# DRIVER DROWSINESS DETECTION USING DEEP LEARNING

A PROJECT REPORT

submitted by

**FEBIN JACS GEORGE**

**KTE18MCA027**

**to**

the APJ Abdul Kalam Technological University

in partial fulfillment of the requirements for the award of the degree

of

Master of Computer Applications


**Department of Computer Applications**

**RAJIV GANDHI INSTITUTE OF TECHNOLOGY**

**(Government Engineering College)**

**KOTTAYAM - 686 501, KERALA**

July 2021

# DECLARATION

I undersigned hereby declare that the project report ”DRIVER DROWSINESS DETECTION USING DEEP LEARNING”, submitted for partial fulfillment of the requirements for the award of degree of Master of Computer Applications of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by me under supervision of prof.Shalu Murali, Assistant Proffessor. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

PAMPADY

July 9, 2021 FEBIN JACS GEORGE

Tecgenix

***Date 10-06-21***

This is to certify that Mr Febin Jacs Geroge(KTE18MCA027) , MCA 6th semester student of Rajiv Gandhi Institute of Technology ,Kottayam has successfully completed Internship on the project " Driver Drowsiness Detection using Deep Learning" in python from April 15th to June 10th at TECGENIX Pvt Ltd During the period of his internship programme with us he was found Sincere, punctual ,hardworking ,efficient and Inquisitive .

*We wish a bright future*

*With Regards*

*HR Manager*

+919048245318

www.tecgenix.com

tecgenix@gmail.com

TL-G102, Nazland building,
Near kairali TV office
Kunnukuzhi lane palayam
Trivandrum 33 kerala

# ACKNOWLEDGEMENT

I wish to thank the multitude of people who have helped me during the course of the MCA.

First of all, I thank God almighty for His grace and blessings for without his unforeseen guidance this would have remained in dreams.

I express my sincere gratitude to **Dr.Jalaja M.J ,Principal**, Rajiv Gandhi Institute of Technology, Kottayam, for providing the ambiance for carrying out the work of this project.

I deeply indebted to **Prof.John C John**, Head of the Department, Computer Applications for providing permission and availing all required facilities for undertaking the project in a systematic way.

I feel deeply honoured in expressing my sincere thanks to **Prof.Shalu Murali,** Assistant Professor of department of Computer Application for the constructive suggestions and inspirations that helped me during the preparation of this project.

I take this opportunity to thank all the technical staffs of Department of Computer Applications for their help.

I also express my gratitude to **Ms.Subilal**, AI developer, TECGENIX Trivandrum for providing me with adequate facilities, Support and guidance ways and means to complete this internship.

Gratitude may be extended to my parents and friends who supported us for the project.

# ABSTRACT

Fatigue and microsleep are the reasons behind many severe road accidents. These can be avoided if the symptoms of fatigue are detected on time. This project describes a real-time system for monitoring driver vigilance.Driver drowsiness detection system in the past have proven to work in controlled environments but have not been implemented on a wide scale as of yet. This project is based on Eye Aspect Ratio (EAR), Yawning detection and detect drowsiness by comparing its instantaneous value with a previously configured value. Here i propose a generalised approach using Convolution Neural Networks (CNN), Harcascade classifier (HCC) and Hidden Markov Model (HMM) in this project. Our project tracks the driver's eyes and feeds it into a pre-trained that predicts the state of the eye. Once the prediction is obtained, we would be able to detect if the driver is drowsy or not. The main components of our system include a webcam, for real time image acquisition, a processor for running algorithms to process the acquired image and an alarm to warn the driver when the symptoms are detected in order to avoid potential accidents.

# CONTENTS

# LIST OF FIGURES

# CHAPTER 1

# INTRODUCTION

## 1.1 Need for the Project

Vehicle accidents are most common if the driving is inadequate. These happen on most factors if the driver is drowsy or if he is alcoholic. Driver drowsiness is recognized as an important factor in the vehicle accidents. It was demonstrated that driving performance deteriorates with increased drowsiness with resulting crashes constituting more than 20 percentage of all vehicle accidents. But the life lost once cannot be rewinded. Driver drowsiness detection is a car safety technology which helps prevent accidents caused by the driver getting drowsy.Various studies have suggested that around 20 percentage of all road accidents are fatigue-related, up to 50 percentage on certain roads.

## 1.2 Outline of the Report

Details of proposed system is provided in chapter 2. Hardware and software specifications for both development and implementations are detailed. Module description and data flow diagrams are described in chapter 3. Chapter 4 includes model evaluation, datsets and some accuracy graph.Test cases and some screenshots of the test cases are summarized in chapter 5.Implimenation method and implimentation plan are in chapter 6.Chapter 7 contain the conclusion and future scope.

## 1.3 Motivation

Our current statistics reveal that just in 2018 in India alone, 148,707 people died due to car related accidents. Of these, at least 21 percent were caused due to fatigue causing drivers to make mistakes. This can be a relatively smaller number still, as among the multiple causes that can lead to an accident, the involvement of fatigue as a cause is generally grossly underestimated. Fatigue combined with bad infrastructure in developing countries like India is a recipe for disaster. Fatigue, in general, is very difficult to measure or observe unlike alcohol and drugs, which have clear key indicators and tests that are available easily. Probably, the best solutions to this problem are awareness about fatigue-related accidents and promoting drivers to admit fatigue when needed. The former is hard and much more expensive to achieve, and the latter is not possible without the former as driving for long hours is very lucrative. When there is an increased need for a job, the wages associated with it increases leading to more and more people adopting it. Such is the case for driving transport vehicles at night. Money motivates drivers to make unwise decisions like driving all night even with fatigue. This is mainly because the drivers are not themselves aware of the huge risk associated with driving when fatigued. Some countries have imposed restrictions on the number of hours a driver can drive at a stretch, but it is still not enough to solve this problem as its implementation is very difficult and costly.

## 1.4 Scope of the project

This project focus on the utilization of outer factors such as EAR(Eye Aspect Ratio), MAR(Mouth Aspect Ratio), Pupil circulatory, MOE and Yawning detection for fatigue measurement. Driver drowsiness pose a major threat to highway safety, and the problem is particularly severe for commercial motor vehicle operators. Twenty-four

hour operations, high annual mileage, exposure to challenging environmental conditions, and demanding work schedules all contribute to this serious safety issue. Monitoring the driver's state of drowsiness and vigilance and providing feedback on their condition.so that they can take appropriate action is one crucial step in a series of preventive measures necessary to address this problem.

# CHAPTER 2

# REQUIREMENT ANALYSIS AND SPECIFICATION

## 2.1 System study

System analysis was done by having frequent meetings with our clients and thereby acquiring a detailed specification of their Requirement. I had done several meetings with one of our client.His main requirement was to make a System inorder to save drivers and co passengers lives.I prepared some questionaires regarding our doubts and cleared it with them.

The system study will done related to some parameters. A detailed review on these measures will provide insight on the present systems, issues associated with them and the enhancements that need to be done to make a robust system.The parameters like the behaviour of the driver, including yawning, eye closure, eye blinking, skin wrinkleness etc is monitored through a camera and the driver is alerted if any of these drowsiness symptoms are detected. Here the camera and alarm is plugged in to the car battery.So camera is automatically on when the car is in the start position.

### 2.1.1 Existing system

By using a non intrusive machine vision based concepts, drowsiness of the driver detected system is developed. Many existing systems require a camera which is installed in front of driver. It points straight towards the face of the driver and monitors the drivers eyes in order to identify the drowsiness. For large vehicle such as heavy

trucks and buses this arrangement is not pertinent. Bus has a large front glass window to have a broad view for safe driving. If we place a camera on the window of front glass, the camera blocks the frontal view of driver so it is not practical. If the camera is placed on the frame which is just about the window, then the camera is unable to detain the anterior view of the face of the driver correctly.

