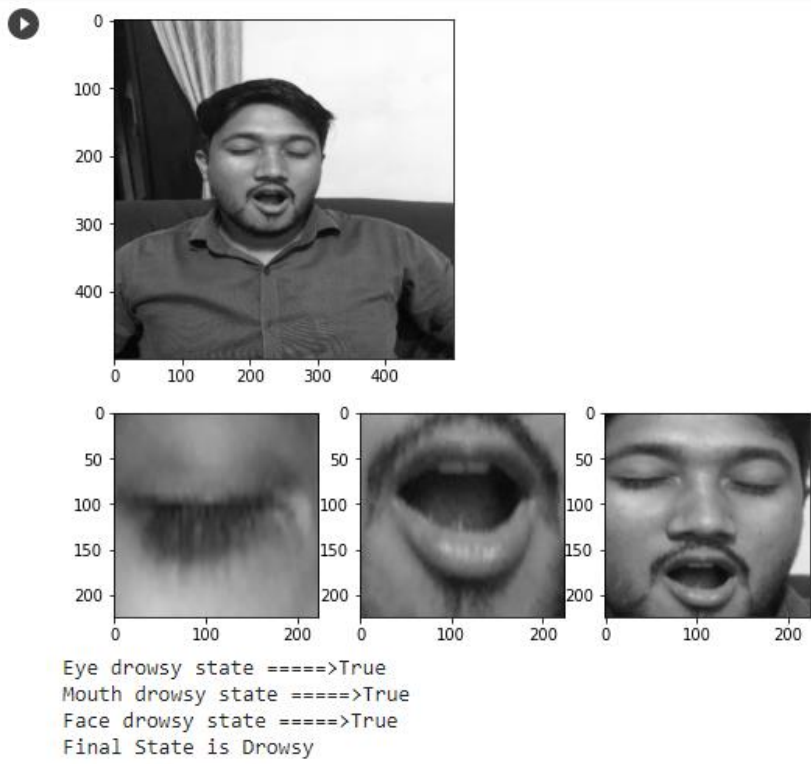


Figure C5. Drowsiness Prediction Results of Drowsy Face

