A PROJECT REPORT

ON

# "DeDrowse Application"

Submitted in partial fulfilment of the requirements

for the degree of

BACHELOR OF ENGINEERING IN COMPUTER ENGINEERING

By

| | |
|---|---|
| **Pooja G Bhagat** | **TE-A-104** |
| **Priyanka D Korde** | **TE-A-118** |
| **Raghuwardayal A Maurya** | **TE-A-123** |
| **Rohit J Mishra** | **TE-A-125** |

**Guide**

**Prof. Rajendra D Gawali**

**Department of Computer Engineering**

**Lokmanya Tilak College of Engineering**

**Koparkhairane, Navi Mumbai - 400 709**

**University of Mumbai**

**(AY 2021-22)**

# CERTIFICATE

This is to certify that the Mini Project entitled **“ DeDrowse Application”** is a bonafide work of **Pooja Bhagat(104), Priyanka Korde(118), Raghuwardayal Maurya(123), Rohit Mishra (125)** submitted to the University of Mumbai in partial fulfilment of the requirement for the award of the degree of **“Bachelor of Engineering”** in **“Computer Engineering” .**

**Prof. Rajendra D. Gawali**
Guide

**Prof. Rajendra D. Gawali**
Head of Department

**Dr. Vivek K. Sunnapwar**
Principal

# Mini Project Approval

This Mini Project entitled “**DeDrowse Application”** by **Pooja Bhagat(104), Priyanka Korde(118), Raghuwardayal Maurya(123), Rohit Mishra (125)** is approved for the degree of **Bachelor of Engineering** in **Computer Engineering.**

**Examiners**

**1 ..............................................**

(Internal Examiner Name & Sign)

**2 ..............................................**

(External Examiner name & Sign)

Date:

Place: Koparkhairane Navi Mumbai

# Contents

**Abstract** **ii**

**Acknowledgments** **iii**

**List of Abbreviations** **iv**

**List of Figures** **v**

**Introduction** **1**

**1.1** Introduction

**1.2** Motivation

**1.3** Problem Statement & Objectives

**Literature Survey** **4**

**2.1** Survey of Existing System

**Proposed System** **7**

**3.1** Introduction

**3.2** Architecture

**3.3** Details of Hardware & Software

**3.4** Process Design

**3.5** Results

**3.4** Conclusion and Future work.

**Reference** **23**

## ABSTRACT

In this project, we are planning to build **Drowsiness Detection System**. This system will monitor the eyes using a camera and by developing a model we can detect symptoms of drowsiness person sleeping. So, this project will help detect a person drowsiness in advance and will give alarming beeps.

# ACKNOWLEDGEMENT

We remain immensely obliged to **Prof. Rajendra Gawali** for providing us with the idea of this topic, and for his invaluable support in gathering resources for us either by way of information via internet/books also his guidance and supervision which made this project successful.

We would like to thank our Principal Dr. Vivek Sunnapwar, Vice Principal Dr. S.K. Shinde, Head of the Computer Engineering Department Prof. Rajendra Gawali and mini project coordinators Prof. Chaitrali Chaudhari & Prof. Shirin Matwankar.

We are also thankful to faculty and staff of Computer Engineering Department and Lokmanya Tilak College of Engineering, Navi Mumbai for their invaluable support.

We would Like to say that It has indeed been a fulfilling experience for working out this project.

## LIST OF ABBREVIATIONS

| **Short Form** | **Full Form** |
|---|---|
| DDS | Drowsiness Detection |
| CNN | Convolutional Neural Networks |
| RNN | Recurrent Neural Network |
| TF | Tensor Flow |
| ANNs | Artificial Neural Networks |
| SNNs | Simulated Neural Networks |

## LIST OF FIGURES

| Figure No. | Title |
|---|---|
| 1 | Neural Network |
| 2 | Deep Neural Network |
| 3 | Convolutional Neural Network |
| 4 | Recurrent Neural Network |
| 5 | Inception V3 Model |
| 6 | Components the Inception V3 model |
| 7 | Five haar like features |
| 8 | Example of Five haar like features |
| 9 | Transfer Learning |
| 10 | Image of annotations of the proposed dataset |
| 11 | Architecture of DDS |
| 12 | Eyes Open (Normal Condition) |
| 13 | Eyes Closed (Normal Condition) |
| 14 | Eyes Open (Dark Condition) |
| 15 | Eyes Closed (Dark Condition) |

# CHAPTER 1
# INTRODUCTION

## 1.1 Introduction

DeDrowse is the real time Drowsiness behaviours which are related to fatigue are in the form of eye closing, head nodding or the brain activity. Drowsiness is a state of abnormal sleeping, people suffering from drowsiness tend to sleep at inappropriate situations. Hence, we can either measure change in physiological signals, such as brain waves, heart rate and eye blinking to monitor drowsiness or consider physical changes such as open/closed state of eyes.

In addition, long time working would result in perspiration on the sensors, diminishing their ability to monitor accurately. The technique is to measure physical changes (i.e. open/closed eyes to detect fatigue) is well suited for real world conditions and detect the changes by using a video camera. In addition, micro sleeps that are short period of sleeps lasting 2 to 3 minutes are good indicators of fatigue. Thus, by continuously monitoring the eyes of the person one can detect the sleepy state of person and a timely warning is issued in the form of alert sound.

Drowsiness detection techniques, in accordance with the parameters used for detection is divided into two sections i.e. intrusive method and a non-intrusive method. The main difference of these two methods is that the intrusive method. An instrument connected to the person and then the value of the instrument is recorded and checked. But intrusive approach has high accuracy, which is proportional to person' discomfort, so this method is rarely used.

## 1.2 Motivation

The main motive of this project is to prevent a person from entering a state of sleepiness by alerting them or by shutting down the system, saving the system resources and allowing the person to rest. This system can be used on various platforms.

During the Covid-19, many changes came to our world and it took some time for everyone to adopt the new normal. The Covid-19 impact was everywhere, which resulted in the closure of Schools and other educational institutions. Taking online lectures and digitally shifting students became more prone to play online games which impacted their mental health. During the lectures or while playing the games,etc they may feel drowsy. So this application may be useful in this situation. If he/ she is drowsy then it will detect and alert them by beep sound. Parents can shut down the system, save the system resources and allow the student to take rest keeping their mental health into consideration.

This has been a great reason which motivated us to work on this project and build an application for the well-being of society.

## 1.3 Problem Statement and Objectives

### Problem Statement

To build a DeDrowse ( Detection Application ) that will monitor the eyes using a camera and by developing a model, we can detect symptoms of drowsiness in a person and alert them via alert messages on-screen.

### Objectives

Drowsiness  is a process where the level of consciousness decreases due to the lack of sleep or fatigue and can cause person falls asleep. The Objective of this application is to develop a system i.e. accurate to detect a person’s drowsiness based on eyelid movement and is reliable to give appropriate voice alert in real time. To differentiate between normal eye blink and drowsiness. To implement deep learning, Convolution Neural Network, and Transfer Learning using Python and to take an input image of a person and provide a model to predict that person's level of drowsiness.

# CHAPTER 2
# LITERATURE  SURVEY

## 1.1 Survey of Existing System

**1. Image Processing For Face Recognition Using HAAR**
**-Maria Dominic Savio, T Deepa1, Anudeep Bonasu , and Talluru Sai Anurag.**

Face recognition is one of the most active areas of research from the past two decades. Attempts are being made to understand how a human recognizes another human face. It is widely accepted that facial recognition can be based on structural information and non- structural / spatial details. In the present study, he is applying differential observations using Eigen / docking characteristics of many built-in facial features and artificial neural networks. The proposed method aims to obtain a facial feature by reducing facial features such as eyes, nose, mouth, and face depending on the importance of facial features. The face recognition system developed in this paper will inform the human face and assess the current percentage of accuracy. Therefore, this work is for human facial recognition and includes a percentage of facial expressions. The implementation of this function also offers many applications such as photography, biometric in bank Lockers, etc.