The present system is all about manually Waking up the driver or by using the safety measures provided by the car manufacturer.No Alert System softwares Present till now.Hence existing system is not applicable for large vehicles. In order to conquer the problem of existing system, new detection system is developed in this project work

### 2.1.2 Proposed system

**DRIVER DROWSINESS DETECTION SYSTEM USING DEEP LEARNIING** is a software. It involves controlling vehicle accident saving driver's life by alerting the driver on real time.This Software based on Eye Aspect Ratio (EAR), yawning detection and detect drowsiness by comparing its instantaneous value with a previously configured value.I propose a generalised approach using Convolution Neural Networks (CNN), Harcascade classifier (HCC) and Hidden Markov Model (HMM) in this project my project tracks the driver's eyes and feeds it into a pre-trained that predicts the state of the eye. in this project driver face is captured using camera in a video mode.The video is converted into image frame format with the help of VGG16 architecture. Whenever driver's eye blinking period is more than 1 second, we consider this condition under drowsiness i.e driver is drowse.We also detect yawning and fatigue face in real time.If the software detect any drowsy nature in driver, Immediate action will be taken Buzzer will immediately blown up alerts wake's up the driver. This drowsiness condition is reported to the remote person through alarm in real time. Coding is done in Python programming language and Machine learning.

## 2.2 System specification

### 2.2.1 Specification for development

#### 2.2.1.1 Hardware Specification

The selection of hardware configuration is a very important task related to the software dvelopment.Random acesss memory may affect adversely on the speed and correspondingly on the efficiency of the entire system.The processor should be powerful to handle all the operation.The hard disk should have sufficient capacity to store the database and the application.The network should be efficient to handle the communication fast.

- Processor :i3, 1.8 GB
- RAM : 4 GB (or greater)
- Hard Drive : 30 GB
- Video :800*600, 256 colors
- Camera : 180 resolution
- Display Adaptor: SMC Ethernet card Elite 16

#### 2.2.1.2 Software Specification

- OS : Windows7 and Above
- Platform : Anaconda/colab
- Database Server : MySQL
- Front end : Python 3.7

## 2.3 Technology used

### 2.3.1 VGG16

VGG16 is a convolutional neural network model proposed by K. Simonyan and A. Zisserman from the University of Oxford in the paper "Very Deep Convolutional Networks for Large-Scale Image Recognition".The model contain 16 layers and achieves 92.7 percentage top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes. It was one of the famous model submitted to ILSVRC-2014. It makes the improvement over AlexNet by replacing large kernel-sized filters (11 and 5 in the first and second convolutional layer, respectively) with multiple 3×3 kernel-sized filters one after another. VGG16 was trained for weeks and was using NVIDIA Titan Black GPU's.The default input size for VGG16 model is 224 x 224 pixels with 3 channels for RGB image. It has convolution layers of 3x3 filter with a stride 1 and maxpool layer of 2x2 filter of stride 2. VGG16 function. In this project we use VGG16 for converting images into frames and filtering the image into more accuracy.

### 2.3.2 keras

Keras is an Open Source Neural Network library written in Python that runs on top of Theano or Tensorflow. It is designed to be modular, fast and easy to use. It was developed by François Chollet, a Google engineer. Keras doesn't handle low-level computation. Instead, it uses another library to do it, called the "Backend.Keras is high-level API wrapper for the low-level API, capable of running on top of TensorFlow, CNTK, or Theano. Keras High-Level API handles the way we make models, defining layers, or set up multiple input-output models. In this level, Keras also compiles our model with loss and optimizer functions, training process with fit function. Keras in Python doesn't handle Low-Level API such as making the computational graph,

making tensors or other variables because it has been handled by the ”backend” engine.


**Fig. 2.1. Keras**

### 2.3.3 Tensor flow

TensorFlow is a software library or framework, designed by the Google team to implement machine learning and deep learning concepts in the easiest manner. It combines the computational algebra of optimization techniques for easy calculation of many mathematical expressions.TensorFlow is well-documented and includes plenty of machine learning libraries. It offers a few important functionalities and methods for the same. TensorFlow is also called a “Google” product. It includes a variety of machine learning and deep learning algorithms. TensorFlow can train and run deep neural networks for handwritten digit classification, image recognition, word embedding and creation of various sequence models.


**Fig. 2.2. Tensor Flow**

### 2.3.4 colab

Colaboratory is a Google research project created to help disseminate machine learning education and research. It's a Jupyter notebook environment that requires no setup to use and runs entirely in the cloud. Colab notebooks allow we to combine executable code and rich text in a single document, along with images, HTML, LaTeX and more.Here we use colaboratory platform to store and developing the code.With Colab we can import an image dataset, train an image classifier on it, and evaluate the model, all in just a few lines of code. Colab notebooks execute code on Google's cloud servers, meaning you can leverage the power of Google hardware, including GPUs and TPUs, regardless of the power of your machine.


**Fig. 2.3. Colab**

### 2.3.5 Deep learning

Deep learning is an artificial intelligence (AI) function that imitates the workings of the human brain in processing data and creating patterns for use in decision making. Deep learning is a subset of machine learning in artificial intelligence that has networks capable of learning unsupervised from data that is unstructured or unlabeled. Also known as deep neural learning or deep neural network.Deep learning has evolved hand-in-hand with the digital era, which has brought about an explosion of data in all forms and from every region of the world. This data, known simply as big data, is drawn from sources like social media, internet search engines, e-commerce platforms,

and online cinemas, among others. This enormous amount of data is readily accessible and can be shared through fintech applications like cloud computing.

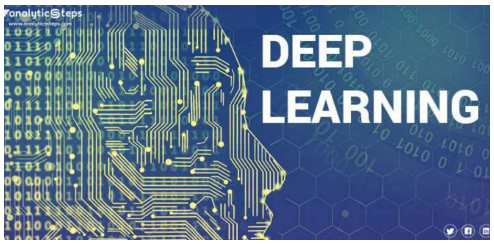



**Fig. 2.4. Deep learning**

### 2.3.6 numpy

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. This tutorial explains the basics of NumPy such as its architecture and environment. It also discusses the various array functions, types of indexing, etc. An introduction to Matplotlib is also provided.


**Fig. 2.5. Numpy**

### 2.3.7 Pandas

Pandas is an open-source, BSD-licensed Python library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics.


**Fig. 2.6. Pandas**

### 2.3.8 Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural ”pylab” interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged.SciPy makes use of Matplotlib.


**Fig. 2.7. Matplotlib**

### 2.3.9 OpenCV

Computer vision is an interdisciplinary scientific field that deals with how computers can gain high-level understanding from digital images or videos. From the perspective of engineering, it seeks to understand and automate tasks that the human visual system can do. Computer vision tasks include methods for acquiring, processing, analyzing and understanding digital images, and extraction of high-dimensional data from the real world in order to produce numerical or symbolic information, e.g. in the forms of decisions. Understanding in this context means the transformation of visual images (the input of the retina) into descriptions of the world that make sense to thought processes and can elicit appropriate action. This image understanding can be seen as the disentangling of symbolic information from image data using models constructed with the aid of geometry, physics, statistics, and learning theory.


**Fig. 2.8. OpenCV**

## 2.4 Software Tools

### 2.4.1 Python

Python 3.8 used in this project. Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using

significant whitespace. Python was conceived in the late 1980s by Guido van Rossum at Centrum Wiskunde Informatica (CWI) in the Netherlands as a successor to the ABC language (itself inspired by SETL), capable of exception handling and interfacing with the Amoeba operating system.Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspectoriented programming (including by meta programming and meta objects (magic methods)). Many other paradigms are supported via extensions, including design by contract and logic programming. Python 2.7 is the last major release in the 2.x series, as the Python maintainers have shifted the focus of their new feature development efforts to the Python 3.x series[4]. This means that while Python 2 continues to receive bug fixes, and to be updated to build correctly on new hardware and versions of supported operated systems, there will be no new full feature releases for the language or standard library.

# CHAPTER 3

# SYSTEM MODELLING

## 3.1 Introduction

System modeling is the inter disciplinary study of the model to conceptualize and construct in business and IT development. A common type of systems modeling is function modeling, with specific techniques such as the Data Flow Diagram. These models can be extended using functional decomposition, and can be linked to requirements models for further systems partition.One of the challenges of developing a real-time model of a system is that many systems have non-linear behavior.Let a computer learn to imitate a system and then analyze that model instead. Since we can artificially constrain the model that the ML algorithm will create, we can learn a lot from the results.

## 3.2 MODULE DESCRIPTION

### 3.2.1 Modules

- Image Capturing
  - This is the stage where image frames are taken from a fixed camera.The image frames are taken in such a manner that only the face of the driver is captured.
- Face Detection

  - The second stage typically aims to detect the face in the image frames. From the image frames, the face is detected first. Convolutional Neural Network (CNN) feeds the whole image to a network that has multiple filters and the face features are extracted from this network.