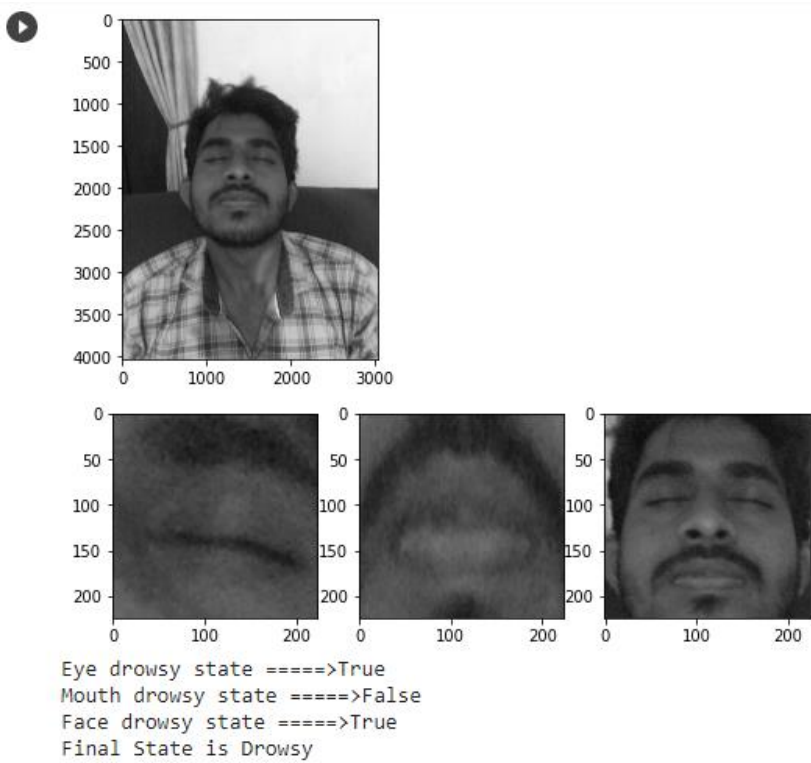
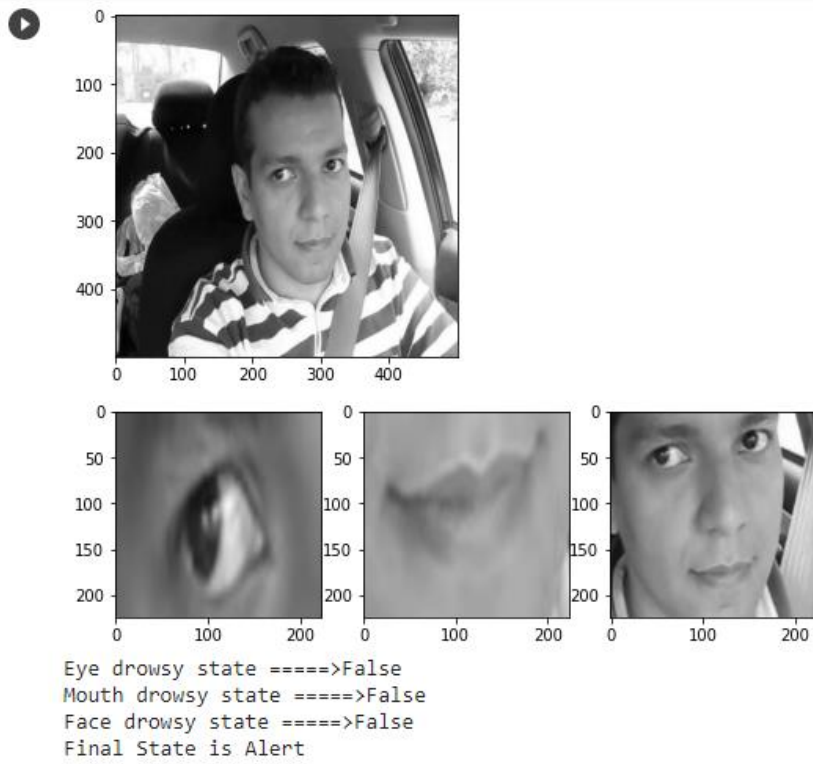
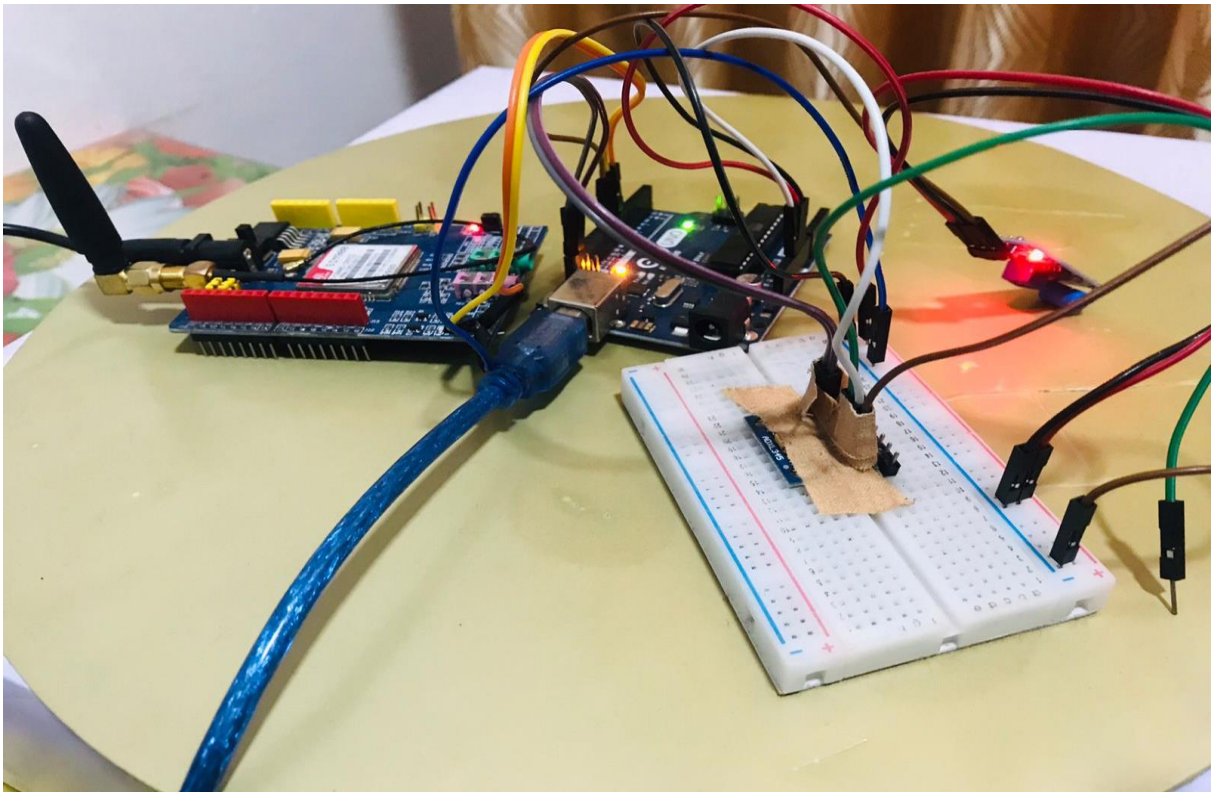