**2. Driver Drowsiness Detection System in Automotive Vehicles**
**-Department of ISE OF VVIET (IJERT/2017)**

Road accidents are usually caused by driver carelessness. The major carelessness exhibited by the driver are drunken behavior and negligence. The driver drowsiness detection system in automotive vehicles focuses on abnormal behavior exhibited by the driver using a microcontroller, the Raspberry pi single board computer. In the proposed system a non-intrusive driver drowsiness monitoring system has been developed using computer vision techniques. Irrespective of the driver wearing spectacles and darkness level inside the vehicle, the system is able to detect the drowsiness. The system will detect drowsiness within the time duration of about two to three seconds. The driver is alerted through alarms in real time.

**3. Facial features monitoring for real time drowsiness detection**
**-B. N. Manu (IEEE/2016)**

This paper describes an efficient method for drowsiness detection by three well defined phases. These three phases are facial features detection using Viola Jones, the eye tracking and yawning detection. Once the face is detected, the system is made illumination invariant by segmenting the skin part alone and considering only the chromatic components to reject most of the non face image backgrounds based on skin color. The tracking of eyes and yawning detection are done by correlation coefficient template matching. The feature vectors from each of the above phases are concatenated and a binary linear support vector machine classifier is used to classify the consecutive frames into fatigue and non fatigue states and sound an alarm for the former, if it is above the threshold time. Extensive real time experiments prove that the proposed method is highly efficient in finding the drowsiness and alerting the driver.

4. **Smart Alarm System (Drowsiness Detection System Review Paper)**
   **-Chiranjit Das, Chirag Bhatt, U. Belim, Taha Bhatia, Jinil patel (Parul University)**

With the increase in the automotive population, accident rates are rising rapidly, one of the major reasons is the state of drowsiness or fatigue. Such fatal incidents can be prevented if the driver is warned in time. With time, several drowsiness alarm systems are been implemented, the system monitors driver’s movements through various techniques, one of the most considerable one is face detection, Open Computer Vision is widely used to detect driver’s movements for a long period of time. A comparative study on different types of approach is summarized in this paper. The main agenda is to help the society to understand more about the techniques and find the most convenient method for them.

# CHAPTER 3
# PROPOSED SYSTEM

## 3.1. Introduction

Drowsiness is a state of abnormal sleeping, people suffering from drowsiness tend to sleep in inappropriate situations.
In this project, we have developed Drowsiness Detection System. This system will monitor the eyes using a camera and a model will predict the eye state of the user according to symptoms of drowsiness person sleeping then beeps alerts will start and if the user open the eyes the beeps will stop automatically and if the user continues to sleep then beeps increases and does not stops and alert messages are also show on the screen untill user awakes and switch it off.

## 3.2. Architecture

**What is Neural Network?**

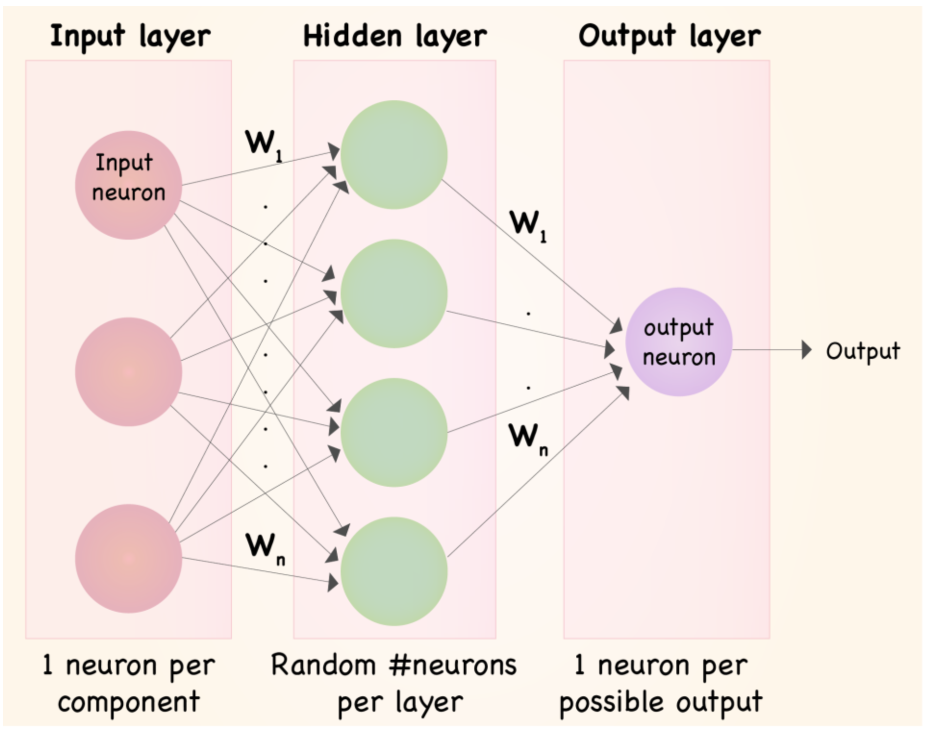


*Fig 1: Neural Network*

Neural networks reflect the behavior of the human brain, allowing computer programs to recognize patterns and solve common problems in the fields of AI, machine learning, and deep learning.

Neural networks, also known as artificial neural networks (ANNs) or simulated neural networks (SNNs), are a subset of machine learning and are at the heart of deep learning algorithms. Their name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another.

Neural networks rely on training data to learn and improve their accuracy over time. However, once these learning algorithms are fine-tuned for accuracy, they are powerful tools in computer science and artificial intelligence, allowing us to classify and cluster data at a high velocity. Tasks image recognition can take minutes versus hours when compared to the manual identification by human experts.

**What is Deep Neural Network?**

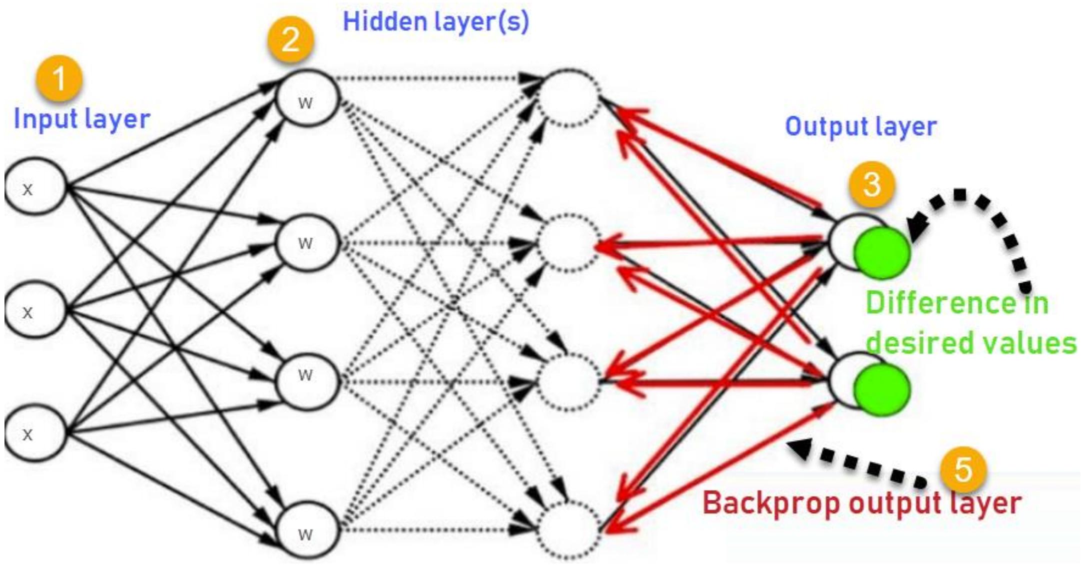


*Fig 2: Deep Neural Network*

Deep neural networks consist of multiple layers of interconnected nodes, each building upon the previous layer to refine and optimize the prediction or categorization. This progression of computations through the network is called forward propagation. The input and output layers of a deep neural network are called *visible* layers. The input layer is where the deep learning model ingests the data for processing, and the output layer is where the final prediction or classification is made.