- Face Extraction

  - If face detection is applied, features are usually extracted using different methods such as landmark localization, Histogram of oriented gradients (HOG), and Local Binary Patterns (LBP). This step simplifies the image by extracting useful information and discarding irrelevant information. Eyes detected by pixel difference, or by using Sobel vertical edge operator.

- Feature Analysis

  - Extracted features can then be processed further, as is the case for PERCLOS (percentage of eyelid closure) or EAR (eye aspect ratio) for eye analysis or mouth-based methods for yawning detection. The features of the face extracted can then be processed further, as is the case for PERCLOS. These algorithms transform the RGB form images to the gradation image by the skin color segmentation and then the eyes are detected. Then it calculates the speed at which the eyes close.

- Classification

  - The classification stage consists of classifiers that are used for decision-making on the level of drowsiness in a driver. If the classifier detects traits of drowsiness based on the weighted parameters, then an alarm will be activated suggesting that a driver takes a break. In the drowsiness detection, HCCs learns from the categorized data into the classified form of data. A

number of measures are used in the detection of driver drowsiness and the level of driver drowsiness. A fully automatic system for the detection of the driver drowsiness is presented [33], HAAR feature algorithm is used for the detection of the Eyes and face detection, HCCs are trained in the states like close, open eyes and to trigger the alarm.

- DNN and Alarm formation

  - DNN is a hierarchical model with deep architecture that consists of a number of convolutional and sub-sampling layers followed by fully connected layers which eliminate the feature extractor in the traditional model of pattern recognition. Convolutional and pooling layers are rotated to process large input signals into small features in small resolution. A set of neurons in the convolutional layers detect different patterns on a patch of the input. Each unit of a layer receives inputs from a set of units located in a small neighbourhood in the previous layer. Neurons can extract elementary features of input with local receptive fields which is then combined by the higher layers. Each neuron contains a set of trainable weights and a bias and feature maps are calculated by applying an activation function to the weighted sum of the inputs plus the bias.

    With a pooling layer, the resolution of the feature maps is reduced Representation learning is used in the proposed algorithm for the detection of driver drowsiness, here the Viola Jones algorithm is used to detect the face. Firstly, the images are cropped in 48*48 size image and then fed up to the outmost layer of the network contained the 20 filters, the output is delivered to the SoftMax layer, but the system leads to failure because it does not consider the head pose. But another author achieved the more accurate result by

using the 3D deep neural network in which face is tackle by the combination of two more filters, the system works well even when the driver is changing the head.

## 3.3 Data Flow Diagram

A data flow diagram is graphical tool used to describe and analyze movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system. These are known as the logical data flow diagrams. The physical data flow diagrams show the actual implements and movement of data between people, departments and workstations.

A full description of a system actually consists of a set of data flow diagrams. Using two familiar notations Yourdon, Gane and Sarson notation develops the data flow diagrams. Each component in a DFD is labeled with a descriptive name. Process is further identified with a number that will be used for identification purpose. The development of DFD'S are done in several levels. Each process in lower level diagrams can be broken down into a more detailed DFD in the next level.

The lop-level diagram is often called context diagram. It consists a single process bit, which plays vital role in studying the current system. The process in the context level diagram is exploded into other process at the first level DFD. The idea behind the explosion of a process into more process is that understanding at one level of detail is exploded into greater detail at the next level. This is done until further explosion is necessary and an adequate amount of detail is described for analyst to understand the process.

## 3.4 System Design

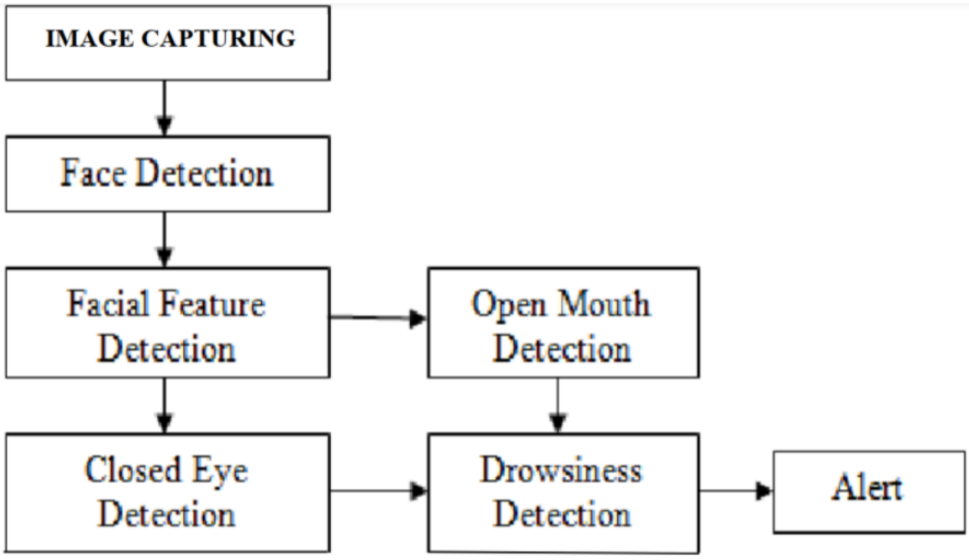



**Fig. 3.1. System Design**

### 3.4.1 Product Backlog

| ID | AS A | I WANT TO BE ABLE TO | SO THAT I CAN | PRIORITY | STATUS |
|---|---|---|---|---|---|
| 1 | Software Developer | As a Sw Developer,I develop the software and hardware like camera and alarm | .View the face of the driver<br>.alarm when he/she is in a drowsy stage | must | To be started |
| 2 | Software Developer | As a Software developer I can extract the image features using vgg16 | .View the images in a frame format<br>.extract the face features like mouth,face,fatigue | must | To be started |
| 3 | Software Analyst | As a sw analyst I can analyse the features of image | .view extracted image<br>.give values to the image | must | To be started |
| 4 | Software Analyst | As a sw analyst I can clarify the dataset image with new image | Compare the dataset values with image value | must | To be started |
| 5 | Software Analyst | As a sw analyst I can manage the alarm | Produce alarm when the value is not matching | must | To be started |

### 3.4.2 Sprint Backlog

| SL NO | DATE | TASKS | STATUS |
|---|---|---|---|
| 1 | 20-4-21<br>TO<br>30-4-21 | .Project topic Introduction<br>.existing system study<br>.proposed system study<br>.module description<br>.data set collection | completed |
| 2 | 01-5-21<br>TO<br>16-5-21 | .Discuss how to extract the features<br>.installation of packages<br>.image frame formation<br>.model building | completed |
| 3 | 17-5-21<br>TO<br>23-5-21 | .model training and learning<br>.extract features<br>.analysis features<br>.cross validation and propagation | completed |
| 4 | 24-5-21<br>TO<br>13-6-21 | .model testing and matching<br>.classification of template matching<br>.alarm formation<br>.implementation | completed |

**Fig. 3.2. sprint backlog details**

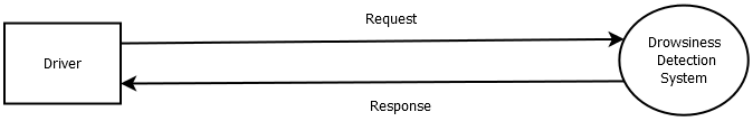
### 3.4.3 DFD Symbol

In DFD, there are four symbols:

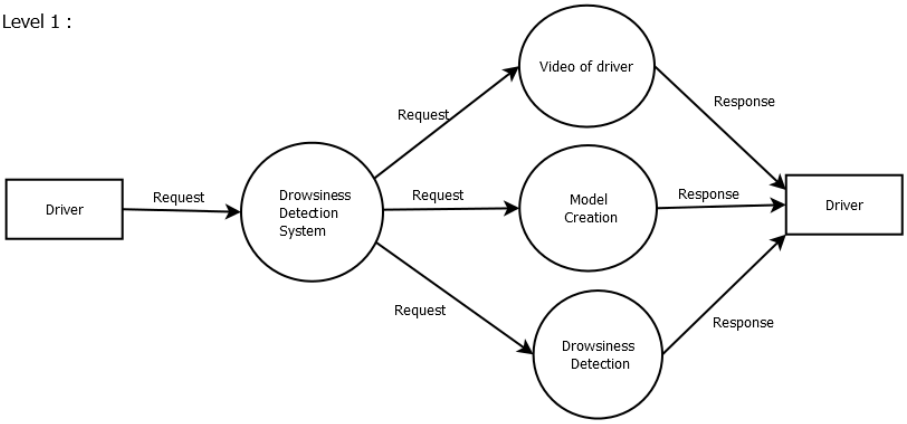
1. A square defines a source (originator) or destination of system data.
2. An arrow identifies data flow. It is the pipeline through which the information flows.
3. A circle or a bubble represents a process that transforms incoming data flow into outgoing data flows.
4. An open rectangle is a data store, data at rest or a temporary repository of data

