

DRIVER DROWSINESS DETECTION SYSTEM

A

Project Report

Submitted

In partial fulfillment for the award
of the Degree of

Bachelor of Technology
In

Computer Science & Engineering



Supervisor:-

Dr. C.P. GUPTA
(PROFESSOR)

Mr. KANHAIYA KUMAR
(ASSISTANT PROFESSOR)

Submitted By:

HARSH BHATT(16\452)

JAI SIPANI(16\454)

JATIN SHARMA(16\455)

KSHITIJ MISHRA(16\459)

Submitted to :-

Department of Computer Science & Engineering

Rajasthan Technical University, Kota

Session (2019-20)



CERTIFICATE

Certified that minor project entitled “**Driver Drowsiness Detection System**” is a bonafide work carried out in the VIII semester in partial fulfillment by the team for award of Bachelor of Technology in Computer Science and Engineering from Rajasthan Technical University, University Departments, Kota during the academic year 2019-2020.

Team Members

Harsh Bhatt – 16/452

Jai Sipani – 16/454

Jatin Sharma – 16/455

Kshitij Mishra – 16/459

Dr. C.P. Gupta
(Professor)

Mr. Kanhaiya Kumar
(Assist. Professor)

CANDIDATE'S DECLARATION

We hereby declare that the work, which is being presented in the Project, entitled “**Driver Drowsiness Detection System** ” in partial fulfillment for the award of Degree of “**Bachelor of Technology**” in Department of Computer Science & Engineering with Specialization in Computer Engineering, and submitted to the **Department of Computer Science & Engineering, University Teaching Department, Rajasthan Technical University** is a record of our own investigations carried under the Guidance of **Dr. C.P. Gupta, Professor** and **Mr. Kanhaiya Kumar, Assistant Professor** Department of Computer Science & Engineering. We have not submitted the matter presented in this Report from anywhere for the award of any other Degree.

(Sign. of Candidate)	(Sign. of Candidate)	(Sign. of Candidate)	(Sign. of Candidate)
Harsh Bhatt	Jai Sipani	Jatin Sharma	Kshitij Mishra
Roll No. 16/452	Roll No. 16/454	Roll No. 16/455	Roll No. 16/459

ACKNOWLEDGEMENT

With great pleasure and deep sense of gratitude, we express our indebtedness to **Prof. C.P. Gupta sir** and **Assist. Prof. Kanhaiya Kumar sir** for their invaluable guidance and constant encouragement at each and every step of our major project. They exposed us to the intricacies of relevant topics and always showed great interest in providing timely support and suitable suggestions.

We are also thankful to **Assist. Prof. Dinesh Soni sir** and **Asso. Prof. R.K. Banyal sir** for their constant encouragement, support and guidance. We would like to extend our gratitude to all teaching staff of Computer Science and Engineering Department, University Departments, RTU, Kota for their guidance and support.

ABSTRACT

Every year the amounts of deaths and injuries in traffic accidents are constantly increasing due to increasing issue of traffic congestion and partly due to human carelessness. Sleeplessness and driving is a very hazardous combination. After alcohol, drowsiness is the second leading cause of the road accidents. In 2018, India had 467,044 reported road accidents, an increase of 0.5% from 464,910 in 2017, according to the Road Ministry's data. Exhausted drivers who doze off at the wheel are responsible for about 40% of road accidents, according to a study by the Central Road Research Institute (CRRI). Most of the traditional methods to detect drowsiness are based on behavioral aspects while some are intrusive and may distract drivers, while some require expensive sensors.

Therefore, in this scenario, a light-weight, real time driver's drowsiness detection system is developed that alerts the driver by beeping an alarm when he closes his eyes for a set period of time due to sleeplessness while driving. The system captures state of eyes in each frame and detects face and data using Haar Classifier, the data is fed to our deep learning model (CNN) which classifies the state of eyes(open or closed). A threshold value is set for the alarm to beep and alert the driver. Drowsiness detection is a safety technology that can prevent the majority of accidents happening on roads and save lives of countless people.

Keywords: Driver, Drowsiness, Detection.

TABLE OF CONTENTS

TOPIC	PAGE NO
1. INTRODUCTION.....	1
1.1 WHAT IS DROWSINESS?	1
1.2 MOTIVATION.....	1
1.3 PROBLEM STATEMENT.....	1
1.4 PROPOSED SOLUTION.....	2
2. FEASIBILITY ANALYSIS.....	3
2.1 TECHNICAL FEASIBILITY.....	3
2.2 ECONOMIC FEASIBILITY.....	3
2.3 OPERATIONAL FEASIBILITY.....	3
3. REQUIREMENT ANALYSIS.....	4
3.1 HARDWARE REQUIREMENTS.....	4
3.2 SOFTWARE REQUIREMENTS.....	4
4. LITERATURE SURVEY.....	5
4.1 STAGES OF DROWSINESS.....	5
4.2 FEATURES ON WHICH DROWSINESS DEPENDS.....	5
4.3 MEASURES FOR DETECTION OF DROWSINESS.....	6
4.4 RESEARCHES BASED ON BEHAVIORAL MEASURES.....	7
4.4.1 TEMPLATE MATCHING.....	7
4.4.2 PCA AND LDA FOR BLINK DETECTION.....	7
4.4.3 HARR CASCADE CLASSIFIER.....	8
4.5 RESEARCHES BASED ON PHYSIOLOGICAL MEASURES.....	8
5. OBJECT DETECTION ALGORITHMS.....	9
5.1 CONVOLUTIONAL NEURAL NETWORKS.....	9
5.1.1 INPUT IMAGE.....	10
5.1.2 CONVOLUTION LAYER—THE KERNEL.....	10
5.1.3 STRIDES.....	13
5.1.4 PADDING.....	13
5.1.5 POOLING.....	13
5.1.6 NON-LINEARITY (RELU).....	15
5.1.7 CLASSIFICATION – FULLY CONNECTED LAYER.....	15
5.1.8 CNN ARCHITECTURE.....	16

5.2 VIOLA- JONES OBJECT DETECTION FRAMEWORK.....	17
5.2.1 HAAR FEATURE SELECTION.....	17
5.2.2 CREATING AN INTEGRAL IMAGE.....	18
5.2.3 ADABOOST TRAINING.....	19
5.2.4 CASCADE CLASSIFIERS.....	19
5.2.5 MODIFICATIONS IN ALGORITHM.....	20
6. PROJECT PLANNING.....	21
6.1 PROJECT WORKFLOW.....	21
6.2 FLOWCHART OF THE PROPOSED SYSTEM.....	22
6.3 MODULES.....	23
7. RESULTS AND VISUALIZATIONS.....	24
7.1 TEST CASES.....	24
7.2 RESULTS.....	28
7.3 CHALLENGES.....	28
8. CONCLUSION AND FUTURE SCOPE.....	29
8.1 CONCLUSION.....	29
8.2 FUTURE SCOPE.....	29
REFERENCES.....	30

LIST OF FIGURES

	PAGE NO
Figure 1: Flattening Of A 3X3 Image Matrix Into A 9X1 Vector.....	9
Figure 2: 4X4X3 RGB Image.....	10
Figure 3: Convoluting A 5X5X1 Image With A 3X3X1 Kernel To Get A 3X3X1 Convolved Feature.....	11
Figure 4: Convolution Operation On A Mxnx3 Image Matrix With A 3X3X3 Kernel.....	11
Figure 5: Same Padding: 5X5X1 Image Is Padded With 0S To Create A 6X6X1 Image.....	12
Figure 6: Stride Of 2 Pixels.....	13
Figure 7: 3X3 Pooling Over 5X5 Convolved Feature.....	14
Figure 8: Types Of Pooling.....	14
Figure 9: Relu Operation.....	15
Figure 10: Fully Connected Layer.....	16
Figure 11: Haar Features.....	17
Figure 12: Integral Image Formation.....	18
Figure 13: Integral Image Calculation.....	18
Figure 14: Formulation Of Adaboost.....	19
Figure 15 : A Cascade Of 'N' Classifiers For Face Detection.....	20
Figure 16: Modules.....	23
Figure 17: Haar Cascade Files For Face And Eye Detection.....	23

CHAPTER 1

INTRODUCTION

1.1 What is Drowsiness?

Drowsiness is defined as a decreased level of awareness portrayed by sleepiness and trouble in staying alarm but the person awakes with simple excitement by stimuli. It might be caused by an absence of rest, medicine, substance misuse, or a cerebral issue. It is mostly the result of fatigue which can be both mental and physical. Physical fatigue, or muscle weariness, is the temporary physical failure of a muscle to perform ideally. Mental fatigue is a temporary failure to keep up ideal psychological execution. The onset of mental exhaustion amid any intellectual action is progressive, and relies on an individual's psychological capacity, furthermore upon different elements, for example, lack of sleep and general well-being. Mental exhaustion has additionally been appeared to diminish physical performance. It can show as sleepiness, dormancy, or coordinated consideration weakness.

In the past years according to available data driver sleepiness has gotten to be one of the real reasons for street mishaps prompting demise and extreme physical injuries and loss of economy. A driver who falls asleep is in an edge of losing control over the vehicle prompting crash with other vehicle or stationary bodies. Keeping in mind to stop or reduce the number of accidents to a great extent the condition of sleepiness of the driver should be observed continuously.[6]

1.2 Motivation

Every year the amounts of deaths and injuries in traffic accidents are constantly increasing due to increasing issue of traffic congestion and partly due to human carelessness. Sleeplessness and driving is a very hazardous combination. After alcohol, drowsiness is the second leading cause of the road accidents.

In 2018, India had 467,044 reported road accidents, an increase of 0.5% from 464,910 in 2017, according to the Road Ministry's data. Exhausted drivers who doze off at the wheel are responsible for about 40% of road accidents, according to a study by the Central Road Research Institute (CRRI). These heartbreaking stats encouraged and motivated us to build this project for the betterment of society.

1.3 Problem Statement

Countless people such as truck drivers, interstate taxi drivers or lorry drivers drive for long hours day and night without adequate sleep or food on highways which puts them as well as other people in danger. Design a solution for reducing the number of accidents happening due to drowsiness while driving.