Another process called backpropagation uses algorithms, like gradient descent, to calculate errors in predictions and then adjusts the weights and biases of the function by moving backwards through the layers in an effort to train the model. Together, forward propagation and backpropagation allow a neural network to make predictions and correct for any errors accordingly. Over time, the algorithm becomes gradually more accurate.

**Convolutional Neural Network Design:**

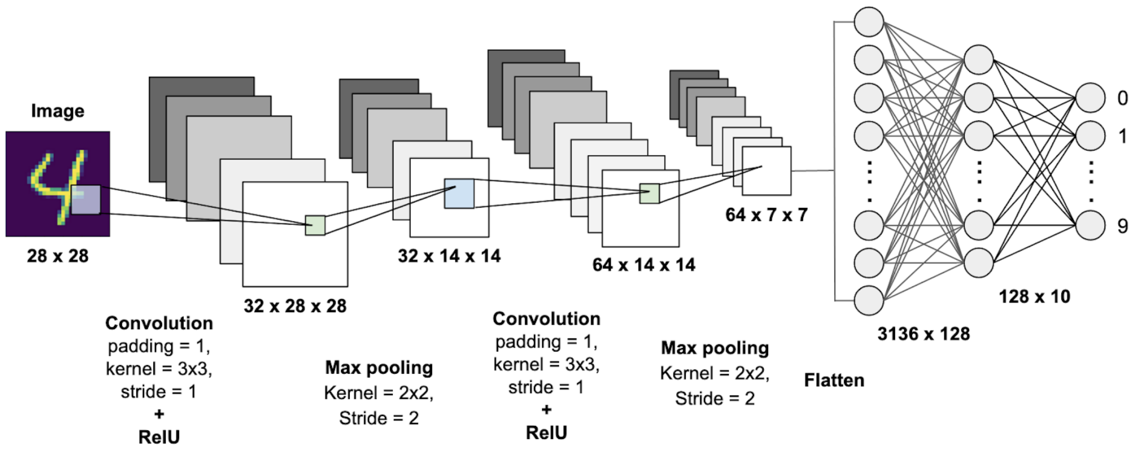


*Fig 3: Convolutional Neural Network*

The architecture of a convolutional neural network is a multi-layered feed-forward neural network, made by stacking many hidden layers on top of each other in sequence. It is this sequential design that allows convolutional neural networks to learn hierarchical features.

The hidden layers are typically convolutional layers followed by activation layers, some of them followed by pooling layers.

A simple convolutional neural network that aids understanding of the core design principles is the early convolutional neural network LeNet-5, published by Yann LeCun in 1998. LeNet is capable of recognizing handwritten characters.

**Recurrent Neural Network:**

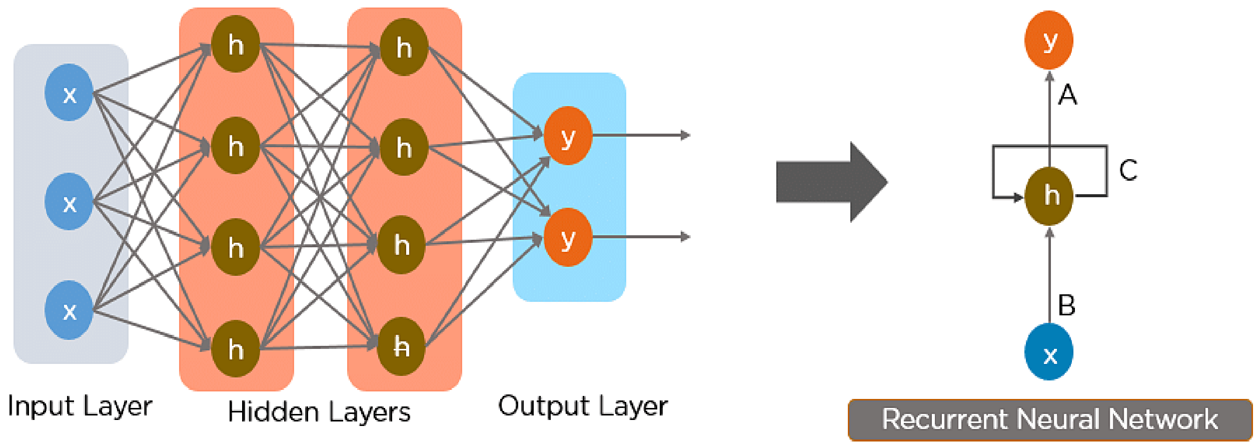


*Fig 4: Recurrent Neural Network*

RNN works on the principle of saving the output of a particular layer and feeding this back to the input in order to predict the output of the layer.

**Inception V3 Model Architecture**

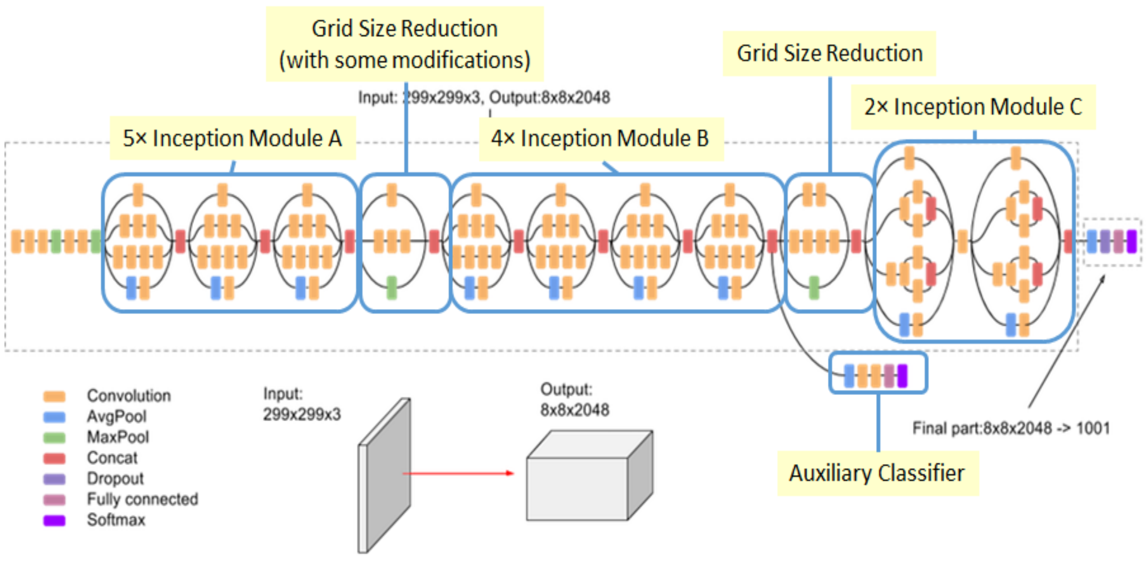
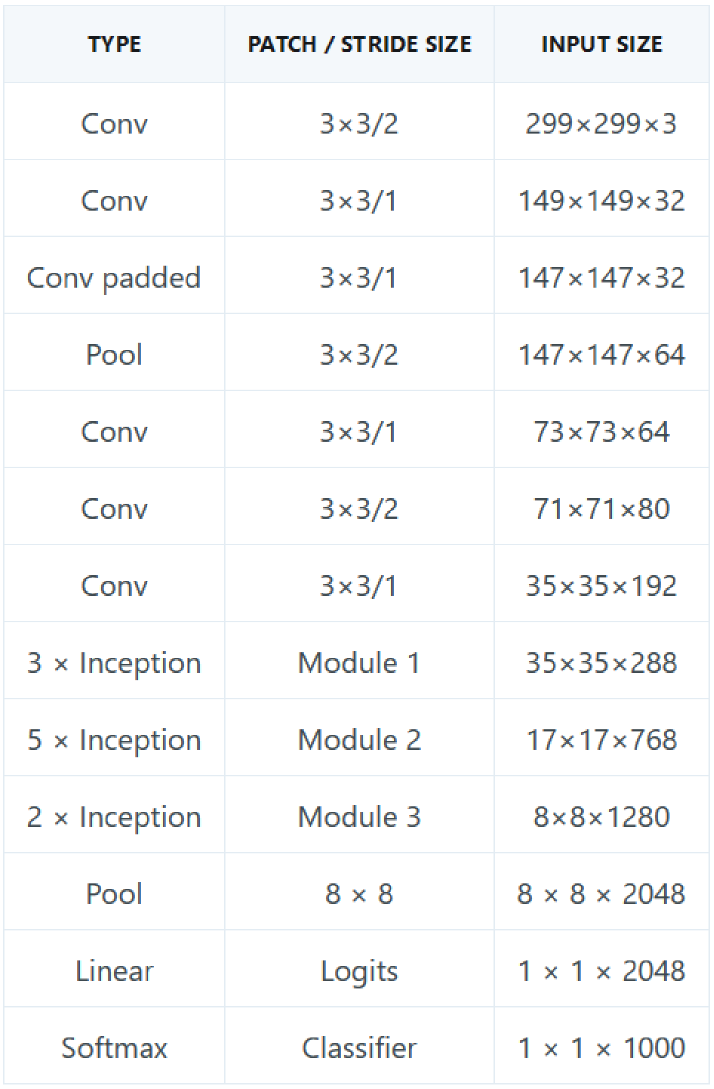