DFD FOR DRIVER DROWSINES DETECTION USING DEEP LEARNING

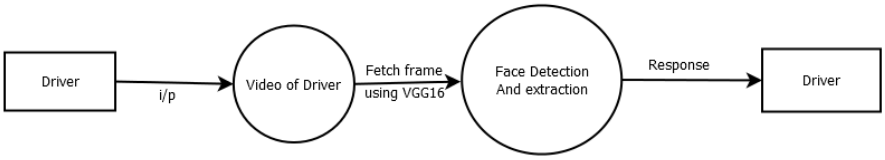
Level 0 :



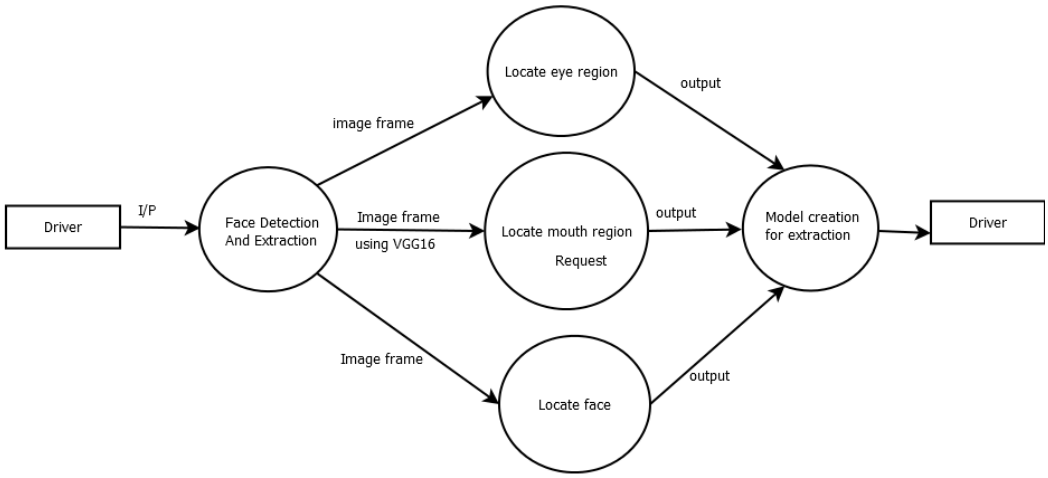

Level 1 :



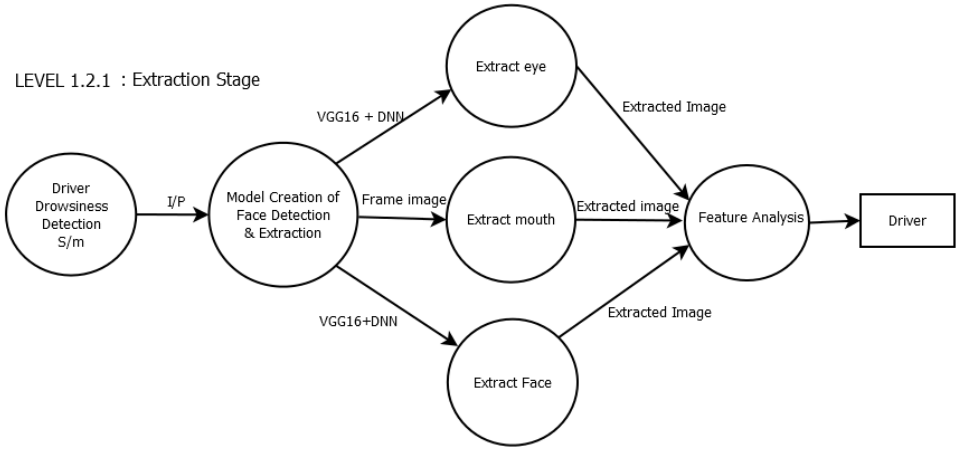

**Level 1.1 :**Image Frame Formation



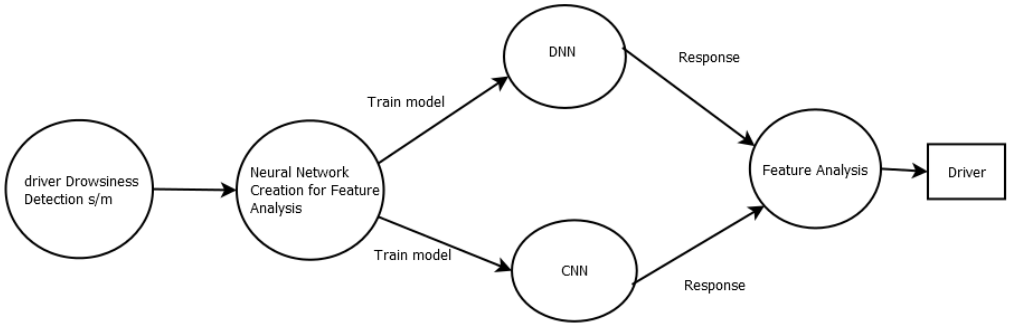
LEVEL 1.2 : Model Creation



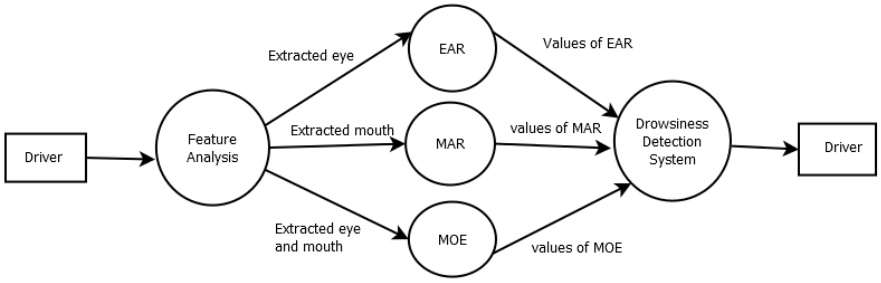

LEVEL 1.2.1 : Extraction Stage



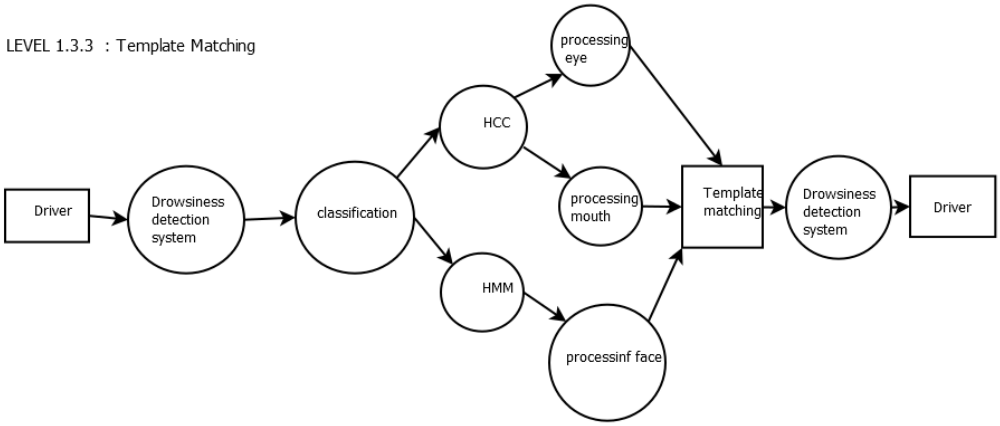

LEVEL 1.3 : Analysis Stage



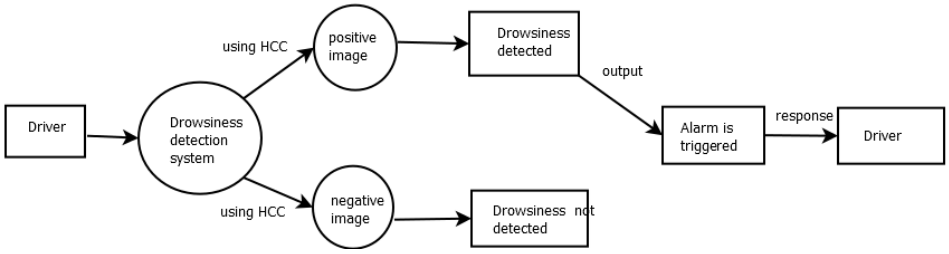
LEVEL 1.3.1 : Value calculation from Image