1.4 Proposed Solution

We have built a Driver Drowsiness Detection System using Python and its libraries that alerts the driver by beeping an alarm when he closes his eyes for a set period of time due to sleeplessness while driving.

Convolutional neural networks (CNNs) are capable of learning rich mid-level image representations that have been proven to be effective for many vision recognition tasks, such as object detection.

The strategy is to have the CNNs to learn good representations on large scale data in either an unsupervised or a supervised manner. Once the CNNs have learned representations of visual features, these representations can be transferred to another domains, with or without adapting them to represent domain-specific features.

The project is broken down into multiple steps:

1. Load and pre-process the image dataset. The dataset used in this project is manually generated by capturing images in different brightness levels and different states of eyes such as open, partially open or closed.
2. Train the classifier on our dataset. We used convolutional neural network to build our model.
3. Use the trained classifier to detect the eyes of driver and classify them as open or closed. If eyes are closed for a set period of time an alarm is beeped to wake up the driver. [9]

CHAPTER 2

FEASIBILITY ANALYSIS

Proposed system is feasible considering the technical, economical and operation factors. By having a detailed feasibility study the management will have a clear view of the proposed system with its benefits and drawbacks. The system has been tested for feasibility in the following aspects.

- Technical Feasibility
- Economic Feasibility
- Operational Feasibility

2.1 Technical Feasibility

The application can be run on any system using the python interpreter. The system is compatible with all the systems that can run a python code provided all the necessary libraries required to run this project are downloaded and installed on the system. Therefore in the forthcoming years, this will not be outdated if for any new implementation technologies kicks in.

2.2 Economic Feasibility

As of now, this architecture has been established for the project purpose only. So, we have used all the modules and functionalities of the python language that are freely available to everyone.

Also, if one wishes to take this to the next level i.e. for business purposes or deploying this architecture on a large working scale, the services included can be upgraded so as to inculcate all the latest features and security services so as to function appropriately. A camera with good lens and resolution power can be used to further improve the efficiency with increase in cost.

2.3 Operational Feasibility

The main advantage of this project is that it is made in python language that is widely used and can be compiled and run on any system having a python interpreter and the necessary libraries. The only requirement is a fast system to train the model and detect face and eyes of a person.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 Hardware requirements

- **Processor** Intel i3 processor 2000 GHz or higher
- **RAM** Minimum 8 GB primary memory
- **Monitor** Preferably color monitor (16 bit) and above
- **Hard Disk** Minimum 64 GB
- **GPU** NVIDIA-GTX 1050 Ti
- **Camera** Webcam/ Dashboard Camera

3.2 Software Requirements

- **Operating System** Windows Operating System/ Linux / Mac OS
- **Language** Python 3
- **Python Libraries** Tensorflow, Keras, PyGame, OpenCV
- **Text Editor** VS code/ Sublime text

LITERATURE SURVEY

4.1 Stages of drowsiness

The term “drowsy” is substitutable with sleepy, that merely means that an inclination to fall asleep. The stage of sleep is often classified as awake, non-rapid eye movement sleep (NREM), and rapid eye movement sleep (REM). The second stage, NREM, is often divided into the subsequent 3 stages.

- **Stage I** : Transition from awake to sleepy (drowsy).
- **Stage II** : Light-Weight Sleep.
- **Stage III** : Deep Sleep.

4.2 Features on which drowsiness depends

One of the challenges in developing an economical drowsiness detection system is a way to acquire proper drowsiness information. Because of safety reasons, drowsiness cannot be manipulated during a real environment, Therefore the drowsiness detection system needs to be developed and tested in a laboratory setting. However, in a laboratory setting, the foremost reliable and informative information that pertains to driver drowsiness depends only on the approach in which the driver falls into the drowsy state. Driver drowsiness principally depends on the quality of the last sleep, the biological time (time of day) and the rise within the period of the driving task. In some analysis experiments, the subjects were totally deprived of sleep, whereas they were only part deprived of sleep in others [1].

Additionally, some researchers recruited night shift staff as their subjects, in this case, the subjects were entirely deprived of sleep as results of the experiments were conducted within the morning. Kokonozi, et al. Conducted an experiment during which they monitored the participants for twenty four before the experiment began to make sure that they were utterly sleep deprived [1]. In certain experiments, researches partly deprived the subjects of sleep by permitting them to sleep for less than a half dozen. Peters, et al. Studies an equivalent subject throughout four consecutive days and regarded the results of no sleep deprivation, partial sleep deprivation and total sleep deprivation on their drowsiness level [2]. They discovered that, even within the case of partial sleep deprivation, the subjects tend to urge drowsy after a while. Hence, the standard of the last sleep is a crucial criterion that influences drowsiness. Otamani, et al. found that sleep deprivation alone doesn't directly influence the brain signals that control, drowsiness,

whereas the period of the task includes a strong influence [3]. Researchers have additionally inferred that prolonged driving on a boring setting stimulates drowsiness. In fact, it has been discovered that the subjects will become drowsy at intervals twenty to twenty five min of driving.

4.3 Measures for detection of drowsiness

The study states that the reason for a mishap can be categorized as one of the accompanying primary classes: human, vehicular, and surrounding factor.

The driver's error represented 91% of the accidents. The other two classes of causative elements were referred to as 4% for the type of vehicle used and 5% for surrounding factors. Several measures are available for the measurement of drowsiness which includes the following:

1. Vehicle based measures.
2. Physiological measures.
3. Behavioral measures.

1. Vehicle based measures: Vehicle-based measures survey path position, which monitors the vehicle's position as it identifies with path markings, to determine driver weakness, and accumulate steering wheel movement information to characterize the fatigue from low level to high level. In many research project, researchers have used this method to detect fatigue, highlighting the continuous nature of this non-intrusive and cost-effective monitoring technique. This is done by:

1. Sudden deviation of vehicle from lane position.
2. Sudden movement of steering wheels.
3. Pressure on acceleration paddles.

For each measures threshold values are decided which when crossed indicated that driver is drowsy.

Advantages:

1. It is non-invasive in nature.
2. Provides almost accurate result.

Disadvantages:

1. Vehicle based measures mostly affected by the geometry of road which sometimes unnecessarily activates the alarming system.
2. The driving style of the current driver needs to be learned and modeled for the system to be efficient.
3. The condition like micro sleeping which mostly happens in straight highways cannot be detected.

2. Physiological measures: Physiological measures are the objective measures of the physical changes that occur in our body because of fatigue. These physiological changes can be simply measure by their respective instruments as follows:

- ECG (electro cardiogram) : For monitoring heart-rate.
- EMG (electromyogram) : For monitoring electrical activity of muscles.
- EOG (electrooculogram) : For monitoring eyeball movements.
- EEG (electroencephalogram): For monitoring brain waves.

3. Behavioral measures: Certain behavioral changes take place during drowsing like:

1. Yawning
2. Amount of eye closure
3. Eye blinking
4. Head position

4.4 Researches based on behavioral measures

4.4.1 Template matching

Template matching is a method for discovering zones of a picture that match to a format picture. There are two picture classifications the source picture the picture in which we hope to discover a match to the format picture and the Template picture the patch picture which will be contrasted with the format picture. To recognize the matching territory, must be contrasting the format picture against the picture by sliding. Sliding is moving the patch one pixel at once (left to right, up to down). At every area, a metric is computed. So it represents how “Great” or “Terrible” the match at that area is (or how comparable the patch is in that specific territory of the source picture). The brightest areas indicate the highest matches [4].

4.4.2 Principal Component Analysis (PCA) and Linear Discriminate Analysis (LDA) for Blink Detection

J. Lee, H. Jung, K.R. Park and J. Kim propose another driver checking system considering driver tiredness and diversion [9]. In the event that the driver is looking ahead, tiredness identification is performed. If not diversion discovery is performed. Besides, another eye recognition, calculation is presented. It joins versatile boosting, versatile layout matching, and blob discovery with eye acceptance. Those calculations diminish the eye discovery lapse and handling time essentially, by accomplished the said calculations. Third, they have used principal component analysis (PCA), and linear discriminate analysis (LDA) with a specific end goal to attain exact eye identification. Fourth, they have proposed a novel eye state detection calculation that joins appearances gimmicks got utilizing PCA and LDA, with measurable peculiarities.

4.4.3 Harr cascade Classifier

J. Suryaprasad classifies the method for face/eye detection methods utilizing image processing in real time [9]. In this project, it further clarifies the method for utilizing the harr cascade tests and the separation of eye blink and drowsiness identification. This paper acquaints a vision based strategy with distinguishing the drowsiness. The significant difficulties are face recognition, Iris location under different conditions and creating the real time system.

4.5 Researches on Drowsiness Detection Using Physiological Measures

Many researchers consider the subsequent physiological signals to observe drowsiness, electrocardiogram (ECG), electro encephalogram (EEG). The heart rate (HR) additionally varies considerably between the different stage of drowsiness, like alertness and fatigue. Therefore, heart rate, which may be simply determined by the ECG signal, can even be used to observe drowsiness. Others have measured drowsiness using heart rate variability (HRV), within which the low (LF) and high (HF) frequencies fall within the range of 0.04-0.15 Hertz and 0.14-0.4 Hertz respectively, shows a physiological signal sensing system that may be integrated into vehicles to observe driver drowsiness. But still, compared to the drowsiness detection using behavioral measures, numbers of researches on physiological measures to detect drowsiness are low.

Real time Nonintrusive Detection of Driver Drowsiness [6] project has done by Xun Yu, University of Minnesota Duluth. In this research they have mainly concern on the detection of the drowsiness using heart rate of the driver. This system aims to develop a real time; nonintrusive driver drowsiness detection system to cut back drowsiness caused accidents. Biosensor is built on the vehicle steering wheel to read driver's heartbeat signals. Heart rate variability (HRV), a physiological signal that has established links to waking/ sleeping stages, therefore can be analyzed from the pulse signals for the detection of driver drowsiness.