*Fig 5: Inception V3 Model*

The inception V3 model is made up of 42 layers which is a bit higher than the previous inception V1 and V2 models. But the efficiency of this model is really impressive

In comparison to VGGNet, Inception Networks (GoogLeNet/Inception v1) have proved to be more computationally efficient, both in terms of the number of parameters generated by the network and the economical cost incurred (memory and other resources). The techniques include factorized convolutions, regularization, dimension reduction, and parallelized computations. What are the components the Inception V3 model is made of:



| TYPE | PATCH / STRIDE SIZE | INPUT SIZE |
|---|---|---|
| Conv | 3×3/2 | 299×299×3 |
| Conv | 3×3/1 | 149×149×32 |
| Conv padded | 3×3/1 | 147×147×32 |
| Pool | 3×3/2 | 147×147×64 |
| Conv | 3×3/1 | 73×73×64 |
| Conv | 3×3/2 | 71×71×80 |
| Conv | 3×3/1 | 35×35×192 |
| 3 × Inception | Module 1 | 35×35×288 |
| 5 × Inception | Module 2 | 17×17×768 |
| 2 × Inception | Module 3 | 8×8×1280 |
| Pool | 8 × 8 | 8 × 8 × 2048 |
| Linear | Logits | 1 × 1 × 2048 |
| Softmax | Classifier | 1 × 1 × 1000 |

*Fig 6: Components of Inception V3 Model*

**Face Detection using Haar Classifier Algorithm:**

For the face Detection it uses Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

The image to be used is divided into different kinds of **sub-windows** and multiple **Haar-like features** to compute it at different scales and positions for each sub-window. The main features are selected using the **Adaboost algorithm**. Then each sub-window is checked for the **presence or absence of face** using a **cascade of classifiers**.

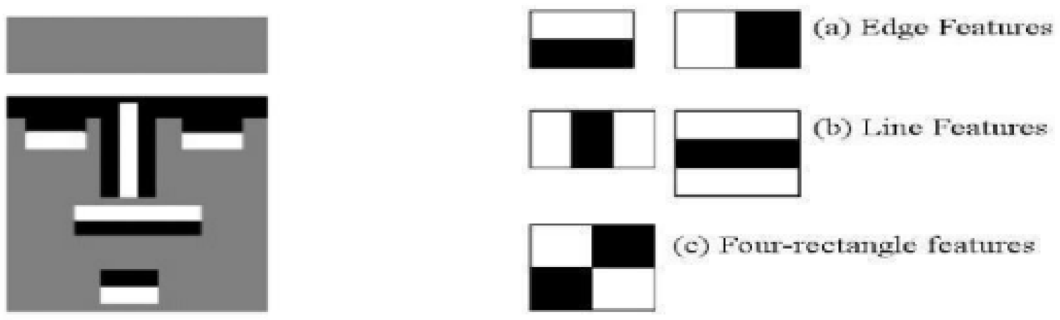


*Fig 7: Five haar like features*

Here we will work with face detection. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. For this, Haar features shown in the below image are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle. Example is shown in Figures.

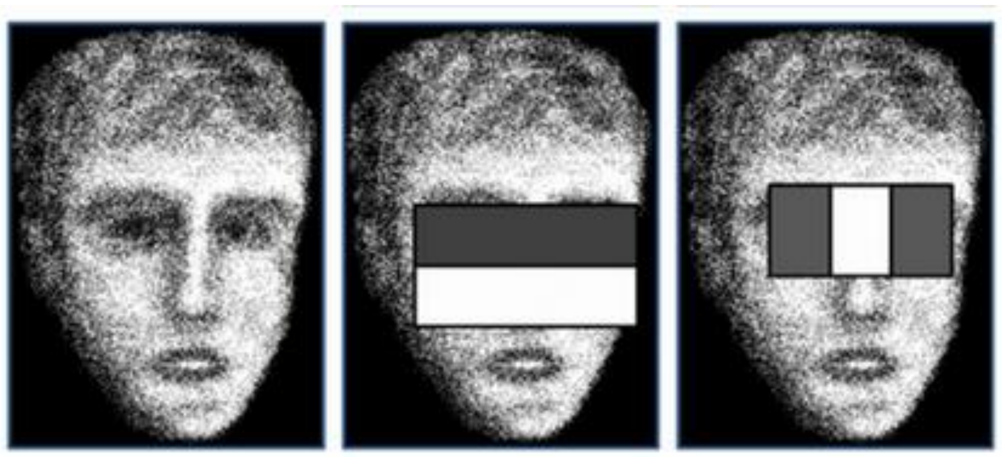


*Fig 8: Example of Five haar like features*

The detection algorithm uses a cascade of classifiers which use Haar-like features. Thus, it is also called the **Haar Cascades based detector**.

**features = sum(pixels in the black area) - sum(pixels in the white area)**

A cascaded Adaboost classifier with the Haar-like features is exploited to find out the face region. First, the compensated image is segmented into numbers of rectangle areas, at any position and scale within the original image. Due to the difference of facial feature, Haar-like feature is efficient for real-time face detection. These can be calculated according to the difference of sum of pixel values within rectangle areas. The features can be represented by the different composition of the black region and white region. A cascaded Adaboost classifier is a strong classifier which is a combination of several weak classifiers. Each weak classifier is trained by Adaboost algorithm. If a candidate sample passes through the cascaded Adaboost classifier, the face region can be found. Almost all of face samples can pass through and nonface samples can be rejected

In the figure shown above, when we calculate the second image features, it will give more features because the **bridge is lighter than the nearby area**. But if the same features, we keep on the head of the face, we will get fewer features. In the third image, we also get more features as it can detect the eye region since the eye region is darker as compared to the region below it.

It should be noted that only a single feature is not capable of detecting faces with high accuracy. But, when many such features vote for the presence or absence of a face, the detection becomes very accurate and robust.