LEVEL 1.3.3 : Template Matching




Level 1.3.3 : Alarm Formation



## 3.5 Block Diagram

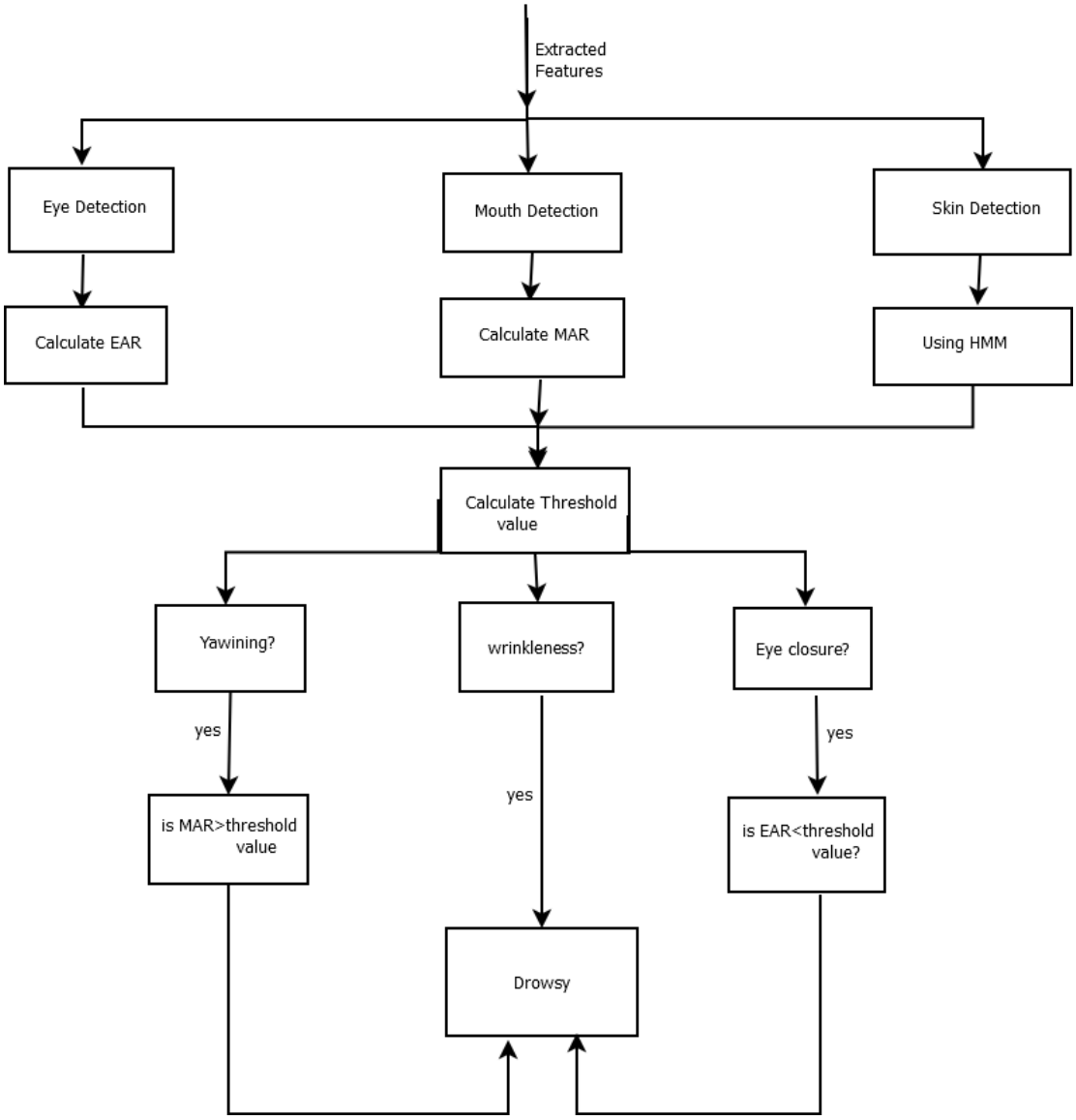



**Fig. 3.3. BlockDiagram**

# CHAPTER 4

# MODEL EVALUATON

## 4.1 Introduction

Machine learning continues to be an increasingly integral component of our lives, whether we're applying the techniques to research or business problems. Machine learning models ought to be able to give accurate predictions in order to create real value for a given organization. The amount that the weights are updated during training is referred to as the step size or the "learning rate". Specifically, the learning rate is a configurable hyper parameter used in the training of neural networks that has a small positive value, often in the range between 0.0 and 1.0. We trained the model with different learning rate and thus obtained the following results.