OBJECT DETECTION ALGORITHMS

5.1 Convolutional Neural Networks

Artificial Intelligence has been witnessing a monumental growth in bridging the gap between the capabilities of humans and machines. Researchers and enthusiasts alike, work on numerous aspects of the field to make amazing things happen. One of many such areas is the domain is Computer Vision.

The agenda for this is to enable machines to view the world as humans do, perceive it in a similar manner and even use the knowledge for a multitude of tasks such as Image & Video recognition, **Image Classification** & Analysis, Media Recreation, Recommendation Systems, Natural Language Processing, etc. The advancements in Computer Vision with Deep Learning has been constructed and perfected with time, primarily over one particular algorithm known as **Convolutional Neural Network**.

An image is nothing but a matrix of pixel values. So we flatten the image (e.g. 3x3 image matrix into a 9x1 vector) and feed it to a Multi-Level Perceptron for classification purposes.

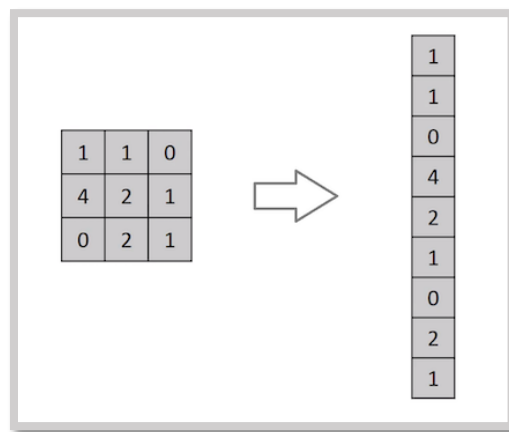


Figure 1: Flattening of a 3x3 image matrix into a 9x1 vector

In cases of extremely basic binary images, the method might show an average precision score while performing prediction of classes but would have little to no accuracy when it comes to complex images having pixel dependencies throughout.

A Conv-Net is able to **successfully capture the Spatial and Temporal dependencies** in an image through the application of relevant filters. The architecture performs a better fitting to the image dataset due to the reduction in the number of parameters involved and reusability of weights. In other words, the network can be trained to understand the sophistication of the image better.

5.1.1 Input Image

In the figure, we have an RGB image which has been separated by its three-color planes—Red, Green, and Blue. There are a number of such color spaces in which images exist—Grayscale, RGB, HSV, CMYK, etc.

We can imagine how computationally intensive things would get once the images reach dimensions, say 8K (7680×4320). The role of the Conv-Net is to reduce the images into a form which is easier to process, without losing features which are critical for getting a good prediction. This is important when we are to design an architecture which is not only good at learning features but also is scalable to massive datasets.

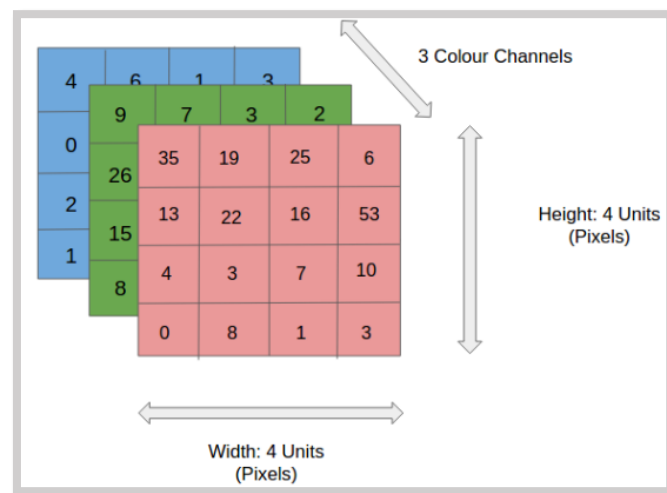


Figure 2: 4x4x3 RGB image

5.1.2 Convolution Layer—The Kernel

Image Dimensions = 5 (Height) x 5 (Breadth) x 1 (Number of channels, e.g. RGB)

In the above demonstration, the green section resembles our **5x5x1 input image, I**. The element involved in carrying out the convolution operation in the first part of a Convolutional Layer is called the **Kernel/Filter, K**, represented in the color yellow.

We have selected **K** as a **3x3x1 matrix**.

$$\text{Kernel / Filter } K, = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

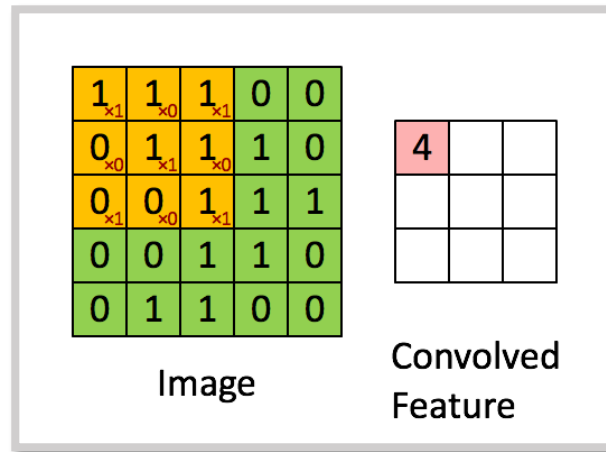


Figure 3: Convoluting a 5x5x1 image with a 3x3x1 kernel to get a 3x3x1 convolved feature

The Kernel shifts 9 times because of **Stride Length = 1 (Non-Strided)**, every time performing a **matrix multiplication operation between K and the portion P of the image** over which the kernel is hovering.

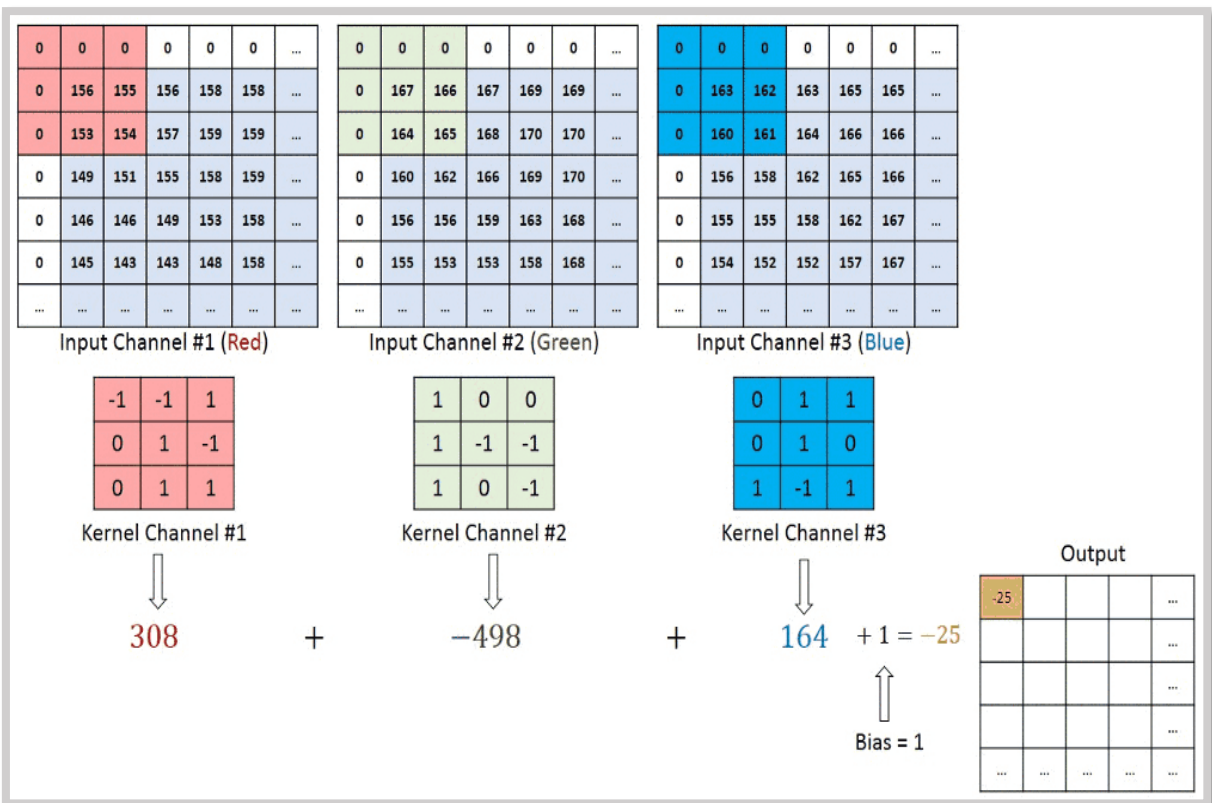


Figure 4: Convolution operation on a MxNx3 image matrix with a 3x3x3 Kernel

The filter moves to the right with a certain Stride Value till it parses the complete width. Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed.

In the case of images with multiple channels (e.g. RGB), the Kernel has the same depth as that of the input image. Matrix Multiplication is performed between K_n and I_n stack ($[K1, I1]; [K2, I2]; [K3, I3]$) and all the results are summed with the bias to give us a squashed one-depth channel Convolved Feature Output.

The objective of the Convolution Operation is to **extract the high-level features** such as edges, from the input image. ConvNets need not be limited to only one Convolutional Layer. Conventionally, the first ConvLayer is responsible for capturing the Low-Level features such as edges, color, gradient orientation, etc. With added layers, the architecture adapts to the High-Level features as well, giving us a network, which has the wholesome understanding of images in the dataset, similar to how we would.

There are two types of results to the operation—one in which the convolved feature is reduced in dimensionality as compared to the input, and the other in which the dimensionality is either increased or remains the same. This is done by applying **Valid Padding** in case of the former, or **Same Padding** in the case of the latter.