These features have actual real importance in the context of face detection:

1. Eye regions tend to be darker than cheek regions.
2. The nose region has more bright pixels than the eye region.

Therefore from the above given five rectangles along with the corresponding difference of sums, we are able to get the features which can classify the face. To detect which features belong to face from the available number of features we use the AdaBoost algorithm to select which ones correspond to facial regions of an image

**What is Transfer Learning?**

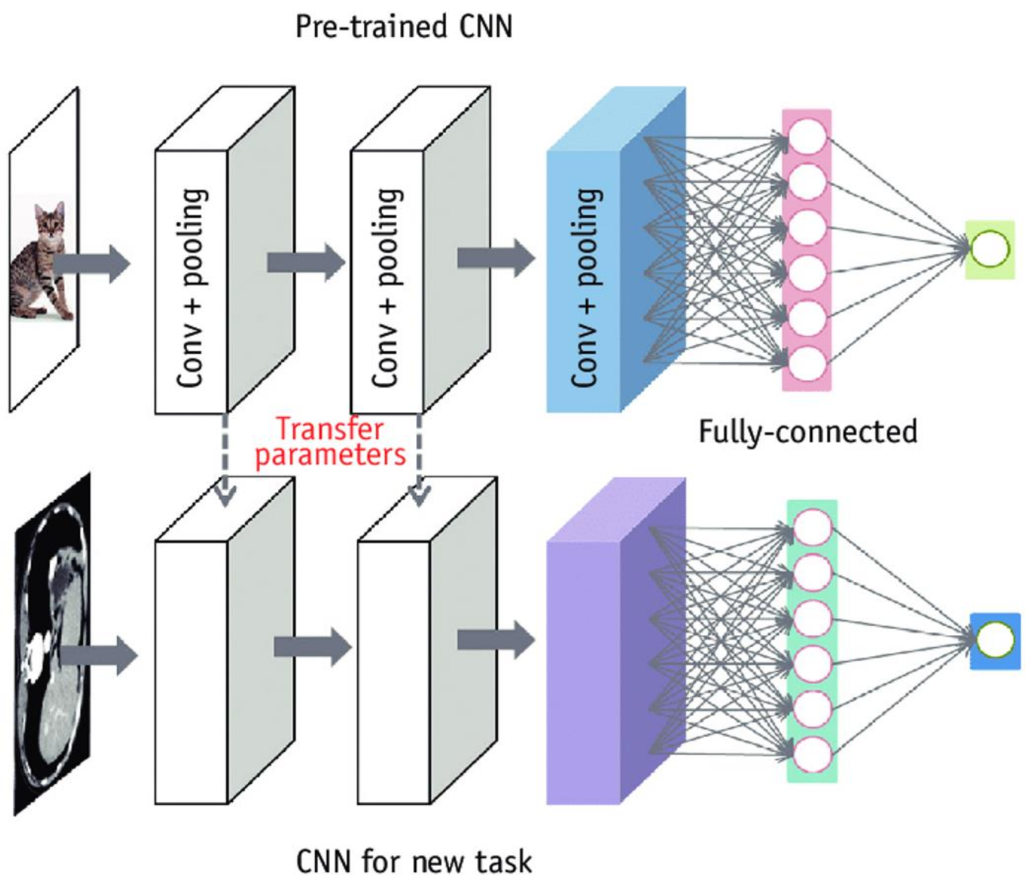


*Fig 9: Transfer Learning*

Transfer learning, used in machine learning, is the reuse of a pre-trained model on a new problem. In transfer learning, a machine exploits the knowledge gained from a previous task to improve generalization about another

In computer vision, for example, neural networks usually try to detect edges in the earlier layers, shapes in the middle layer and some task-specific features in the later layers.
In transfer learning, the early and middle layers are used and we only retrain the latter layers. It helps leverage the labelled data of the task it was initially trained on.

Transfer learning has several benefits, but the main advantages are saving training time, better performance of neural networks (in most cases), and not needing a lot of data. Usually, a lot of data is needed to train a neural network from scratch but access to that data isn't always available — this is where transfer learning comes in handy. With transfer learning a solid machine learning model can be built with comparatively little training data because the model is already pre-trained. This is especially valuable in natural language processing because mostly expert knowledge is required to create large labeled datasets. Additionally, training time is reduced because it can sometimes take days or even weeks to train a deep neural network from scratch on a complex task.

Dataset Introduction:
Dataset Link: http://mrl.cs.vsb.cz/eyedataset

Detailed explanation of dataset:
In the dataset, we annotated the following properties (the properties are indicated in the following order):
subject ID; has data of 37 different persons (33 men and 4 women)
image ID; the dataset consists of 84,898 images
gender [0 - man, 1 - woman]; the dataset contains the information about gender for each image (man, woman)
glasses [0 - no, 1 - yes]; the information if the eye image contains glasses is also provided for each image (with and without the glasses)
eye state [0 - closed, 1 - open]; this property contains the information about two eye states (open, close)
reflections [0 - none, 1 - small, 2 - big]; we annotated three reflection states based on the size of reflections (none, small, and big reflections)
lighting conditions [0 - bad, 1 - good]; each image has two states (bad, good) based on the amount of light during capturing the videos
sensor ID [01 - RealSense, 02 - IDS, 03 - Aptina]; at this moment, the dataset contains the images captured by three different sensors

example: s001_00123_0_0_0_0_0_01.png

An examples of image annotations of the proposed dataset:

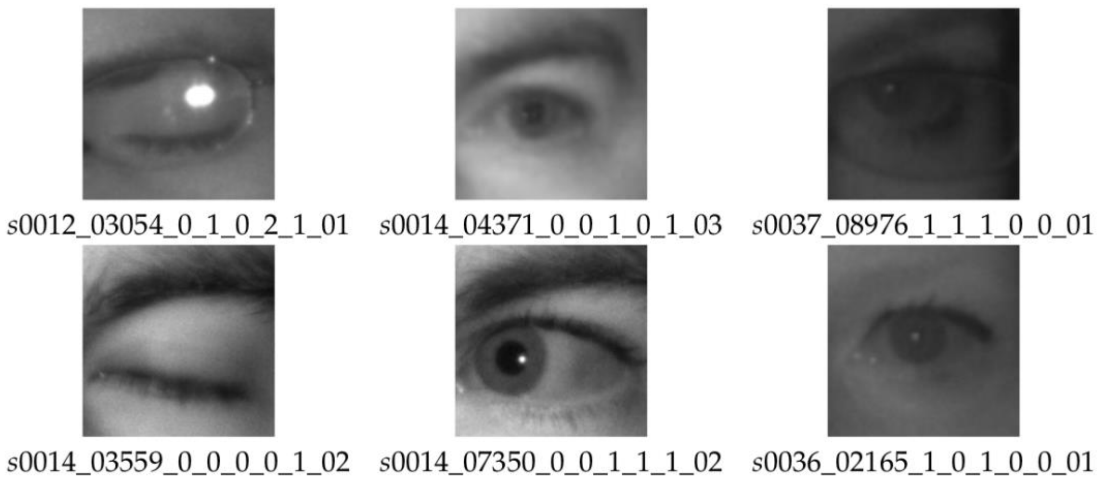


*Fig 10: Image of annotations of the proposed dataset*

## 3.3. Details of Hardware & Software

**Hardware**

- Web Camera
- NVIDIA GPU GTX 1050
- Windows OS

**Software**

- Tensorflow libraries
- NVIDIA Cuba libraries
- Google Colab
- Jupyter Notebook
- Anaconda Navigator
- Python Intepreter