Figure C6. Drowsiness Prediction Results of Drowsy Face



*Figure C7. Drowsiness Prediction Results of Alert Face*

## Appendix D: Arduino Platform



*Figure D1. Arduino Platform View*



## Appendix E: Send Notification Code Snippet

```
1  var admin = require("firebase-admin");
2
3  class Message {
4    static async send(registrationTokens, payload) {
5
6      const options = {
7        priority: 'high',
8        contentAvailable: true,
9        timeToLive: 60*60*24
10     }
11
12     await admin.messaging()
13       .sendToDevice(registrationTokens, payload, options)
14       .then((response) => {
15         console.log('Sucessfully send message');
16         console.log(response);
17       })
18       .catch(error => {
19         console.log('Error while send message');
20         console.log(error);
21         Promise.reject(error);
22       });
23   }
24 }
25
26 module.exports = Message;
```

Figure E1. Code Segment of Send Notification

# Appendix F: Participant Information Sheet & Consent Form

## INFORMATION SHEET FOR RESEARCH PARTICIPANTS

### NAME OF THE EXPERIMENT: RESEARCH ON DRIVER DROWSINESS PREDICTION AND ACCIDENT DETECTION - DATASET COLLECTION

We would like to invite you to take part in an experiment. Here, you will find relevant information about the study which will help you decide whether to take part or not. You may choose whether to participate. Before getting involved in our experiment you can ask any questions you have, and you should be able to obtain satisfactory answers from them. In addition, you may withdraw without giving any reason and without consequences.

Please understand that your participation is voluntary, and you have the right to withdraw your consent or discontinue participation at any time in any experiment without penalty. Your images might be published in the this. Therefore, you have the full freedom to exit from the experiment any time. If you are dissatisfied at any time with any aspect of the study, you may contact anonymously, through the provided email or phone number immediately.

#### 1. What is the purpose of the research?

This research project aims to investigate about driver drowsiness and provide solution to prevent them from accidents.

#### 2. Why have I been invited?

You have been chosen because as you are a person who drives vehicles frequently.

#### 3. Do I have to take part?

We let you know that you have the total freedom to decide whether to take part. If you do decide to take part, you will be able to keep a copy of this information sheet and you should indicate your agreement to the consent form. You can still go out at any time.

#### 4. What will the study involve?

This study includes a session where your picture will be taken based on several conditions.

#### 5. Are there any risks in taking part in this study?

Participating in the research is not anticipated to cause you any disadvantages or discomfort. The potential physical and/or psychological harm or distress will be the same as any experienced in everyday life.

#### 6. Are there any benefits from taking part in this study?

Whilst there are no immediate benefits for those people participating in the project, it is hoped that this work will have a beneficial impact on detecting drowsy drivers and prevent them from accidents.

#### 7. Who has reviewed this study?

This project has been ethically approved University of Colombo School of Computing's Research ethics committee.

#### 8. Who is organizing and funding the research?

The project is organized under master's degree program research and no other parties will fund on this.

#### 9. What will happen to the results of the research?

Results of the research will be published. Your images might be available in the evaluation section of the report. If you wish to be given a copy of any reports resulting from the research, feel free to contact us.

#### 10. What if something goes wrong?

If you have any complaints about the project in the first instance you can contact any member of the research team. If you feel your complaint has not been handled to your satisfaction you can contact the University of Colombo's Registrar and Other members to take your complaint further or even, you can take legal actions.