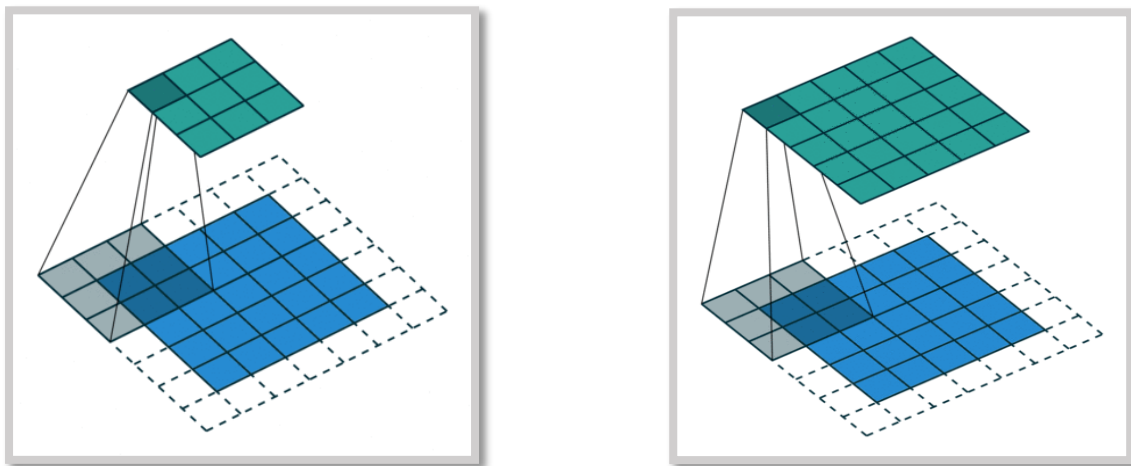


Figure 5: SAME padding: 5x5x1 image is padded with 0s to create a 6x6x1 image

When we augment the 5x5x1 image into a 6x6x1 image and then apply the 3x3x1 kernel over it, we find that the convolved matrix turns out to be of dimensions 5x5x1. Hence the name—**Same Padding**.

On the other hand, if we perform the same operation without padding, we are presented with a matrix which has dimensions of the Kernel (3x3x1) itself—**Valid Padding**.

The following repository houses many such GIFs which would help we get a better understanding of how Padding and Stride Length work together to achieve results relevant to our needs.

5.1.3 Strides

Stride is the number of pixels shifts over the input matrix. When the stride is 1 then we move the filters to 1 pixel at a time. When the stride is 2 then we move the filters to 2 pixels at a time and so on. The below figure shows convolution would work with a stride of 2.

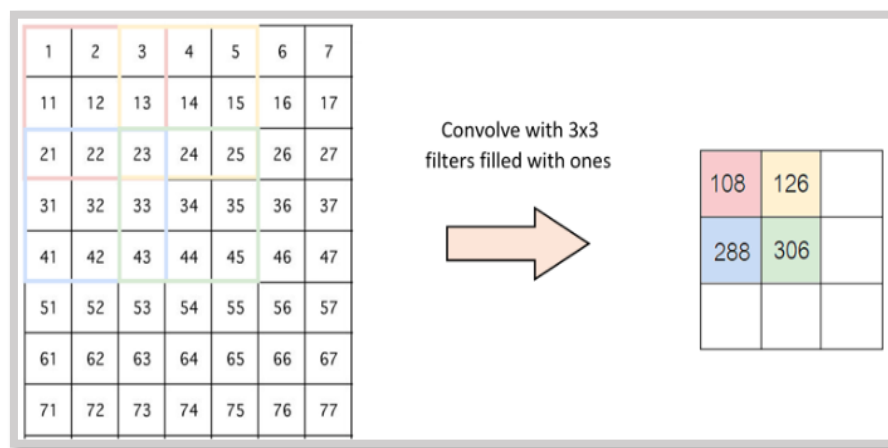


Figure 6: Stride of 2 pixels

5.1.4 Padding

Sometimes filter does not fit perfectly fit the input image. We have two options:

- Pad the picture with zeros (zero-padding) so that it fits.
- Drop the part of the image where the filter did not fit. This is called valid padding which keeps only valid part of the image.

5.1.5 Pooling

Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to **decrease the computational power required to process the data** through dimensionality reduction. Furthermore, it is useful for **extracting dominant features** which are rotational and positional invariant, thus maintaining the process of effectively training of the model.

There are two types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On

the other hand, **Average Pooling** returns the **average of all the values** from the portion of the image covered by the Kernel.

Max Pooling also performs as a **Noise Suppressant**. It discards the noisy activations altogether and also performs de-noising along with dimensionality reduction. On the other hand, Average Pooling simply performs dimensionality reduction as a noise suppressing mechanism. Hence, we can say that **Max Pooling performs a lot better than Average Pooling**.

The Convolutional Layer and the Pooling Layer, together form the i-th layer of a Convolutional Neural Network. Depending on the complexities in the images, the number of such layers may be increased for capturing low-levels details even further, but at the cost of more computational power.

After going through the above process, we have successfully enabled the model to understand the features. Moving on, we are going to flatten the final output and feed it to a regular Neural Network for classification purposes.

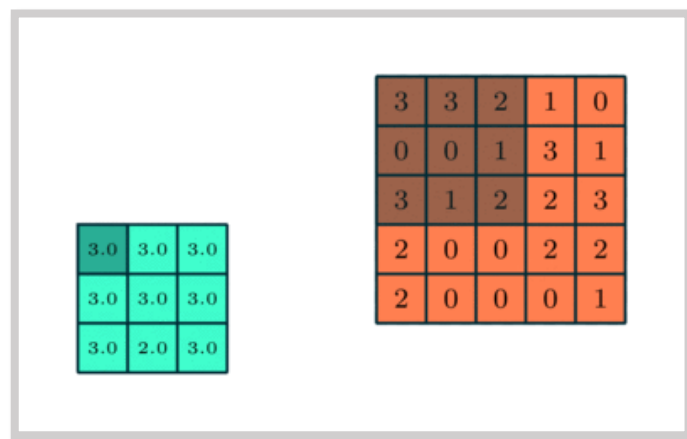


Figure 7: 3x3 pooling over 5x5 convolved feature

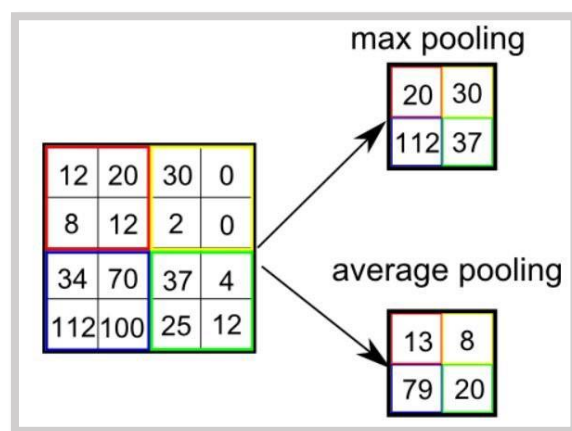


Figure 8: Types of pooling

5.1.6 Non-Linearity (ReLU)

ReLU stands for Rectified Linear Unit for a non-linear operation.

The output is $f(x) = \max(0, x)$.

Why ReLU is important: ReLU's purpose is to introduce non-linearity in Conv-Net. Since, the real-world data would want Conv-Net to learn would be non-negative linear values.

There are other non-linear functions such as tanh or sigmoid can also be used instead of ReLU. Most of the data scientists uses ReLU since performance wise ReLU is better than other two.

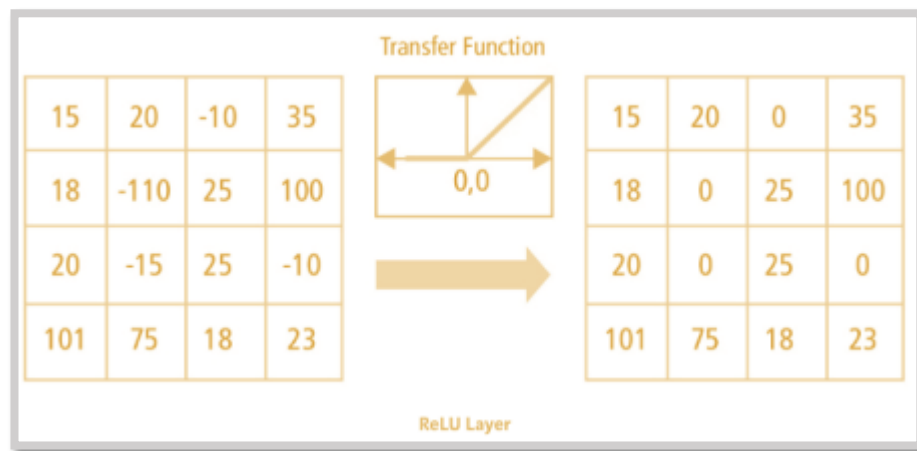


Figure 9: ReLU operation

5.1.7 Classification – Fully Connected Layer

Adding a Fully-Connected layer is a (usually) cheap way of learning non-linear combinations of the high-level features as represented by the output of the convolutional layer. The Fully-Connected layer is learning a possibly non-linear function in that space.

Now that we have converted our input image into a suitable form for our Multi-Level Perceptron, we shall flatten the image into a column vector. The flattened output is fed to a feed-forward neural network and backpropagation applied to every iteration of training. Over a series of epochs, the model is able to distinguish between dominating and certain low-level features in images and classify them using the **SoftMax Classification** technique.