### 3.4. Process Design

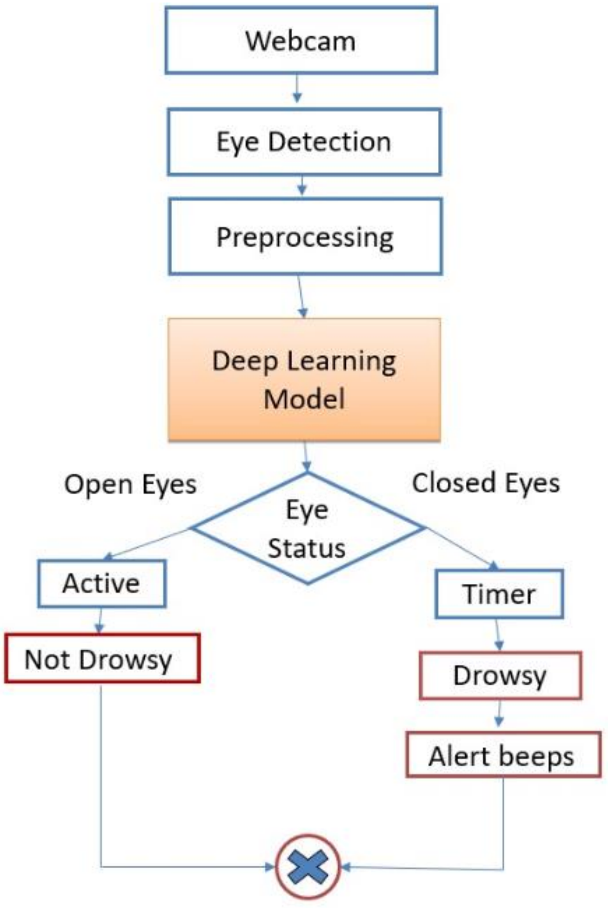


*Fig 11: Architecture of DDS*

Step 1: Taking Real Time Image or Video from the Web Camera

Step 2: Haar Classifier from Open-CV will detect Eye region and provide that Images for further image processing

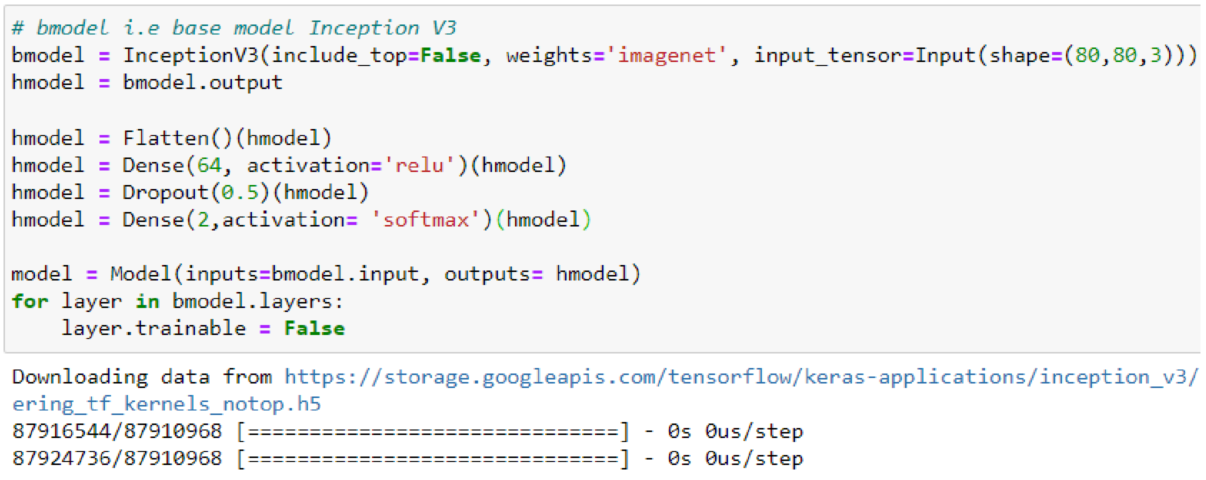
Step 3: Image will be converted from 3 Channels rgb index to 2 Channels Gray Scale and feed that image to the Deep Learning Model

Step 4: Deep Learning Model is Model created using Transfer Learning the algorithm used is Inception V3 and last layers are removed and convolutional layers are added at the end and softmax to predict wheather the eyes are drowsy or not and according to eye status.

Step 5: Prediction calculated will be check for open eye and closed eye threshold conditions according to that score will be increased and decreased and score will be printed on the screen

Step 6: If eye status is Drowsy then Alert Beeps will start and flask message of user is drowsy will be printed on the screen and beeps won't stop until user awakes and if user is not drowsy then alert beeps will stop.

## Model Preparation:



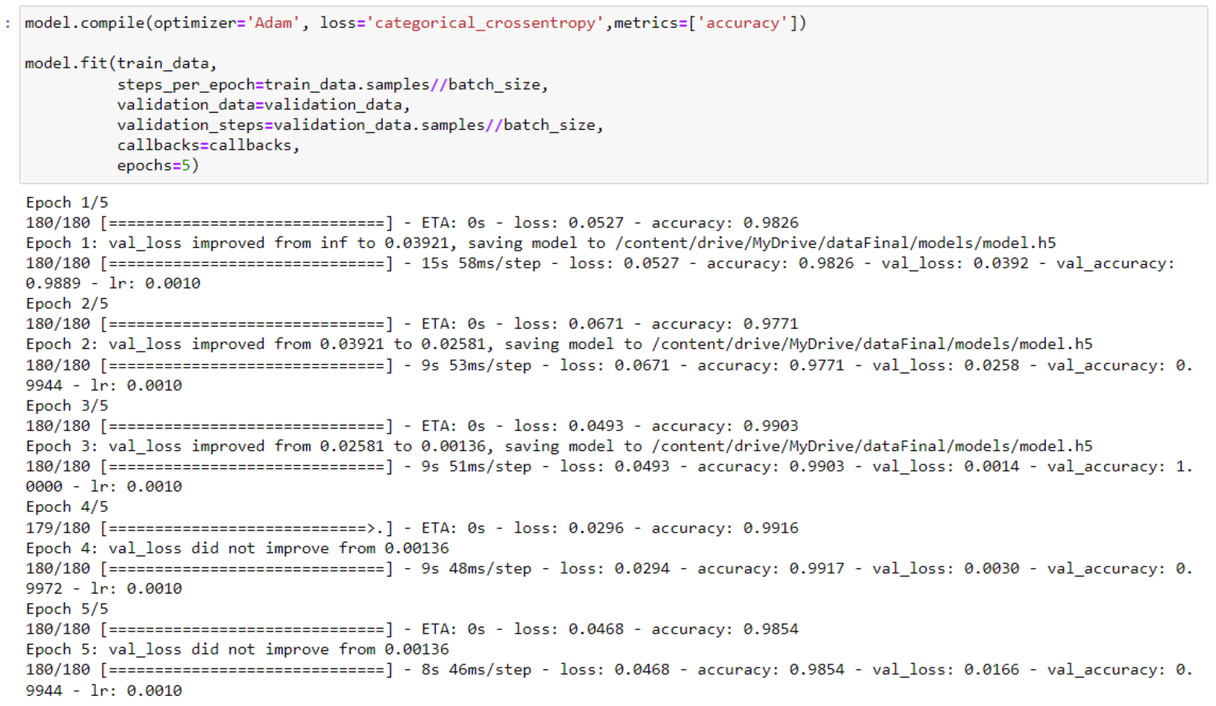
```
# bmodel i.e base model Inception V3
bmodel = InceptionV3(include_top=False, weights='imagenet', input_tensor=Input(shape=(80,80,3)))
hmodel = bmodel.output

hmodel = Flatten()(hmodel)
hmodel = Dense(64, activation='relu')(hmodel)
hmodel = Dropout(0.5)(hmodel)
hmodel = Dense(2,activation= 'softmax')(hmodel)

model = Model(inputs=bmodel.input, outputs= hmodel)
for layer in bmodel.layers:
    layer.trainable = False
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/inception_v3/
ering_tf_kernels_notop.h5
87916544/87910968 [==============================] - 0s 0us/step
87924736/87910968 [==============================] - 0s 0us/step
```