*Figure F1. Participant Information Sheet*

**PARTICIPANT CONSENT FORM**

NAME OF THE STUDY: RESEARCH ON DRIVER DROWSINESS PREDICTION AND ACCIDENT DETECTION - DATASET COLLECTION

RESEARCHER: N.A. SHARAAF

SUPERVISOR: DR. HIRAN EKANAYAKE, SENIOR LECTURER, UCSC

1. I confirm that I have read and understand the information sheet dated 20/03/2021 concerning participation in experimental studies conducted in the study/research have had the opportunity to ask questions and have had satisfactory answers to any questions.

Yes No

2. I understand that my participation is voluntary and that I am free to withdraw at any time, without giving any reason, and without any adverse consequences.

Yes No

3. I understand that the information I provide in the experiments may be looked at by responsible individuals running the experiments. I give permission for these individuals to have access to the information provided for the research project (thesis).

Yes No

4. I understand that this project has been reviewed by, and received ethics clearance through, the University of Colombo Research Ethics Committee, and understand how to raise a concern and make a complaint.

Yes No

5. I agree to participate in this study.

_____	_____	_____
Name of the Participant	Date	Signature
_____	_____	_____
Name of the Researcher	Date	Signature

*Figure F2. Consent Form*

## Appendix G: Arduino Code Snippet

```
#include <Wire.h> // Wire library - used for I2C communication
#include <SoftwareSerial.h>

#include <Adafruit_Sensor.h>
#include <Adafruit_ADXL345_U.h>

SoftwareSerial SIM900(7, 8);

int vib_pin=10;
int led_pin=13;

Adafruit_ADXL345_Unified accel = Adafruit_ADXL345_Unified();
int ADXL345 = 0x53; // The ADXL345 sensor I2C address
float X_out, Y_out, Z_out; // Outputs
float roll,pitch,rollF,pitchF,rollInitial,pitchInitial=0;
int count, vibrationValue = 0;
String userId = "CGWdp8Bt7bnRX50nLJJ9";
String vehicleId = "1T4GFUUad4dArRTtkVgH";

void setup() {

  Serial.println("Starting the application!");
  Serial.begin(9600); // Initiate serial communication for printing the results on the Serial
  monitor

  Serial.println("Starting the accelerometer!");
  Wire.begin(); // Initiate the Wire library
  // Set ADXL345 in measuring mode
  Wire.beginTransmission(ADXL345); // Start communicating with the device
  Wire.write(0x2D); // Access/ talk to POWER_CTL Register - 0x2D
  // Enable measurement
  Wire.write(8); // Bit D3 High for measuring enable (8dec -> 0000 1000 binary)
  Wire.endTransmission();
  delay(10);
  //Off-set Calibration
  //X-axis
  Wire.beginTransmission(ADXL345);
  Wire.write(0x1E);
  Wire.write(3); // 1
  Wire.endTransmission();
  delay(10);
  //Y-axis
  Wire.beginTransmission(ADXL345);
  Wire.write(0x1F);
  Wire.write(1); // -2
  Wire.endTransmission();
  delay(10);
  //Z-axis
```

```

Wire.beginTransmission(ADXL345);
Wire.write(0x20);
Wire.write(-50); // -9
Wire.endTransmission();
delay(10);

Serial.println("Starting the GSM/GPRS!");
SIM900.begin(9600);
}
void loop() {

if(!accel.begin())
{
Serial.println("No Valid Accelerometer found");
//while(1);
} else {
Serial.println("Valid Accelerometer found");
}

// === Read accelerometer data === //
Wire.beginTransmission(ADXL345);
Wire.write(0x32); // Start with register 0x32 (ACCEL_XOUT_H)
Wire.endTransmission(false);
Wire.requestFrom(ADXL345, 6, true); // Read 6 registers total, each axis value is stored in
2 registers
X_out = ( Wire.read() | Wire.read() << 8); // X-axis value
X_out = X_out / 256; //For a range of +-2g, we need to divide the raw values by 256,
according to the datasheet
Y_out = ( Wire.read() | Wire.read() << 8); // Y-axis value
Y_out = Y_out / 256;
Z_out = ( Wire.read() | Wire.read() << 8); // Z-axis value
Z_out = Z_out / 256;

Serial.print("X Value => ");
Serial.println(X_out);

Serial.print("Y Value => ");
Serial.println(Y_out);

Serial.print("Z Value => ");
Serial.println(Z_out);

// Calculate Roll and Pitch (rotation around X-axis, rotation around Y-axis)
roll = atan(Y_out / sqrt(pow(X_out, 2) + pow(Z_out, 2))) * 180 / PI;
Serial.println(roll);
pitch = atan(-1 * X_out / sqrt(pow(Y_out, 2) + pow(Z_out, 2))) * 180 / PI;
// Low-pass filter
rollF = 0.94 * rollF + 0.06 * roll;
pitchF = 0.94 * pitchF + 0.06 * pitch;