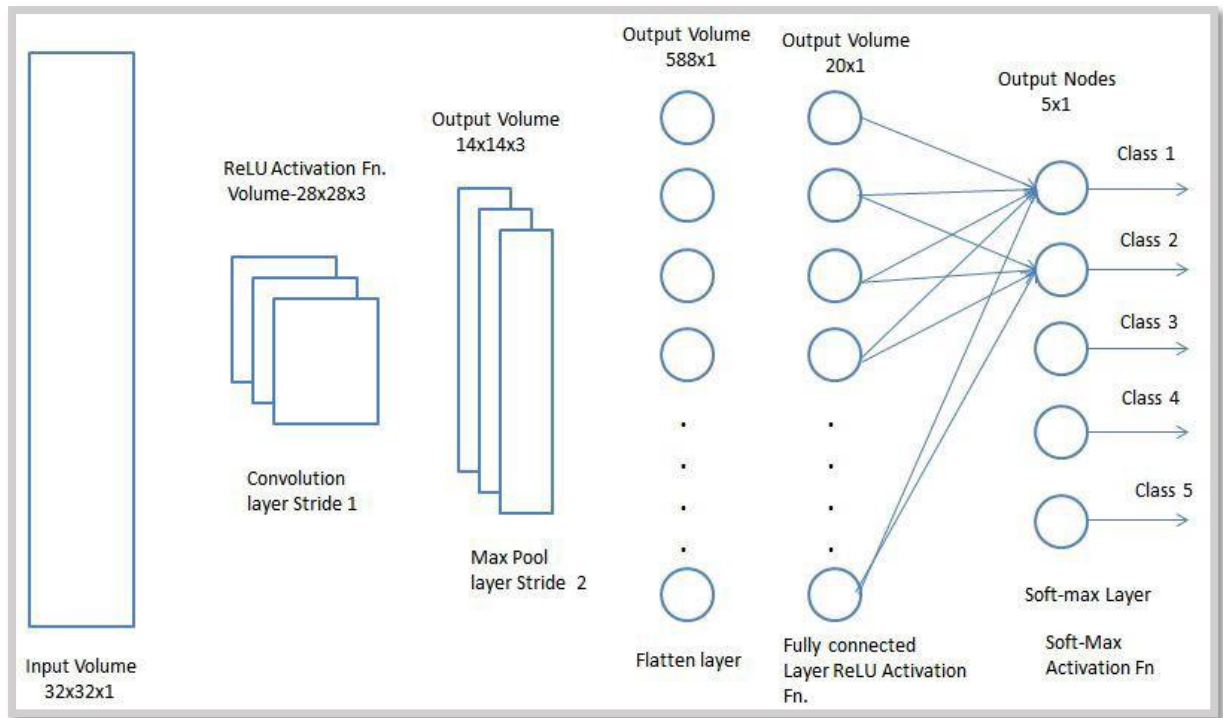


Figure 10: Fully Connected Layer

5.1.8 CNN Architecture

The model we used is built with Keras using **Convolutional Neural Networks (CNN)**. A convolutional neural network is a special type of deep neural network which performs extremely well for image classification purposes. A CNN basically consists of an input layer, an output layer and a hidden layer which can have multiple numbers of layers. A convolution operation is performed on these layers using a filter that performs 2D matrix multiplication on the layer and filter.

The CNN model architecture consists of the following layers:

- Convolutional layer; 32 nodes, kernel size 3
- Convolutional layer; 32 nodes, kernel size 3
- Convolutional layer; 64 nodes, kernel size 3
- Fully connected layer; 128 nodes

The final layer is also a fully connected layer with 2 nodes. In all the layers, a Relu activation function is used except the output layer in which we used Softmax.

5.2 Viola- Jones object detection framework

The face detection method used in OpenCv is developed in 2001 by Paul Viola and Michael Jones, very well referred to as the Viola-Jones method. Though this method can be used for several objects but most specifically here it is used for face and eye detection in real time. Viola-Jones algorithm has four stages:

1. Haar Feature Selection
2. Creating an Integral Image
3. Adaboost Training
4. Cascading Classifiers

5.2.1 Haar Feature Selection

All human faces share some similar properties. These regularities may be matched using **Haar Features**.

A few properties common to human faces:

- The eye region is darker than the upper-cheeks.
- The nose bridge region is brighter than the eyes.

Composition of properties forming matchable facial features:

- Location and size: eyes, mouth, bridge of nose.
- Value: oriented gradients of pixel intensities.

The four features matched by this algorithm are then sought in the image of a face.

Rectangle features:

- Value = Σ (pixels in black area) - Σ (pixels in white area).
- Three types: two-, three-, four-rectangles, Viola & Jones used two-rectangle features

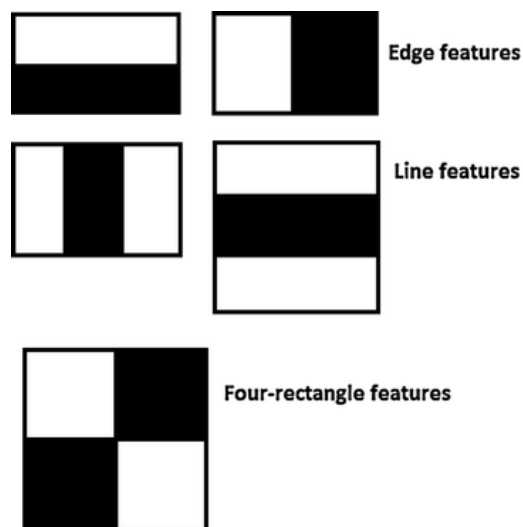


Figure 11: Haar features

- For example: the difference in brightness between the white & black rectangles over a specific area.
- Each feature is related to a special location in the sub-window.

5.2.2 Creating an Integral Image

We know each point of an image is represented by a pixel value. As we so we need to know the output of applied Haar features so we need to find the sum of pixel value of all those area and solve the summation. But this is a huge task. To reduce the number of calculation concept of INTEGRAL IMAGE is introduced.

Definition of Integral Image: Basically Integral image is a matrix same as size of the window. The integral image at location (x, y) is the sum of the pixels above and to the left of (x, y).

0	1	1	1		0	1	2	3
1	2	2	3		1	4	7	11
1	2	1	1		2	7	11	16
1	3	1	0		3	11	16	21

Figure 12: Integral image formation

Here the pixel value of each box is modified by sum of all those box left and above it so that we can use a formula mentioned below to get the output of Haar features with much less calculation reducing the time of calculation.

For example the integral sum of inside rectangle can be computed as:

$ii(d) + ii(a) - ii(b) - ii(c)$; Where ii stands for integral image value.

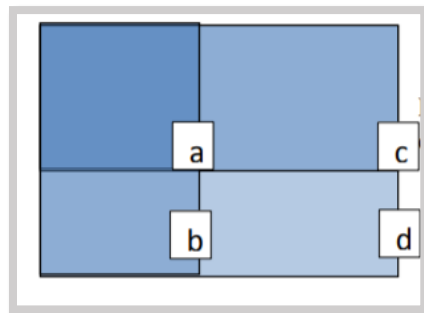


Figure 13: Integral image calculation

Hence if we want to calculate the pixel value of a rectangle we can do so by just taking four points from integral image as above.

5.2.3 AdaBoost training

AdaBoost stands for “Adaptive” boost. Here we construct a strong classifier as linear combination of weak classifier as there are so many features which are absolutely invalid in finding the facial features. It can be formulated as below:

$$F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + \alpha_3 f_3(x) + \dots$$

Strong Classifier

Image

Weight

Weak classifier

Figure 14: Formulation of AdaBoost

Features of weak classifiers:

- Each single rectangle is regarded as a simple weak classifier.
- Each weak classifier is assigned a weigh function as per its importance of position.
- Finally the strong classifier is formed by their linear combination.

5.2.4 Cascade Classifiers

After going through Adaboost stage now let's say we have 600 no of strong classifiers. So to detect if a frame contains a face or not: Instead of applying all the 600 features on a window, group the features into different stages of classifiers and apply one-by-one. If a window fails the first stage, discard it. We don't consider remaining features on it. If it passes, apply the second stage of features and continue the process. The window which passes all stages is a face region.

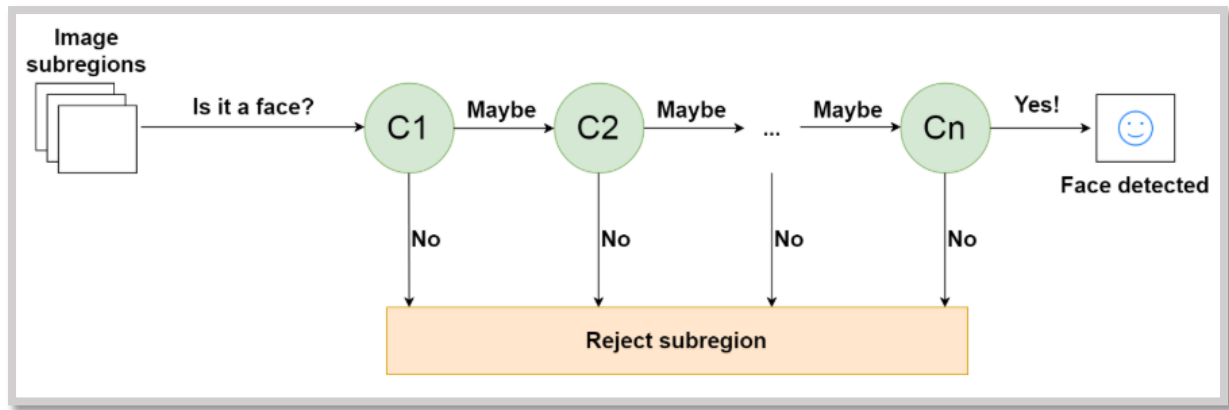


Figure 15 : A cascade of ‘n’ classifiers for face detection

5.2.5 Modifications in Algorithm

Modification in face detection: For detection of face, loading of cascade file is done. Then the requirement is to pass on the captured frame to a function which does detection of edge. After this process almost all the possible type objects may get detected corresponding to different sizes. Hence the task of reduction of processing amount comes here. To achieve this instead of detecting all the objects present in the frame because we know that our required object is face and in most case it occupies almost all the portion of the captured frame. So we can modify the algorithm to detect only in that fashion.

Modification in eye detection: For eye detection as the amount of processing will be very high if we apply the features to all the portion of the face. Hence to avoid this situation we should be interested only in those portion of face where we are certain that eye exist there. We set our work region for finding the eye by taking the following facts:

1. Only upper parts of the face contain the maximum probability of finding an eye.
2. The place of occurrence of eyes are a few pixel below the fore head.

Modification in colour selection: For drowsiness detection instead of using the colored image, the image is converted to gray scale to reduce the no of channel parameters which helps to increase the speed of calculation of the classifiers.

CHAPTER 6

PROJECT PLANNING

6.1 Project Workflow

Let's now understand how our algorithm works step by step.

Step 1 – Take Image as Input from a Camera

With a webcam, we will take images as input. So to access the webcam, we made an infinite loop that will capture each frame. We use the method provided by OpenCV, **cv2.VideoCapture(0)** to access the camera and set the capture object (cap). **cap.read()** will read each frame and we store the image in a frame variable.