## Model Training:

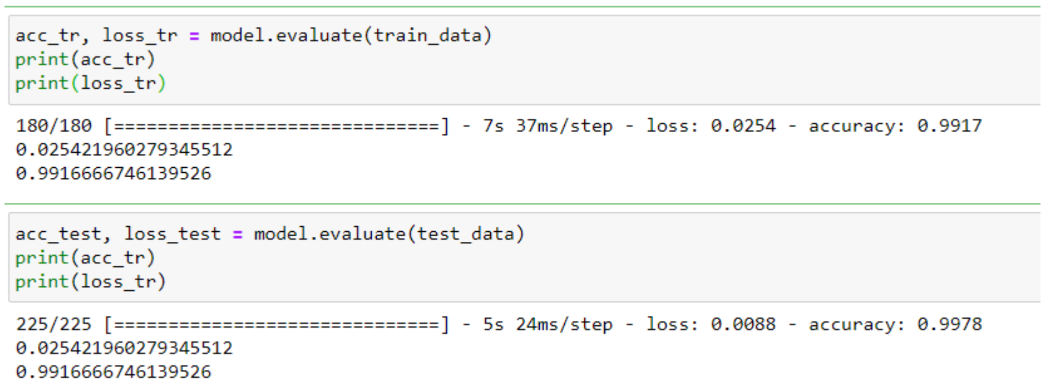


```
model.compile(optimizer='Adam', loss='categorical_crossentropy',metrics=['accuracy'])

model.fit(train_data,
          steps_per_epoch=train_data.samples//batch_size,
          validation_data=validation_data,
          validation_steps=validation_data.samples//batch_size,
          callbacks=callbacks,
          epochs=5)
```

```
Epoch 1/5
180/180 [==============================] - ETA: 0s - loss: 0.0527 - accuracy: 0.9826
Epoch 1: val_loss improved from inf to 0.03921, saving model to /content/drive/MyDrive/dataFinal/models/model.h5
180/180 [==============================] - 15s 58ms/step - loss: 0.0527 - accuracy: 0.9826 - val_loss: 0.0392 - val_accuracy:
0.9889 - lr: 0.0010
Epoch 2/5
180/180 [==============================] - ETA: 0s - loss: 0.0671 - accuracy: 0.9771
Epoch 2: val_loss improved from 0.03921 to 0.02581, saving model to /content/drive/MyDrive/dataFinal/models/model.h5
180/180 [==============================] - 9s 53ms/step - loss: 0.0671 - accuracy: 0.9771 - val_loss: 0.0258 - val_accuracy: 0.
9944 - lr: 0.0010
Epoch 3/5
180/180 [==============================] - ETA: 0s - loss: 0.0493 - accuracy: 0.9903
Epoch 3: val_loss improved from 0.02581 to 0.00136, saving model to /content/drive/MyDrive/dataFinal/models/model.h5
180/180 [==============================] - 9s 51ms/step - loss: 0.0493 - accuracy: 0.9903 - val_loss: 0.0014 - val_accuracy: 1.
0000 - lr: 0.0010
Epoch 4/5
179/180 [============================>.] - ETA: 0s - loss: 0.0296 - accuracy: 0.9916
Epoch 4: val_loss did not improve from 0.00136
180/180 [==============================] - 9s 48ms/step - loss: 0.0294 - accuracy: 0.9917 - val_loss: 0.0030 - val_accuracy: 0.
9972 - lr: 0.0010
Epoch 5/5
180/180 [==============================] - ETA: 0s - loss: 0.0468 - accuracy: 0.9854
Epoch 5: val_loss did not improve from 0.00136
180/180 [==============================] - 8s 46ms/step - loss: 0.0468 - accuracy: 0.9854 - val_loss: 0.0166 - val_accuracy: 0.
9944 - lr: 0.0010
```

## Model Evalution:



```
acc_tr, loss_tr = model.evaluate(train_data)
print(acc_tr)
print(loss_tr)
```

```
180/180 [==============================] - 7s 37ms/step - loss: 0.0254 - accuracy: 0.9917
0.025421960279345512
0.9916666746139526
```

```
acc_test, loss_test = model.evaluate(test_data)
print(acc_tr)
print(loss_tr)
```

```
225/225 [==============================] - 5s 24ms/step - loss: 0.0088 - accuracy: 0.9978
0.025421960279345512
0.9916666746139526
```