```

```

Serial.print("Roll Value => ");
Serial.println(rollF);

Serial.print("Pitch Value => ");
Serial.println(pitchF);

Serial.print("Vibration === > ");
Serial.println(isVibrationAvailable());
vibrationValue = isVibrationAvailable();

delay(250);

if(count == 50) {
    pitchInitial = pitchF;
    rollInitial = rollF;
}

if(count > 50) {
    // send network request
    Serial.println(pitchInitial);
    Serial.println(rollInitial);
    run();
}

count = count + 1;

//run();
}

void run() {

    SIM900.listen();
    // SIM900.println("AT+CREG?");
    // delay(500);
    // toSerial();

    // SIM900.println("AT+CGATT?");
    // delay(500);
    // toSerial();

    // bearer settings
    SIM900.println("AT+SAPBR=3,1,\"CONTYPE\",\"GPRS\");
    delay(500);
    toSerial();

    // bearer settings
    SIM900.println("AT+SAPBR=3,1,\"APN\",\"AirtelLive\");
    delay(500);
    toSerial();
}

```

```

SIM900.println("AT+SAPBR=1,1");
delay(500);
toSerial();

SIM900.println("AT+SAPBR=2,1");
delay(500);
toSerial();

SIM900.println("AT+HTTTPINIT");
delay(500);
toSerial();

SIM900.println("AT+HTTTPSSL=1");
delay(500);
toSerial();

SIM900.println("AT+HTTTPARA=\"CID\",1");
delay(1000);
toSerial();

// set http param value
SIM900.println("AT+HTTTPARA=\"URL\", \"crash-free-
backend.herokuapp.com/api/track?userId="+userId+"&vehicleId="+vehicleId+"&vibration
Value="+vibrationValue+"&rollValueInitial="+rollInitial+"&pitchValueInitial="+pitchInit
ial+"&rollValue="+rollF+"&pitchValue="+pitchF+"");

delay(2500);
toSerial();

// set http action type 0 = GET, 1 = POST, 2 = HEAD
SIM900.println("AT+HTTTPACTION=0");
delay(2500);
toSerial();

// read server response
SIM900.println("AT+HTTTPREAD");
delay(2000);
toSerial();

SIM900.println("");
SIM900.println("AT+HTTTPTERM");
toSerial();
delay(500);
}

void toSerial(){
  while(SIM900.available()!=0){
    Serial.write(SIM900.read());
  }
}

```

```
bool isVibrationAvailable() {  
  
    int val;  
    val=digitalRead(vib_pin);  
    if(val==1)  
    {  
        digitalWrite(led_pin,HIGH);  
    }  
    else{  
        digitalWrite(led_pin,LOW);  
    }  
    return val;  
}
```