Step 2 – Detect Face in the Image and Create a Region of Interest (ROI)

To detect the face in the image, we need to first convert the image into grayscale as the OpenCV algorithm for object detection takes gray images in the input. We don't need color information to detect the objects. We will be using haar cascade classifier to detect faces. This line is used to set our classifier **face = cv2.CascadeClassifier(' path to our haar cascade xml file')**. Then we perform the detection using **faces = face.detectMultiScale(gray)**. It returns an array of detections with x,y coordinates, and height, the width of the boundary box of the object. Now we can iterate over the faces and draw boundary boxes for each face.

```
for (x,y,w,h) in faces:
    cv2.rectangle(frame, (x,y), (x+w, y+h), (100,100,100), 1 )
```

Step 3 – Detect the eyes from ROI and feed it to the classifier

The same procedure to detect faces is used to detect eyes. First, we set the cascade classifier for eyes in **leye** and **reye** respectively then detect the eyes using **left_eye = leye.detectMultiScale(gray)**. Now we need to extract only the eyes data from the full image. This can be achieved by extracting the boundary box of the eye and then we can pull out the eye image from the frame with this code.

```
l_eye = frame[ y : y+h, x : x+w ]
```

l_eye only contains the image data of the eye. This will be fed into our CNN classifier which will predict if eyes are open or closed. Similarly, we will be extracting the right eye into **r_eye**.

Step 4 – Classifier will Categorize whether Eyes are Open or Closed

We are using CNN classifier for predicting the eye status. To feed our image into the model, we need to perform certain operations because the model needs the correct dimensions to start with. First, we convert the color image into grayscale using **r_eye = cv2.cvtColor(r_eye, cv2.COLOR_BGR2GRAY)**. Then, we resize the image to 24*24 pixels as our model was trained on 24*24 pixel images **cv2.resize(r_eye, (24,24))**. We normalize our data for better convergence **r_eye = r_eye/255** (All values will be between 0-1). Expand the dimensions to feed into our classifier. We loaded our model using **model = load_model('models/cnnCat2.h5')**.

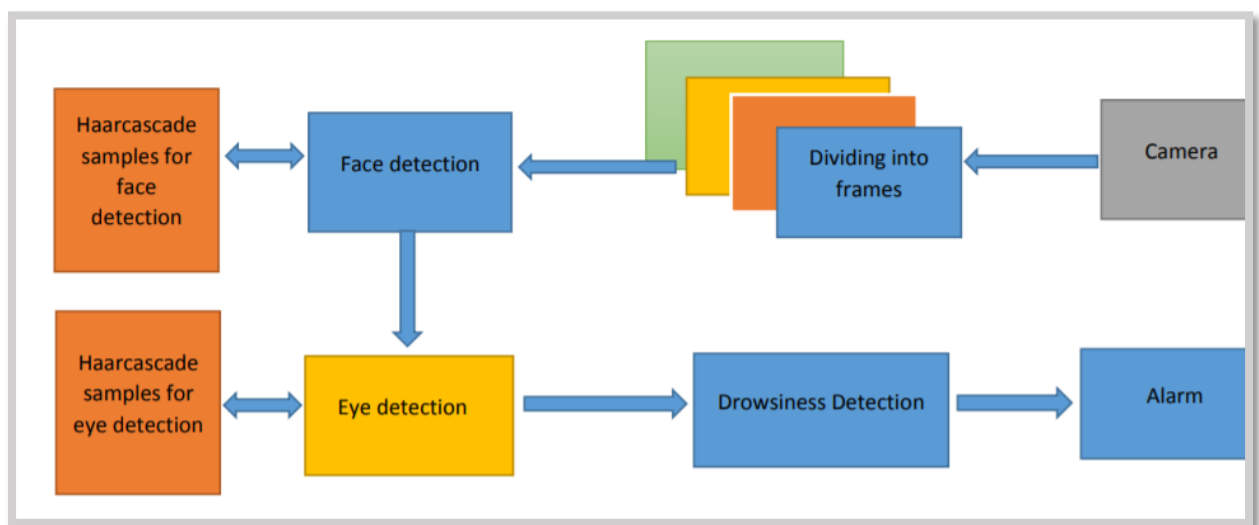
Now we predict each eye with our model **lpred = model.predict_classes(l_eye)**. If the value of **lpred[0] = 1**, it states that eyes are open, if value of **lpred[0] = 0** then, it states that eyes are closed.

Step 5 – Calculate Score to Check whether Person is Drowsy

The score is basically a value we will use to determine how long the person has closed his eyes. So if both eyes are closed, we will keep on increasing score and when eyes are open, we decrease the score. We are drawing the result on the screen using **cv2.putText()** function which will display real time status of the person.

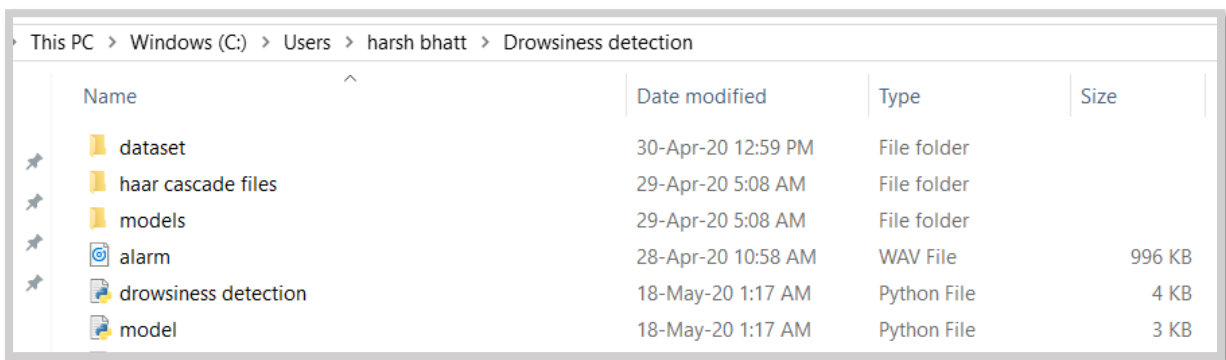
```
cv2.putText(frame, "Open", (10, height-20), font, 1, (255,255,255), 1, cv2.LINE_AA )
```

6.2 Flowchart of the Proposed System



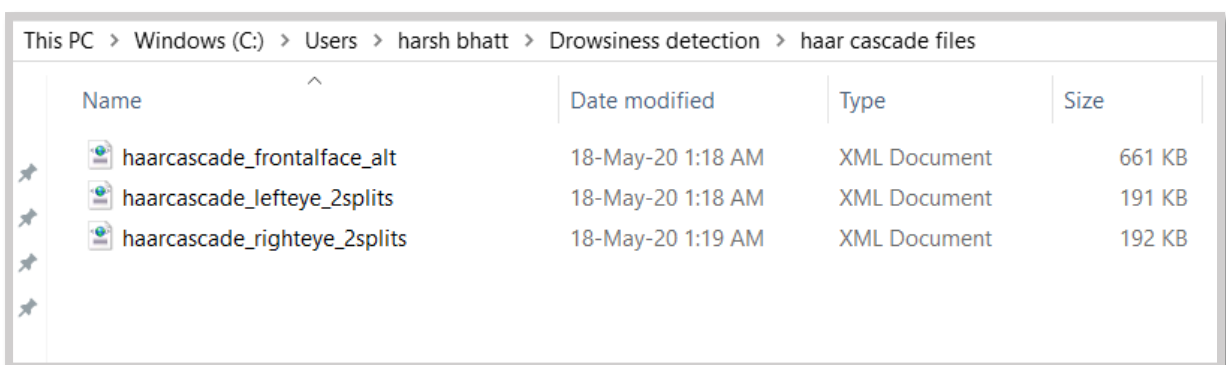
6.3 Modules

- The “haar cascade files” folder consists of the xml files that are needed to detect objects from the image. In our case, we are detecting the face and eyes of the person.
- The models folder contains our model file “cnnCat2.h5” which was trained on convolutional neural networks.
- We have an audio clip “alarm.wav” which is played when the person is feeling drowsy.
- “Model.py” file contains the program through which we built our classification model by training on our dataset. You can see the implementation of convolutional neural network in this file.
- “Drowsiness detection.py” is the main file of our project. To start the detection procedure, we have to run this file.



Name	Date modified	Type	Size
dataset	30-Apr-20 12:59 PM	File folder	
haar cascade files	29-Apr-20 5:08 AM	File folder	
models	29-Apr-20 5:08 AM	File folder	
alarm	28-Apr-20 10:58 AM	WAV File	996 KB
drowsiness detection	18-May-20 1:17 AM	Python File	4 KB
model	18-May-20 1:17 AM	Python File	3 KB

Figure 16: Modules



Name	Date modified	Type	Size
haarcascade_frontalface_alt	18-May-20 1:18 AM	XML Document	661 KB
haarcascade_lefteye_2splits	18-May-20 1:18 AM	XML Document	191 KB
haarcascade_righteye_2splits	18-May-20 1:19 AM	XML Document	192 KB