## 3.5. Result

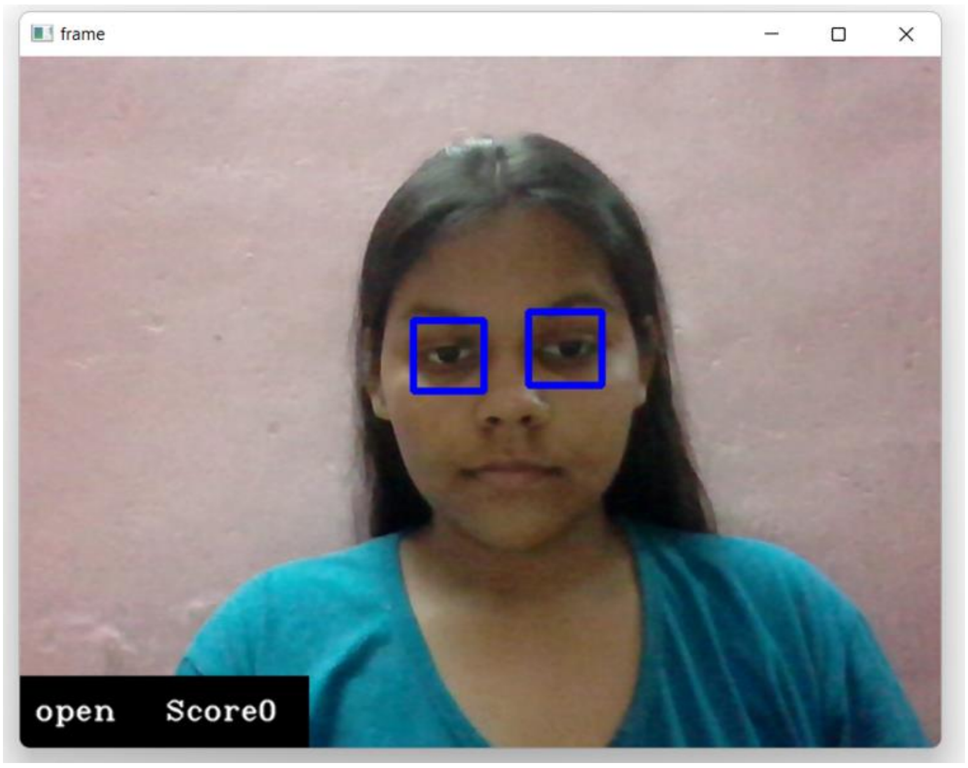


*Fig 12: Eyes Open*

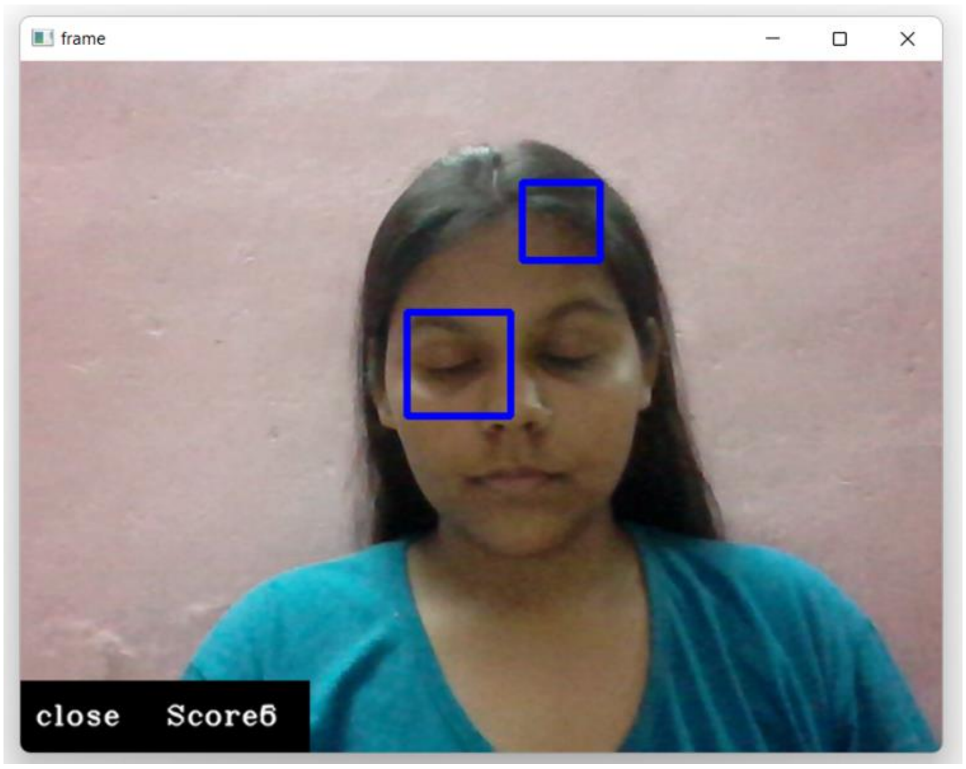


*Fig 13: Eyes Closed*

Eyes Closed so alarm with Beeps to awake the person

Predictions in dark condition

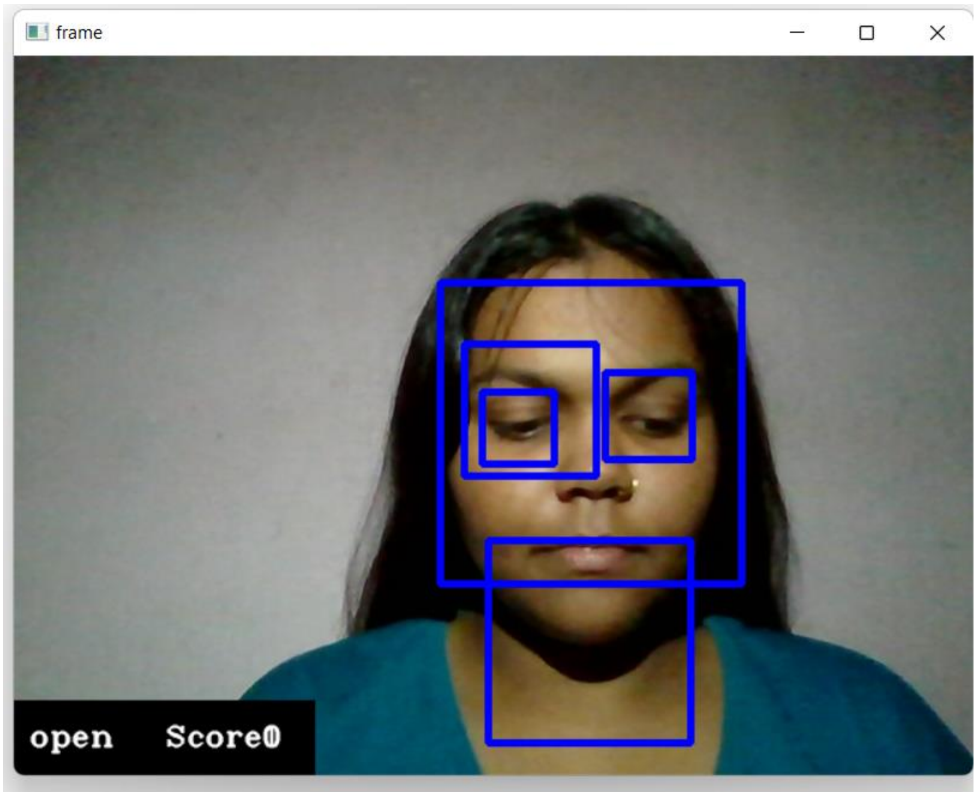


*Fig 14: Eyes Open (Dark Condition)*

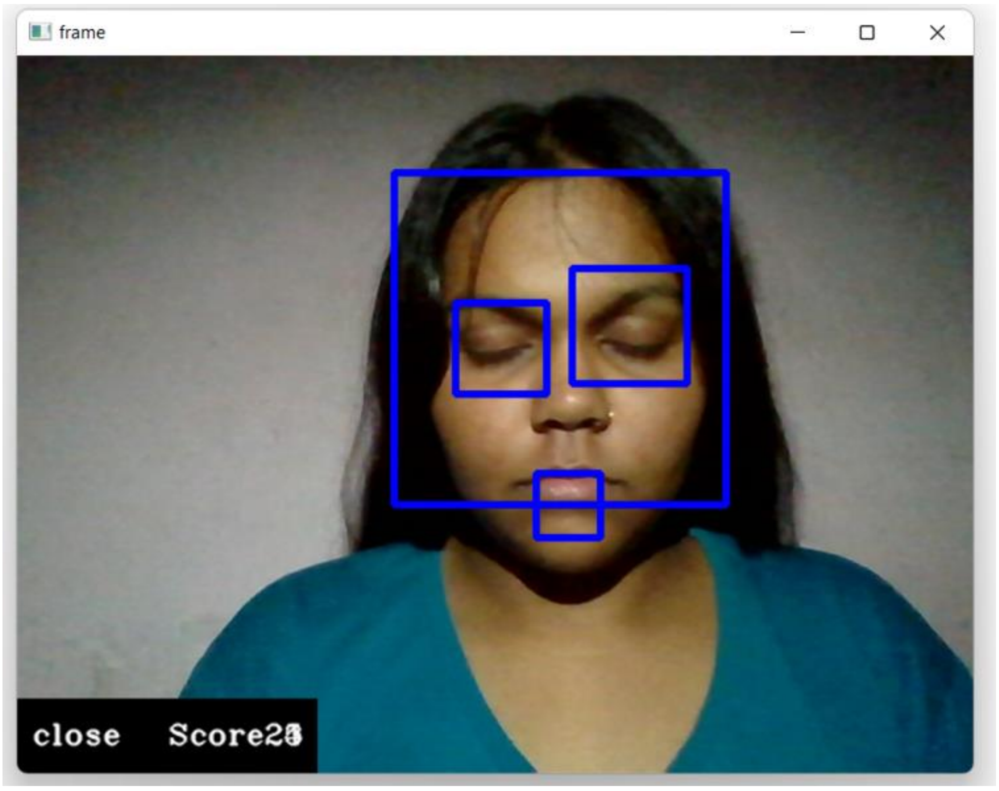


*Fig 15: Eyes Closed (Dark Condition)*

## 3.6. Conclusion

This project enhance health lifestyle energy efficient and can be used anywhere. This system which can differentiate eyes are opened or closed and detects drowsiness which can prevent the person from entering the state of sleepiness.
Predicts the person's drowsiness on the basis of continuous eye moments. The general flow of our drowsiness detection algorithm is fairly straightforward. A camera is setup to monitor stream of faces. After which, we apply opencv to extract the eye regions.
If eye and lips remain close for more than certain time then alert beep is generated and also continues monitoring make this system more accurate.This system will be portable, scalable, reliable.

**Limitations:**

**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 Objects detection:** In this case, considering multiple faces in frame so it can detect multiple nearby objects using Haar Classifier but Dlib classifier can be used to remove fluctuate but it won’t consider multiple people in one single frame.

# CHAPTER 4
# REFERENCE

1. Smart Alarm System (Drowsiness Detection System Review Paper) Chiranjit Das1 , Chirag Bhatt2 , Uvesh Belim3 , Taha Bhatia4 , Jinil patel5 1, 2, 3, 4, 5Parul University
2. Driver Drowsiness Detection System in Automotive Vehicles Mrs. Ashwini P Department of Information Science and Engineering VVIET, Mysuru, India.
3. https://www.youtube.com/watch?v=O5_--oZPbgQ
4. https://www.tensorflow.org/api_docs/python/tf/keras
5. https://www.academia.edu/38928274/REAL_TIME_SLEEP_DROWSINESS_DETECTION_Project_Report