## REFERENCES

- 2018 Traffic Safety Culture Index [WWW Document], 2019. . AAA Foundation. URL <https://aaafoundation.org/2018-traffic-safety-culture-index/> (accessed 5.11.21).
- Aloul, F., Zualkernan, I., Abu-Salma, R., Al-Ali, H., Al-Merri, M., 2015. iBump: Smartphone application to detect car accidents. *Computers & Electrical Engineering* 43, 66–75. <https://doi.org/10.1016/j.compeleceng.2015.03.003>
- Arunasalam, M., Yaakob, N., Amir, A., Elshaikh, M., Azahar, N.F., 2020. Real-Time Drowsiness Detection System for Driver Monitoring. *IOP Conf. Ser.: Mater. Sci. Eng.* 767, 012066. <https://doi.org/10.1088/1757-899X/767/1/012066>
- Ashton, K., 2016. Beginning the Internet of Things [WWW Document]. Medium. URL [https://medium.com/@kevin\\_ashton/beginning-the-internet-of-things-6d5ab6178801](https://medium.com/@kevin_ashton/beginning-the-internet-of-things-6d5ab6178801) (accessed 5.15.21).
- Awais, M., Badruddin, N., Drieberg, M., 2017. A Hybrid Approach to Detect Driver Drowsiness Utilizing Physiological Signals to Improve System Performance and Wearability. *Sensors (Basel, Switzerland)* 17, 1991-. <https://doi.org/10.3390/s17091991>
- Bhatti, F., Shah, M.A., Maple, C., Islam, S.U., 2019. A Novel Internet of Things-Enabled Accident Detection and Reporting System for Smart City Environments. *Sensors* 19, 2071. <https://doi.org/10.3390/s19092071>
- Bolme, D.S., Beveridge, J.R., Draper, B.A., Lui, Y.M., 2010. Visual object tracking using adaptive correlation filters, in: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Presented at the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 2544–2550. <https://doi.org/10.1109/CVPR.2010.5539960>
- Chaturvedi, N., Srivastava, P., 2014. Automatic Vehicle Accident Detection and Messaging System Using GSM and GPS Modem. *IJAREEIE* 3, 10723–10727. <https://doi.org/10.15662/ijareeie.2014.0307062>
- Cortes, C., Vapnik, V., 1995. Support-vector networks. *Mach Learn* 20, 273–297. <https://doi.org/10.1007/BF00994018>
- Dalal, N., Triggs, B., 2005. Histograms of Oriented Gradients for Human Detection, in: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). Presented at the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), IEEE, San Diego, CA, USA, pp. 886–893. <https://doi.org/10.1109/CVPR.2005.177>
- Deng, W., Wu, R., 2019. Real-Time Driver-Drowsiness Detection System Using Facial Features. *IEEE access* 7, 118727–118738. <https://doi.org/10.1109/access.2019.2936663>
- D'Orazio, T., Leo, M., Guaragnella, C., Distanto, A., 2007. A visual approach for driver inattention detection. *Pattern Recognition* 40, 2341–2355. <https://doi.org/10.1016/j.patcog.2007.01.018>
- Ghoddosian, R., Galib, M., Athitsos, V., 2019. A Realistic Dataset and Baseline Temporal Model for Early Drowsiness Detection. *arXiv:1904.07312 [cs]*.
- Hashemi, M., Mirrashid, A., Shirazi, A.B., 2020. CNN-based Driver Drowsiness Detection. *SN COMPUT. SCI.* 1, 289. <https://doi.org/10.1007/s42979-020-00306-9>
- Henriques, J.F., Caseiro, R., Martins, P., Batista, J., 2015. High-Speed Tracking with Kernelized Correlation Filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37, 583–596. <https://doi.org/10.1109/TPAMI.2014.2345390>