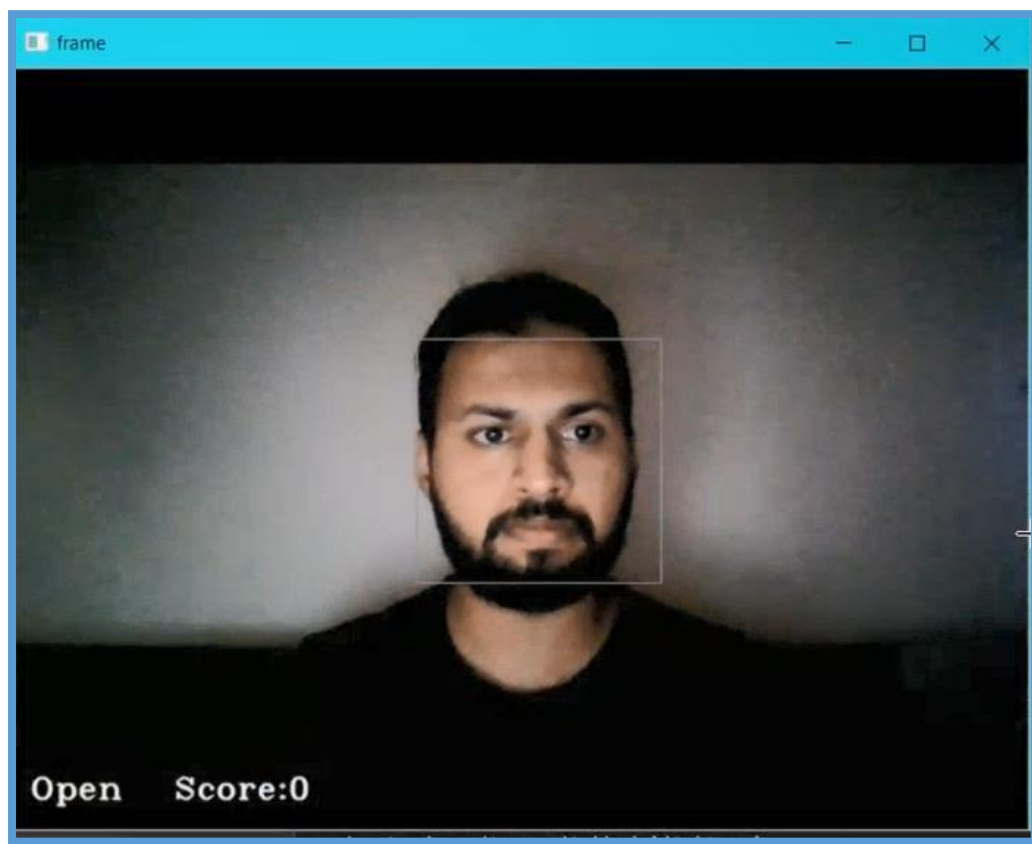
Figure 17: Haar Cascade files for face and eye detection

RESULTS AND VISUALIZATIONS

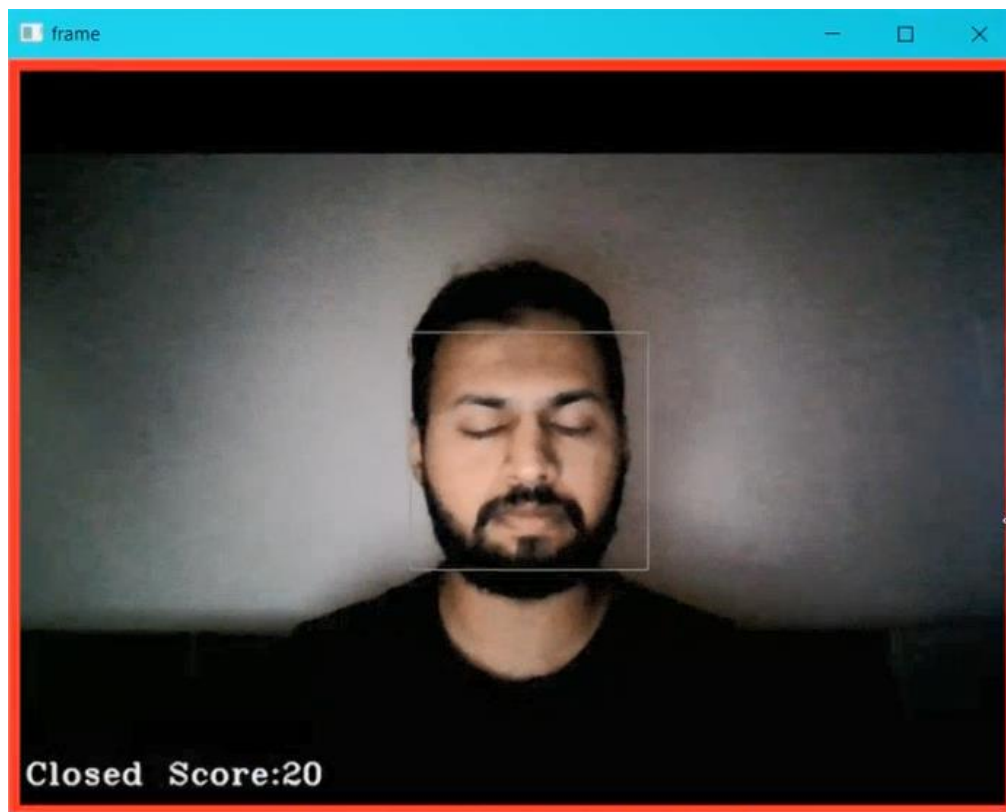
7.1 Test cases

Testing the model in real time using different states of eyes and in varying brightness levels.

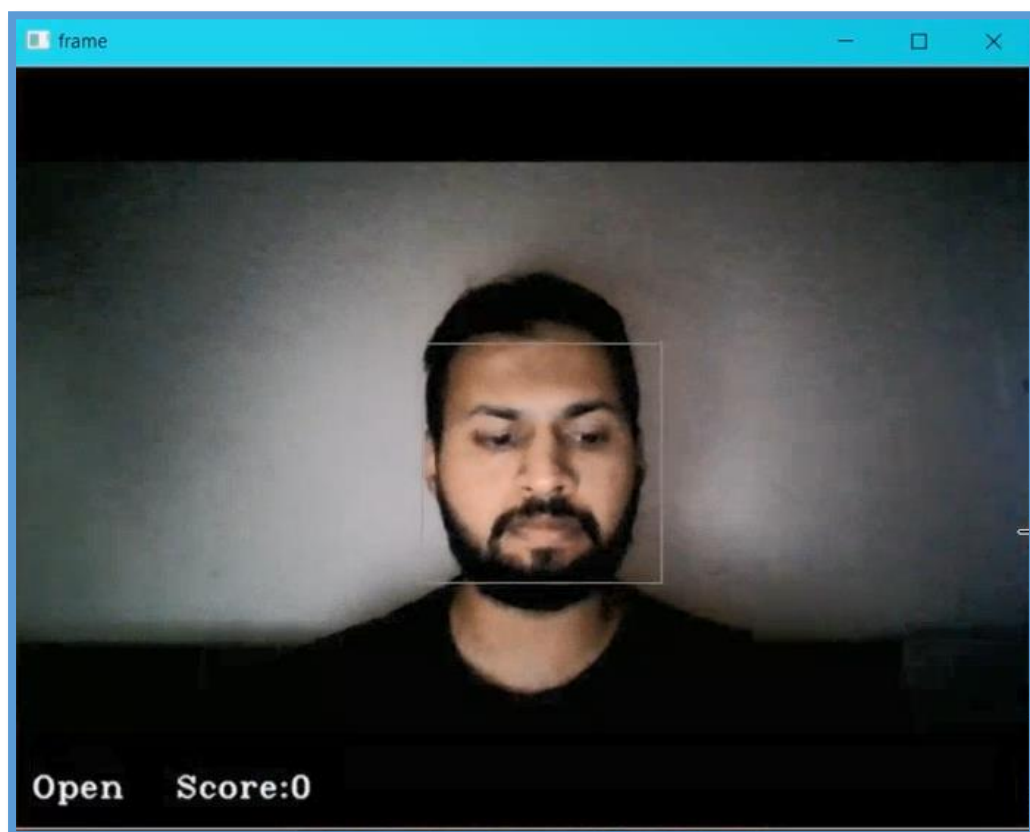
1. Fully opened eyes in high brightness.



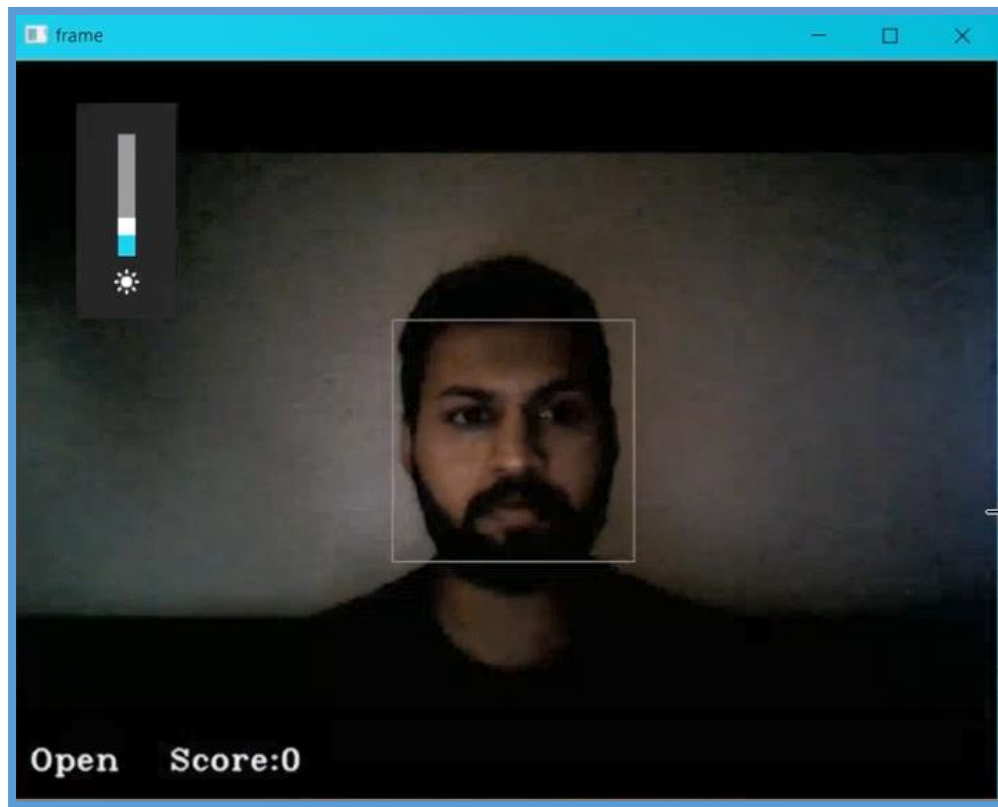
2. Closed eyes in high brightness.