## 4.2 Datasets

### 4.2.1 Drowsiness in eye

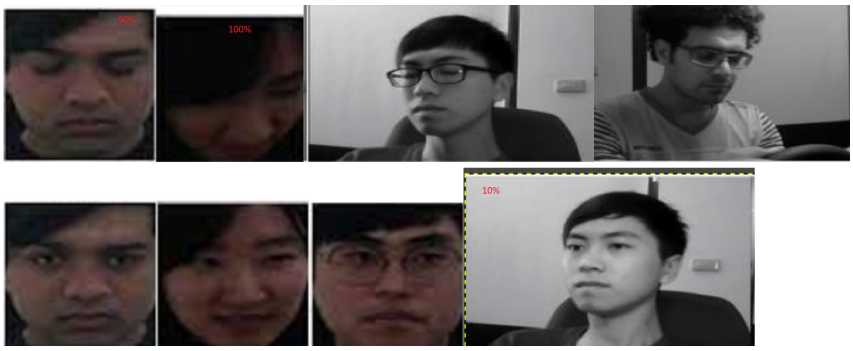



**Fig. 4.1. eye drowsiness stage**

### 4.2.2 Drowsiness in mouth


**Fig. 4.2. drowsiness in yawning stage**

### 4.2.3 Redness in eye

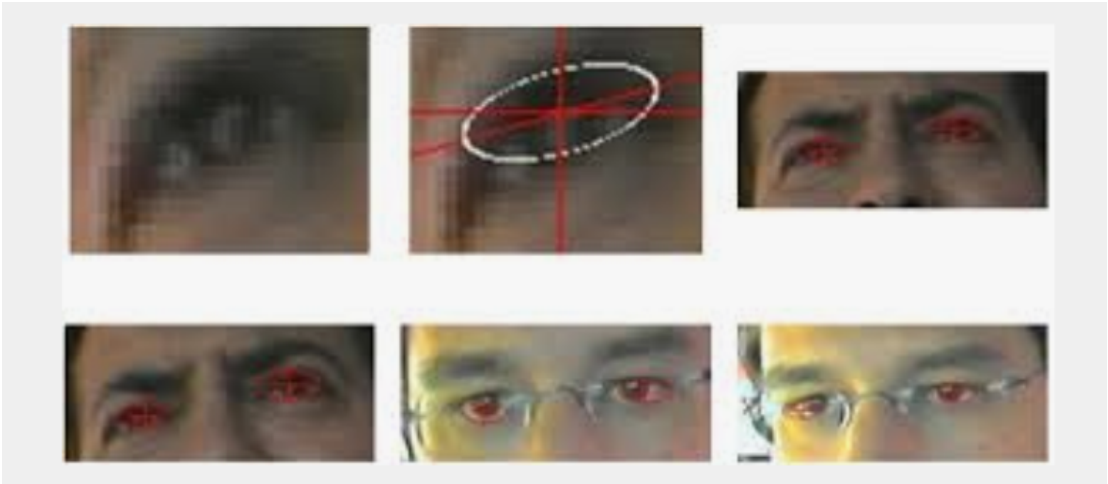


**Fig. 4.3. eye redness**

### 4.2.4 Model Accuracy

| Layer (type) | Output Shape | Param # |
|---|---|---|
| input_1 (InputLayer) | [(None, 224, 224, 3)] | 0 |
| block1_conv1 (Conv2D) | (None, 224, 224, 64) | 1792 |
| block1_conv2 (Conv2D) | (None, 224, 224, 64) | 36928 |
| block1_pool (MaxPooling2D) | (None, 112, 112, 64) | 0 |
| block2_conv1 (Conv2D) | (None, 112, 112, 128) | 73856 |
| block2_conv2 (Conv2D) | (None, 112, 112, 128) | 147584 |
| block2_pool (MaxPooling2D) | (None, 56, 56, 128) | 0 |
| block3_conv1 (Conv2D) | (None, 56, 56, 256) | 295168 |
| block3_conv2 (Conv2D) | (None, 56, 56, 256) | 590080 |
| block3_conv3 (Conv2D) | (None, 56, 56, 256) | 590080 |
| block3_pool (MaxPooling2D) | (None, 28, 28, 256) | 0 |
| block4_conv1 (Conv2D) | (None, 28, 28, 512) | 1180160 |
| block4_conv2 (Conv2D) | (None, 28, 28, 512) | 2359808 |

**Fig. 4.4. Accuracy table**

### 4.2.5 Output Graph

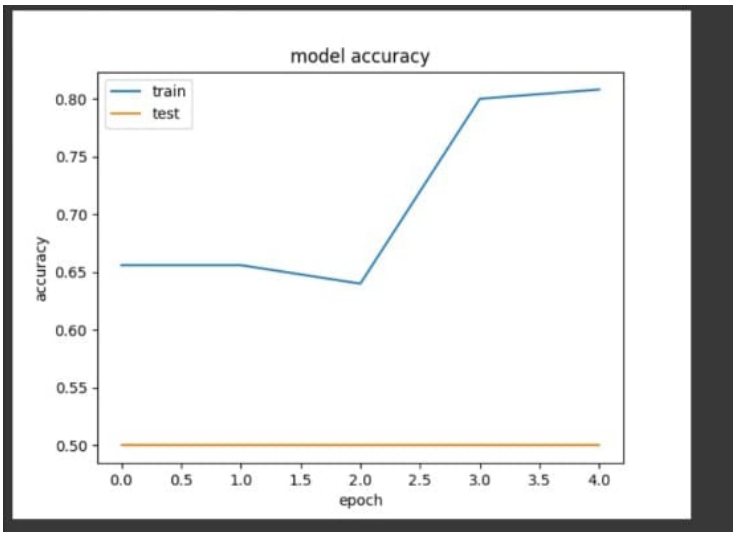



**Fig. 4.5. output graph**

### 4.2.6 Training loss and accuracy

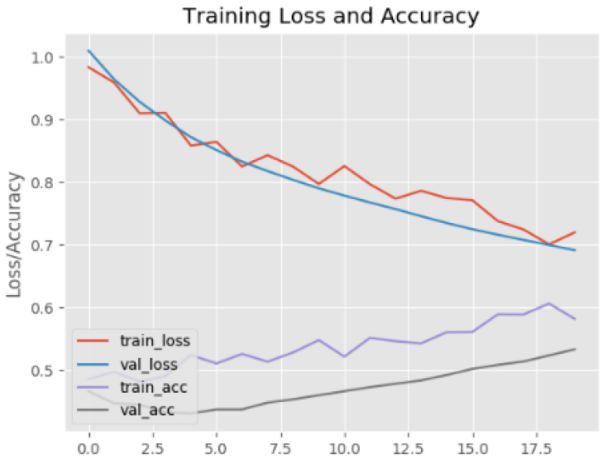



**Fig. 4.6. training loss accuracy graph**

# CHAPTER 5

# SYSTEM TESTING

## 5.1 Introduction

Testing is the process of examining the software to compare the actual behavior with that of the excepted behavior. The major goal of software testing is to demonstrate that faults are not present. In order to achieve this goal the tester executes the program with the intent of finding errors. Though testing cannot show absence of errors but by not showing their presence it is considered that these are not present. System testing is the first Stage of implementation, which is aimed at ensuring that the system works accurately and efficiently before live operations commences . Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct and the goal will be successfully achieved. A series of testing are performed for the proposed system before the proposed system is ready for user acceptance testing.

### 5.1.1 Unit testing

In this each module is tested individually before integrating it to the final system . Unit test focuses verification in the smallest unit of software design in each module. This is also known as module testing as here each module is tested to check whether it is producing the desired output and to see if any error occurs.

### 5.1.2 Integration testing

Integration testing is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing. The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items.

### 5.1.3 User acceptance testing

No system could be useful if it does not produce the required output in the specific format. Output testing is performed to ensure the correctness of the output and its format. The output generated or displayed by the system is tested asking the users about the format required by them .