- Henriques, J.F., Caseiro, R., Martins, P., Batista, J., 2012. Exploiting the Circulant Structure of Tracking-by-Detection with Kernels, in: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (Eds.), *Computer Vision – ECCV 2012, Lecture Notes in Computer Science*. Springer, Berlin, Heidelberg, pp. 702–715. [https://doi.org/10.1007/978-3-642-33765-9\\_50](https://doi.org/10.1007/978-3-642-33765-9_50)
- Jabbar, R., Al-Khalifa, K., Kharbeche, M., Alhajyaseen, W., Jafari, M., Jiang, S., 2018. Real-time Driver Drowsiness Detection for Android Application Using Deep Neural Networks Techniques. *Procedia computer science* 130, 400–407. <https://doi.org/10.1016/j.procs.2018.04.060>
- Jumana Waleed, Thekra Abbas, Taha Mohammed Hasan, 2020. Implementation of driver's drowsiness assistance model based on eye movements detection. *Eastern-European journal of enterprise technologies* 5, 6–13. <https://doi.org/10.15587/1729-4061.2020.211755>
- Kamel, M., Guan, L., 1989. Histogram Equalization Utilizing Spatial Correlation For Image Enhancement, in: *Visual Communications and Image Processing IV*. Presented at the Visual Communications and Image Processing IV, International Society for Optics and Photonics, pp. 712–723. <https://doi.org/10.1117/12.970082>
- Katyal, Y., Alur, S., Dwivedi, S., 2014. Safe driving by detecting lane discipline and driver drowsiness, in: *2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies*. Presented at the 2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies, pp. 1008–1012. <https://doi.org/10.1109/ICACCCT.2014.7019248>
- Kazemi, V., Sullivan, J., 2014. One millisecond face alignment with an ensemble of regression trees. *undefined*.
- Ki, Y.-K., Lee, D.-Y., 2007. A Traffic Accident Recording and Reporting Model at Intersections. *IEEE Transactions on Intelligent Transportation Systems* 8, 188–194. <https://doi.org/10.1109/TITS.2006.890070>
- King, D.E., n.d. *Dlib-ml: A Machine Learning Toolkit 4*.
- Kong, W., Zhou, L., Wang, Y., Zhang, J., Liu, J., Gao, S., 2015. A System of Driving Fatigue Detection Based on Machine Vision and Its Application on Smart Device [WWW Document]. *Journal of Sensors*. <https://doi.org/10.1155/2015/548602>
- Kowalski, M., Naruniec, J., Trzcinski, T., 2017. Deep Alignment Network: A convolutional neural network for robust face alignment. *arXiv:1706.01789 [cs]*.
- Lahn, J., Peter, H., Braun, P., 2015. Car crash detection on smartphones, in: *Proceedings of the 2nd International Workshop on Sensor-Based Activity Recognition and Interaction, IWOAR '15*. Association for Computing Machinery, New York, NY, USA, pp. 1–4. <https://doi.org/10.1145/2790044.2790049>
- Li, G., Chung, W.-Y., 2013. Detection of Driver Drowsiness Using Wavelet Analysis of Heart Rate Variability and a Support Vector Machine Classifier. *Sensors (Basel, Switzerland)* 13, 16494–16511. <https://doi.org/10.3390/s131216494>
- Li, G., Lee, B.-L., Chung, W.-Y., 2015. Smartwatch-Based Wearable EEG System for Driver Drowsiness Detection. *IEEE sensors journal* 15, 7169–7180. <https://doi.org/10.1109/jsen.2015.2473679>
- Li, Y., Zhu, J., 2015. A Scale Adaptive Kernel Correlation Filter Tracker with Feature Integration, in: Agapito, L., Bronstein, M.M., Rother, C. (Eds.), *Computer Vision - ECCV 2014 Workshops, Lecture Notes in Computer Science*. Springer International Publishing, Cham, pp. 254–265. [https://doi.org/10.1007/978-3-319-16181-5\\_18](https://doi.org/10.1007/978-3-319-16181-5_18)

- Li, Z., Li, S.E., Li, R., Cheng, B., Shi, J., 2017. Online Detection of Driver Fatigue Using Steering Wheel Angles for Real Driving Conditions. *Sensors (Basel)* 17. <https://doi.org/10.3390/s17030495>
- Liang, G.J., 2015. Automatic Traffic Accident Detection Based on the Internet of Things and Support Vector Machine. *International Journal of Smart Home* 9, 97–106. <https://doi.org/10.14257/ijsh.2015.9.4.10>
- Lipkin, L., 1970. *Picture Processing by Computer*. Azriel Rosenfeld. Academic Press, New York, 1969. x + 198 pp., illus. \$11.50. *Computer Science and Applied Mathematics*. Science 169, 166–167. <https://doi.org/10.1126/science.169.3941.166>
- Lucas, B.D., Kanade, T., 1981. An iterative image registration technique with an application to stereo vision, in: *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2, IJCAI'81*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 674–679.
- Ma, W., Huang, L., Liu, C., 2008. Advanced Local Binary Pattern Descriptors for Crowd Estimation. Presented at the Pacific-Asia Workshop on Computational Intelligence and Industrial Application, pp. 958–962. <https://doi.org/10.1109/PACIIA.2008.258>
- McConnell, R.K., 1986. Method of and apparatus for pattern recognition. US4567610A.
- Mohamed, F., Sathees Kumar Nataraj, Ahmed, S.F., Sazali Yaacob, 2018. An Approach In Determining Fatigueness And Drowsiness Detection Using EEG. <https://doi.org/10.13140/RG.2.2.12630.50243>
- Omidyeganeh, M., Javadtalab, A., Shirmohammadi, S., n.d. Intelligent Driver Drowsiness Detection through Fusion of Yawning and Eye Closure 7.
- Pinto, A., Bhasi, M., Bhalekar, D., Hegde, P., Koolagudi, S.G., 2019. A Deep Learning Approach to Detect Drowsy Drivers in Real Time, in: *2019 IEEE 16th India Council International Conference (INDICON)*. Presented at the 2019 IEEE 16th India Council International Conference (INDICON), pp. 1–4. <https://doi.org/10.1109/INDICON47234.2019.9030305>
- Prabha, C., Sunitha, R., Anitha, R., 2014. Automatic Vehicle Accident Detection and Messaging System Using GSM and GPS Modem. *IJAREEIE* 3, 10723–10727. <https://doi.org/10.15662/ijareeie.2014.0307062>
- Rahmad, C., Asmara, R.A., Putra, D.R.H., Dharma, I., Darmono, H., Muhiqqin, I., 2020. Comparison of Viola-Jones Haar Cascade Classifier and Histogram of Oriented Gradients (HOG) for face detection. *IOP Conf. Ser.: Mater. Sci. Eng.* 732, 012038. <https://doi.org/10.1088/1757-899X/732/1/012038>
- Ramzan, M., Khan, H.U., Awan, S.M., Ismail, A., Ilyas, M., Mahmood, A., 2019. A Survey on State-of-the-Art Drowsiness Detection Techniques. *IEEE Access* 7, 61904–61919. <https://doi.org/10.1109/ACCESS.2019.2914373>
- Routh, J., das, A., Kundu, P., Thakur, M., 2019. Automatic Vehicle Accident Detection and Messaging System Using GPS and GSM Module. *International Journal of Engineering Trends and Technology* 67, 69–72. <https://doi.org/10.14445/22315381/IJETT-V67I8P211>
- Saini, V., Saini, R., 2014. Driver Drowsiness Detection System and Techniques: A Review 5, 5.
- Sarada Devi, M., Bajaj, P., 2008. Driver Fatigue Detection Using Mouth and Yawning Analysis 8.
- Sun, Y., Wang, X., Tang, X., 2013. Deep Convolutional Network Cascade for Facial Point Detection, in: *2013 IEEE Conference on Computer Vision and Pattern Recognition*. Presented at the 2013 IEEE Conference on Computer Vision and Pattern Recognition