3. Partially open eyes in high brightness.



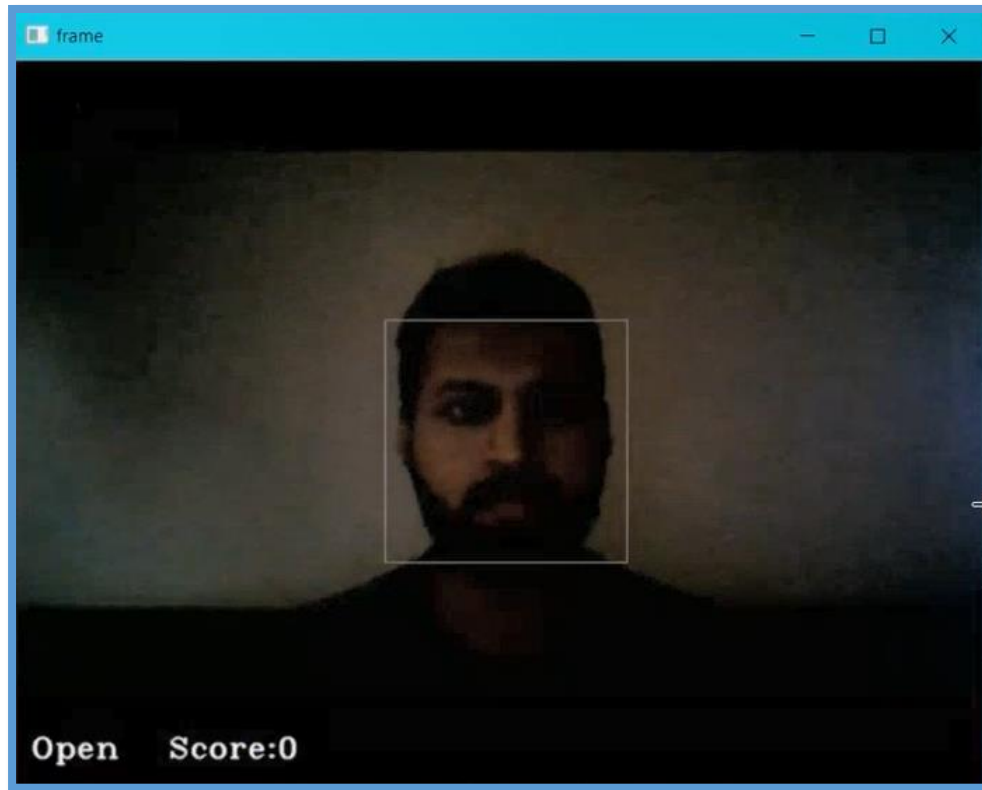
4. Fully open eyes in low brightness.



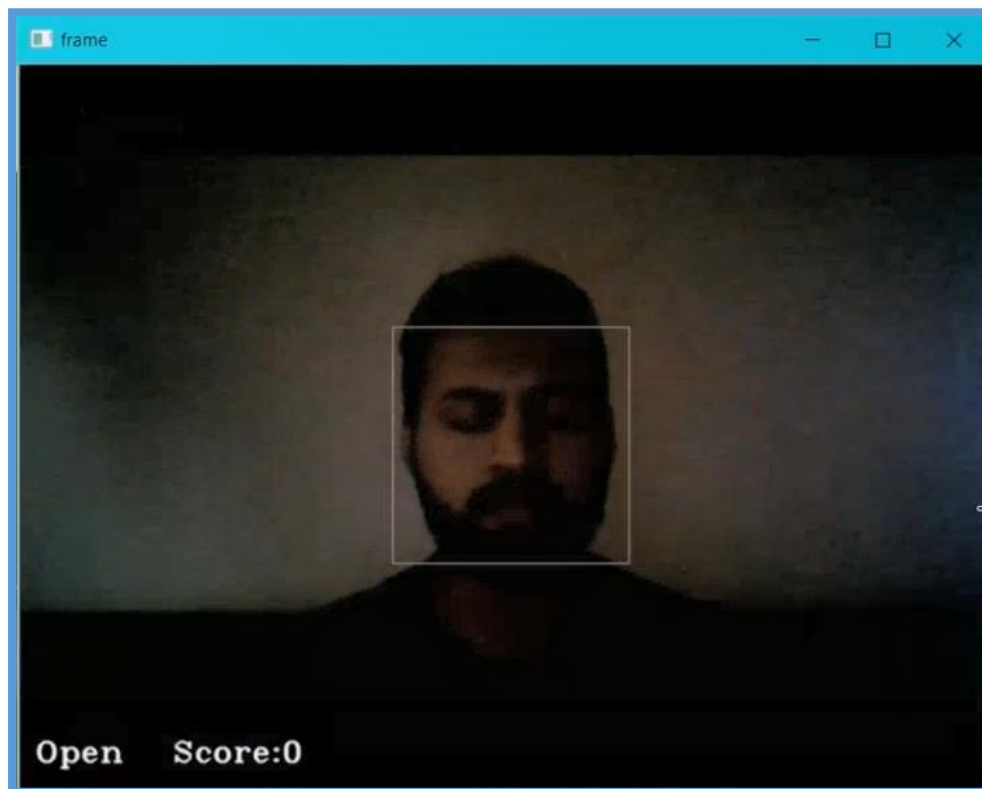
5. Closed eyes in low brightness.



6. Open eyes in very low brightness.



7. Closed eyes in very low brightness.



7.2 Results

We have successfully implemented the driver drowsiness detection system using OpenCV, Keras, Tensorflow and PyGame libraries in python. As observed from various test cases we can see that the system is able to detect faces and eyes of a person accurately in high as well as low brightness. When both eyes are detected closed, the score variable increments itself and after crossing threshold value 15, an alarm starts beeping and alerts the driver.

7.3 Challenges

As every project has its challenges, our project is also subject to the following constraints:

Dependence on ambient light: The model developed for this purpose strongly depends on the ambient light condition. As our algorithm considers the eye sight as a dark region when it is closed and brighter region when it is open so if the ambient condition affects such that there may be possibility of brighter and darker condition depending on light source then it causes error in the result. Also this model depends on certain minimum level of light condition otherwise it becomes very difficult to detect. To avoid this error we can use either LED light for better detection or we can use an infrared camera.

Distance of camera from driver face: For best result we have assumed and designed the code according to the fact that the distance between camera and face should be nearly 100 cm. Hence the designed set up output may vary from vehicle to vehicle as different vehicle have different types of seat lengths.

Use of spectacles: In case the user uses spectacle then it is difficult to detect the state of the eye. As it hugely depends on light hence reflection of spectacles may give the output for a closed eye as opened eye. Hence for this purpose the closeness of eye to the camera is required to avoid light.

Multiple face problem: If multiple face arise in the window then the camera may detect more number of faces undesired output may appear. Because of different condition of different faces. So we need to make sure that only the driver face come within the range of the camera. Also the speed of detection reduces because of operation on multiple faces.

CHAPTER 8

CONCLUSION AND FUTURE SCOPE

8.1 CONCLUSION

Implementation of drowsiness detection with OpenCV was done which includes the following steps:

- Successful runtime capturing of video with camera.
- Captured video was divided into frames and each frames were analyzed.
- Successful detection of face followed by detection of eye.
- If closure of eye for successive frames were detected then it is classified as drowsy condition else it is regarded as normal blink and the loop of capturing image and analyzing the state of driver is carried out again and again.
- In this implementation the system detects whether both eyes are closed or not. For every successful detection of closed eyes, the score value increases and after crossing the threshold value 15, an alarm is beeped to wake up the driver. The score value decreases when either of the eyes is open or both eyes are open.

8.2 FUTURE SCOPE

Our model is designed for detection of drowsy state of eye and give and alert signal or warning may be in the form of audio or any other means. But the response of driver after being warned may not be sufficient enough to stop causing the accident meaning that if the driver is slow in responding towards the warning signal then accident may occur. Hence to avoid this we can design and fit a motor driven system and synchronize it with the warning signal so that the vehicle will slow down after getting the warning signal automatically. Further, a system that uses other measurements such as heart rate variability, brain waves detection, muscle fatigue etc. of a person to detect drowsiness can be used to build a more efficient and accurate system.

REFERENCES

- [1]. K. C. Yowand, R. Cipolla, "Feature-based human face detection", "Image Vision Computation", *vol.15, no.9, pp.713–735, 1997.*
- [2] Peters R.D., Wagner E., Alicandri E., Fox J.E., Thomas M.L., Thorne D.R., Sing H.C., Balwinski S.M, "Effects of partial and total sleep deprivation on driving performance", *1999.*
- [3]. M. H. Yang, D. J. Kriegman, and N. Ahuja, "Detecting faces in images: *A survey*," *IEEE Trans. Pattern Anal. Mach. Intell.*, *vol.24, no.1, , Jan. 2002, pp. 34–58.*
- [4]. W. Zhao, R. Chellappa, P.J. Phillips, and A. Rosenfeld, "Face Recognition: A Literature Survey," *ACM Computing Surveys*, *vol. 35, pp. 399-459, 2003.*
- [5] Nan-Ning Zheng, Shuming Tang, Hong Cheng and Qing Li, Guanpi Lai and Fei-Yue Wang, "Toward Intelligent Driver-Assistance and Safety Warning Systems", *Intelligent Transportation System, IEEE 2004.*
- [6] Otmani S, Pebayle T, Roge J, Muzet A, "Effect of driving duration and partial sleep deprivation on subsequent alertness and performance of car drivers", *2005.*
- [7] Ole Helvig Jensen, "Implementation of Viola Jones Algorithm", *University of Denmark, pp: 20-36, 2008.*
- [8] Christian Scharfenberger, Samarjit Chakraborty, John Zelek and David Clausi, "Anti-Trap Protection for an Intelligent Smart Car Door System", *15th International IEEE Conference on Intelligent Transportation System, Anchorage, Alaska, USA, September 16-19, 2012.*
- [9] Dr.Suryaprasad J, Sandesh D, Saraswathi V, "Real time drowsy driver detection using haarcascade samples", *2013.*
- [10] Yi-Quin Wang, " An Analysis of Viola Jones algorithm for face detection", *University of Malaysia Phang, pp: 15-20, 2014.*