### 5.1.4 Test Cases

Detecting Feature of normal eye

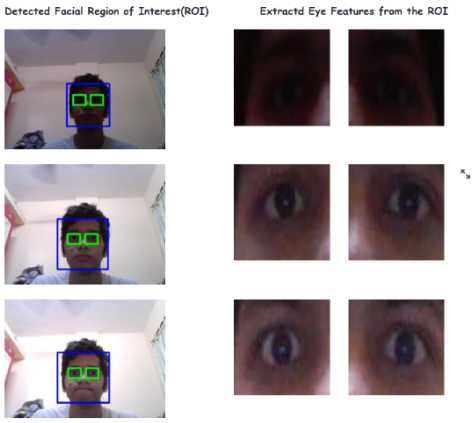



**Fig. 5.1. normal eye detecting**

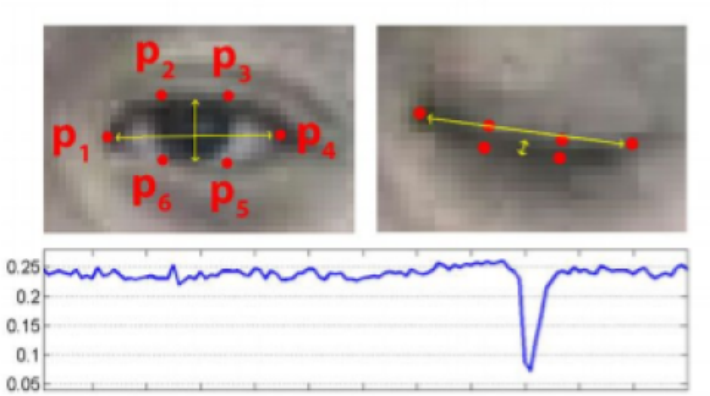



**Fig. 5.2. Eye distance calculating**

**Fig. 5.3. Facial contour**

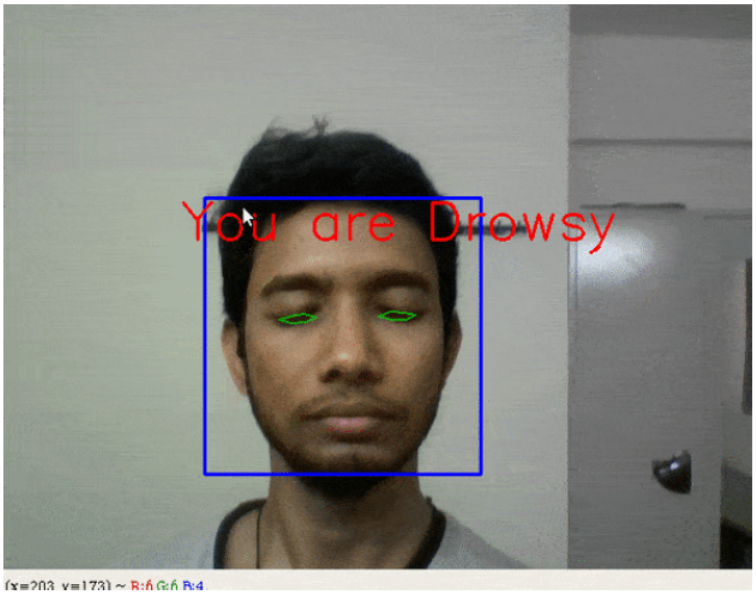



**Fig. 5.4. Drowsiness stage**

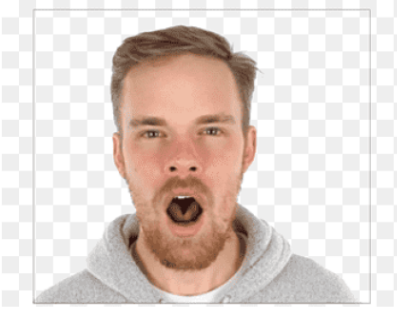


**Fig. 5.5. yawning stage**

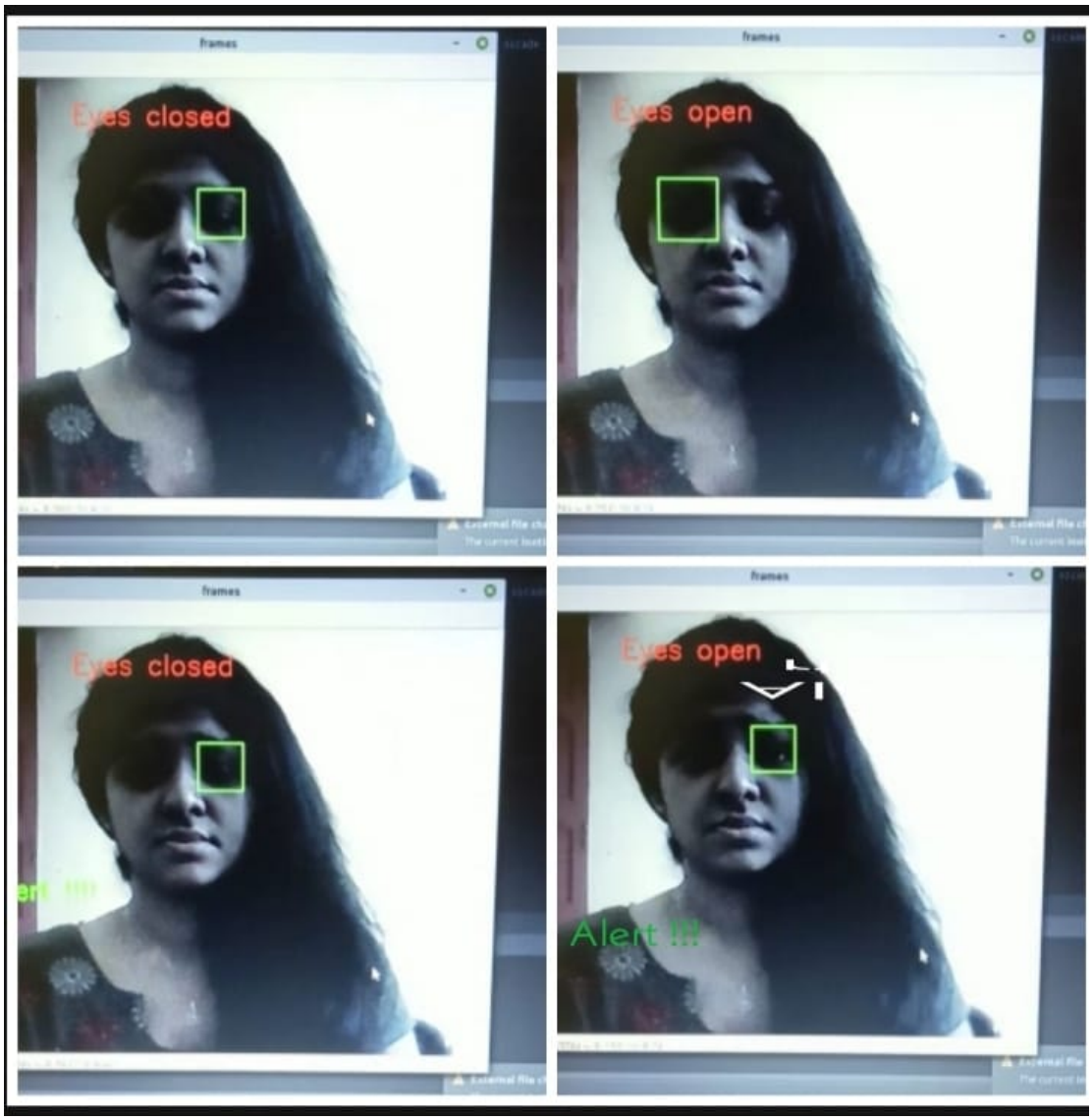



**Fig. 5.6. Eye closed**

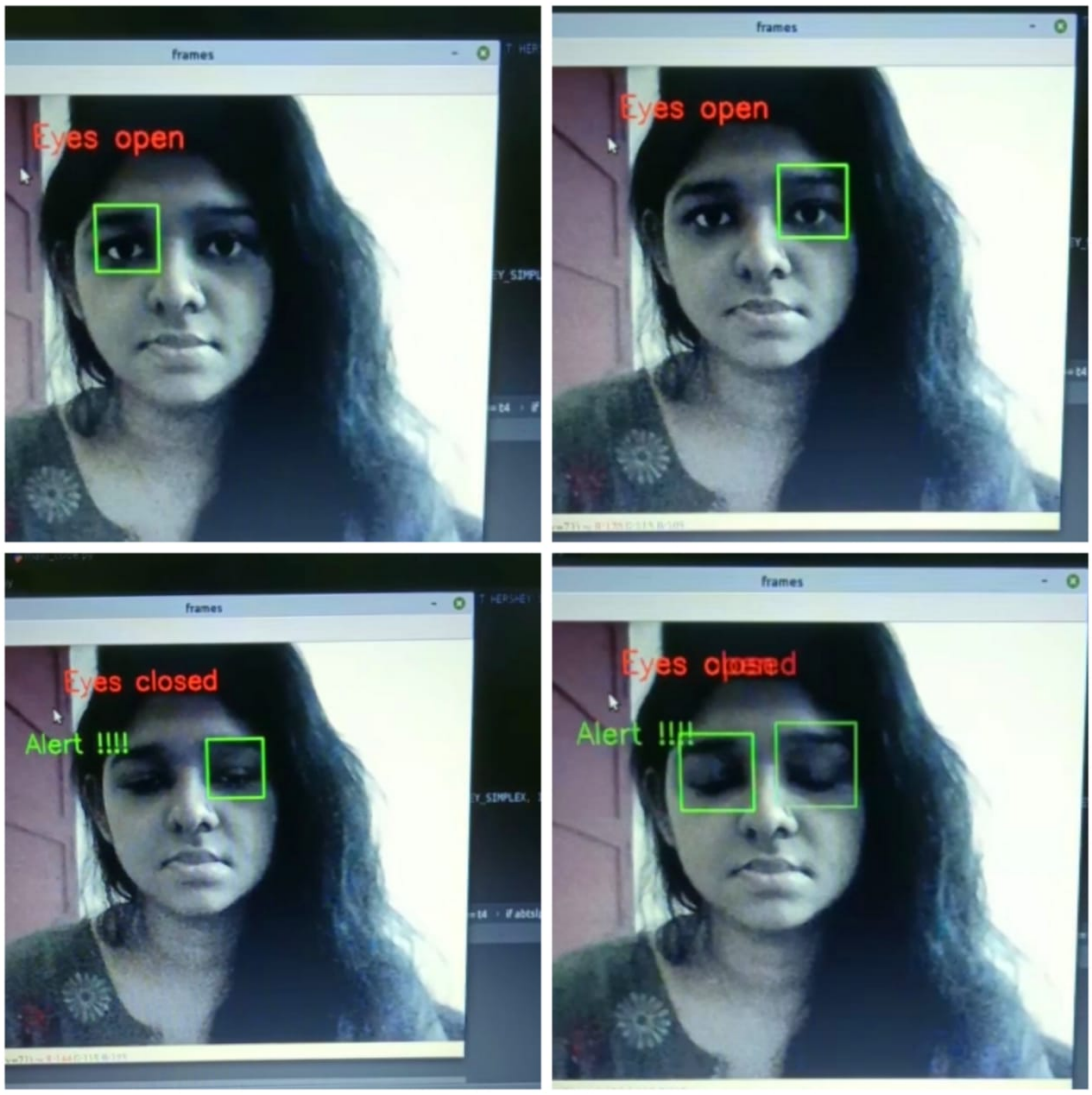



**Fig. 5.7. drowsy**

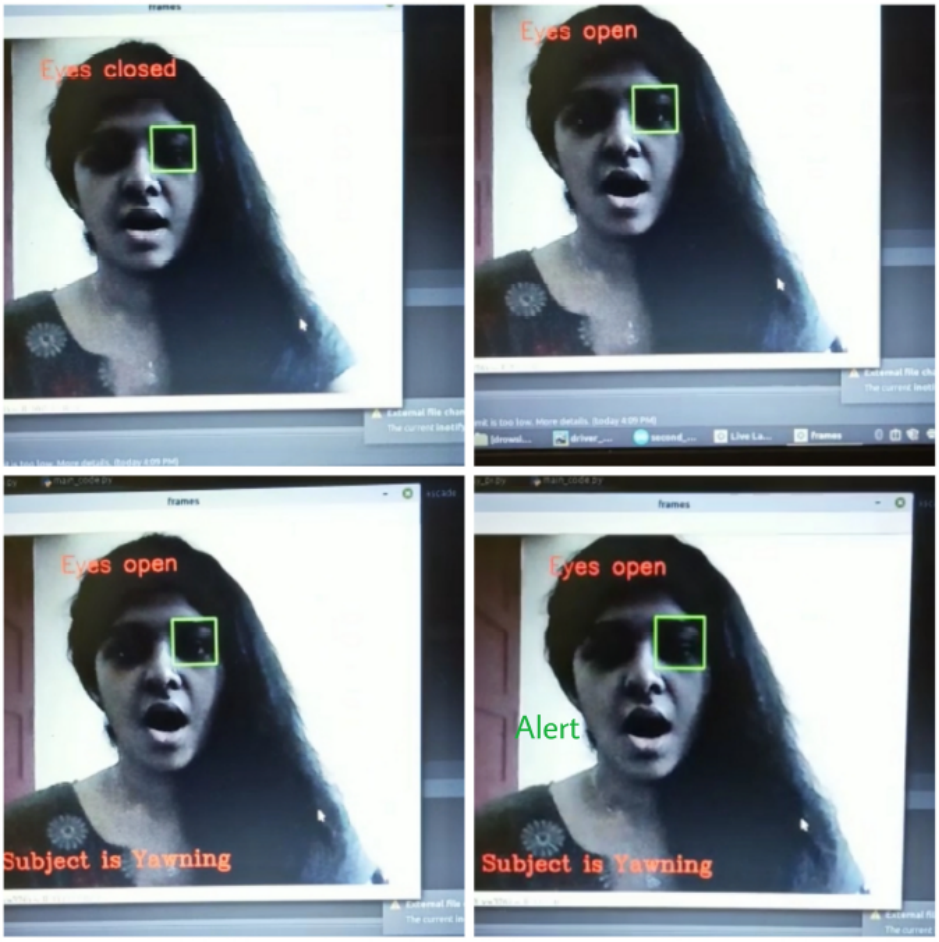



**Fig. 5.8. yawning**

# CHAPTER 6

# SYSTEM IMPLEMENTATION

## 6.1 Implementation Methods

Software implementation refers to the process of adopting and integrating a software application into a business workflow. Implementation of new tools and software into an enterprise can be complex, depending on the size of the organization and the software. Prior to implementation, the software should be selected by assessing needs, budget, potential benefits, obstacles, and so forth. Once the solution is chosen, implementation can begin. For software implementation initiatives to succeed, it's important to follow a few key steps:

- Pilot Program – Prior to full-scale installation, pre-test the software with a test group within the company
- Installation – The IT department should work with the vendor to install the application across all target machines
- Onboarding and Training – Once installed, develop an onboarding program and a software training program to ensure employees are able to utilize all functions and features

Software implementation can be costly and time-consuming. For this reason, many businesses use digital adoption platforms (DAPs) training and gain insights from app usage data. DAPs provide in-app training and guidance, which significantly

reduces human labor time, training time, and employee frustration.Regardless of the software being implemented, businesses should seriously invest time and energy into effective training programs. Doing so will increase the chances of a successful implementation, boost productivity, and increase employee satisfaction In order to fulfill all of the challenging prerequisites we have to test out limitations of our chosen off-the-shelf components. Some of our preliminary tests were devised to help us out with that. We started by setting our work environment in MATLAB. For the purposes of testing, we simplified our dynamic solution to its core, into a horizontal, linear solution that can perform basic face/eye detection as well as simple classification task of determining eye state. The goal of the devised set of experiments is to test the basic performance in various ways. From the very beginning i decided to use ViolaJones algorithm, Haar-like feature based face/eyes detection algorithm available in MATLAB.

It is known for being stable and computationally non-intensive algorithm. Haar-like feature can be described as a set of two or more adjacent rectangular regions organized in a specific way in a given detection window. This algorithm proves to be compute non-intensive since it's core operation is simple summing up of the pixel intensities in a given regions, followed by calculation of differences between them. That difference is then used to categorize subsections of an image to be face/eyes or not. Also, for differentiating between open and closed eyes we decided to go with a proven two-class Harcascade classifier. HCC is a non-probabilistic binary linear classifier. It takes a set of input data and predicts, for each given input, which of two possible classes forms the output. This is perfect since our basic problem can be defined as two class problem: open eyes vs. closed eyes.

## 6.2 Implementation Plan

Implementing new software is always a challenge, even for the most flexible companies. Succeeding demands planning transitions smoothly and providing your team with relevant training material on the new software. implementation process steps:

- Planning Ahead: Always start with a plan! Identify which processes and teams are going to be affected by the implementation and create a shared timeline for carrying the execution gradually. At this stage, you should try and question every possible thing that may get affected by the new software:
- Process Design: Process design is a technique that allows you to organize and run things more efficiently, no matter whether it's a business, software or a team. Most of the commercially available software nowadays has been created using process design methods. This is why when implementing new software you will probably need to adjust many of your new processes to the logic that was used for building that software.
- Solution Design: Once you have created the process design, it is time to work on the solution design, which is essentially a roadmap of business requirements and processes.
- Configuration and Customization: It is now time to install the software and proceed to configure those features that can be used immediately. As a matter of fact, this step should always come first, before defining any processes or rushing to customize any module.
- Integration: Integration is a critical step within software implementation and it involves migrating data from one system to another. With proper integrations, you can save your team from having to copy data between systems manually. They will thank you for having thought about them

- Reporting: This phase is about understanding what information is valuable for your teams in order to improve their decision-making process on a daily basis.
- Training Testing Training may come in different forms and to different groups, from educating your project team on the new software to teaching the end-user how it works. Like training, testing is also an ongoing process that should be done throughout the software implementation process, but also after the team has been using the software for a while.
- Here i implimented this software in car or any other vehicle.It can reduce the accidents by means of drowsiness in driver.The hardware portion include only a camera and a alarm.

# CHAPTER 7

# CONCLUSION AND FUTURE SCOPE

The Driver drowsiness detection and system developed is capable of detecting drowsiness in a rapid manner. The system which can differentiate normal eye blink,Yawning and drowsiness which can prevent the driver from entering the state of sleepiness while driving. The system works well even in case of drivers wearing spectacles and under low light conditions also. During the monitoring, the system is able to decide if the eyes are opened or closed. When the eyes have been closed for about two seconds, the alarm beeps to alert the driver. By doing this many accidents will reduced and provides safe life to the driver and vehicle safety. A system for driver safety and car security is presented only in the luxurious costly cars. Using drowsiness detection system, driver safety can be implemented in normal cars also. The future works may focus on the utilization of outer factors such as vehicle states, sleeping hours, weather conditions, mechanical data, etc, for fatigue measurement.Monitoring the driver's state of drowsiness and vigilance and providing feedback on their condition so that they can take appropriate action is one crucial step in a series of preventive measures necessary to address this problem. Currently there is not adjustment in zoom or direction of the camera during operation. Future work may be to automatically zoom in on the eyes once they are localized.

# REFERENCE


[1] *D*rowsiness Detection System Using Heartbeat Rate in Android-based Handheld Devices

[2] *E*mbedded based Drowsiness Detection using EEG Signals.

[3] *D*river Drowsiness Detection System Based on Visual Features

[4] *D*evelopment of Drowsiness detection system based on respiration changes using Heart rate monitoring

[5] *A*ndroid-Based Application To Detect Drowsiness When Driving Vehicle