- (CVPR), IEEE, Portland, OR, USA, pp. 3476–3483.  
<https://doi.org/10.1109/CVPR.2013.446>
- Tefft, B.C., 2014. Prevalence of Motor Vehicle Crashes Involving Drowsy Drivers, United States, 2009 – 2013.
- Tushara, D., P.A., H.V., 2016. Wireless vehicle alert and collision prevention system design using Atmel microcontroller. pp. 2784–2787.  
<https://doi.org/10.1109/ICEEOT.2016.7755203>
- Viola, P., Jones, M., 2001. Rapid Object Detection using a Boosted Cascade of Simple Features, IEEE Conf Comput Vis Pattern Recognit.  
<https://doi.org/10.1109/CVPR.2001.990517>
- Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P., 2004. Image Quality Assessment: From Error Visibility to Structural Similarity. IEEE Trans. on Image Process. 13, 600–612. <https://doi.org/10.1109/TIP.2003.819861>
- World Health Organization | Global status report on road safety 2013 [WWW Document], n.d. . WHO. URL  
[http://www.who.int/violence\\_injury\\_prevention/road\\_safety\\_status/2013/en/](http://www.who.int/violence_injury_prevention/road_safety_status/2013/en/) (accessed 5.11.21).
- Wu, Y., Hassner, T., Kim, K., Medioni, G., Natarajan, P., 2016. Facial Landmark Detection with Tweaked Convolutional Neural Networks. arXiv:1511.04031 [cs].
- Zhao, Z., Zhou, N., Zhang, L., Yan, H., Xu, Y., Zhang, Z., 2020. Driver Fatigue Detection Based on Convolutional Neural Networks Using EM-CNN. Computational Intelligence and Neuroscience 2020, e7251280. <https://doi.org/10.1155/2020/7251280>
- Zhenhai, G., DinhDat, L., Hongyu, H., Ziwen, Y., Xinyu, W., 2017. Driver Drowsiness Detection Based on Time Series Analysis of Steering Wheel Angular Velocity, in: 2017 9th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA). Presented at the 2017 9th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA), pp. 99–101.  
<https://doi.org/10.1109/ICMTMA.2017.0031>
- Zhou, E., Fan, H., Cao, Z., Jiang, Y., Yin, Q., 2013. Extensive Facial Landmark Localization with Coarse-to-Fine Convolutional Network Cascade, in: 2013 IEEE International Conference on Computer Vision Workshops. Presented at the 2013 IEEE International Conference on Computer Vision Workshops (ICCVW), IEEE, Sydney, Australia, pp. 386–391. <https://doi.org/10.1109/ICCVW.2013.58>

