SENTIMENT ANALYSIS USING MACHINE LEARNING

A Minor Project Report
submitted by

ABHIGYAN SEN
TNU2019020100032

Under the Supervision of

Dr. USHA RANI GOGOI
Assistant Professor
(*Dept. of Computer Science Engineering*)

In partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE ENGINEERING

at

THE NEOTIA UNIVERSITY, KOLKATA, WEST BENGAL, INDIA

December, 2022

CERTIFICATE

We hereby recommend that the Project entitled "*Sentiment Analysis using Machine Learning*" worked under our guidance may please be accepted in the partial fulfillment of the requirement for the degree of Bachelor in Technology in the Computer Science and Engineering with specialization in Cyber Security of 'The Neotia University'.

The project report in our opinion is worthy for its acceptance. During the work Abhigyan Sen was found to be sincere, regular and hard-working and has successfully completed the thesis work assigned to him.

Dr. (*Prof.*) Partha Kumar Mukherjee	Dr. Usha Rani Gogoi
(HOD, CSE)	(Project Guide)
The Neotia University	The Neotia University

DECLARATION OF ORIGINALITY AND COMPLIANCE OF ACADEMIC ETHICS

I hereby declare that this report contains literature survey and original research work done by the under signed candidate, as part of my "Bachelor in Technology Studies".

All information in this document have been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by these rules and conduct, I have cited and referenced all materials that are not original to this work.

Name: ABHIGYAN SEN
Roll No: TNU2019020100032
Specialization: Cyber Security

Project Title: SENTIMENT ANALYSIS USING MACHINE LEARNING

Signature with Date:

Contents

Chapter 1

Introduction

Real Time Drowsiness behaviours which are related to fatigue are in the form of eye closing, head nodding or the brain activity. 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 sagging posture, leaning of driver's head and open or closed state of eyes.

Physical changes such as Nodding and Yawning are not practical and it varies from person to person and hence are not considered as factors for detecting Drowsiness. So, the best way is by Vision Based Techniques which is judging the state by the status of eyes which if closed or partially open depict Drowsiness or else Active or Normal State. In addition micro sleeps are that short period of sleeps that last for 2 to 3 minutes and are good indicators of fatigue. Thus, by continuously monitoring of eyes of the driver one can detect the sleepy state of the driver and a warning is issued.

In this project Drowsiness is detected by using various Deep Learning Techniques and Live Detection using Machine Learning.

Chapter 1.1

Background

Road safety continues to be a major developmental issue, a public health concern and a leading cause of death and injury across the world, killing more than 1.35 million globally as reported in the Global Status report on Road Safety 2018 with 90% of these casualties taking place in the developing countries and 11% alone being accounted for by India.

Accidents are killing more people in India than terrorism or Natural Disasters and yet we never talk about them. In 2016, the total number of RTAs has been noted as 4,80,652 accidents with mortality of 1,50,785 lives in the country. This shows that on average, 1317 RTAs and 413 deaths are taking place on Indian roads everyday or 55 accidents and 17 deaths every hour.

As per the Report on Road accidents in India 2019, the accident related deaths in India in 2019 were 1,51,113 in number. It is indeed a matter of great concern that despite the continuing efforts of the Government in this regard and our commitments for halving fatalities we have not been able to register significant progress on this front.

Exhausted Drivers who doze off at the wheel are responsible for about 40% of Road Accidents, says a study by the Central Road Research Institute (CRRI) on the 300km Agra-Lucknow Expressway.

Hence, in a country where road accidents claim nearly 3 lives every minute, the report related to sleep drivers has underlined the need for educating highway motorists about the importance of taking frequent breaks and proper sleep for safety.

Chapter 1.2

Objective

The aim here is to implement the system as a prototype by capturing live images and detecting if an individual is Drowsy or not.

There are two objectives:

Primary Objective is to design a model to detect Drowsiness. To explore the efficiency of the different models in Drowsiness Detection.

Secondary Objective is to design an application to detect Drowsiness.

Chapter 1.3

Scope

How much work done and what work is done

Chapter 1.4

Organization of Group Project

Chapter 1 gives an abstract about the project and also denotes why is the project important.

Chapter 2 summarizes of the existing models for the same and also the limitations and gaps in those models.

Chapter 3 describes the preparation of our Dataset used in the project along with its origin.

Chapter 4 shows the methods for detecting Drowsiness using Deep Learning Models and Detecting Drowsiness Live using Machine Learning.

Chapter 5 shows the results of the experiments performed.

Chapter 2

Literature Review

In this section we will see a summarized version of the other existing methodologies that have been proposed by researchers for Drowsiness detection and Blink detection during the recent years.
We also get to see their accuracy as well as their limitations.

Chapter 2.1

Literature survey on cnn based methods afte

Convolutional Neural Network Documentation

S No.	NAME	AUTHOR	METHOD	ACCURACY	DATASET	LIMITATION	HYPERLINK
01	**A Survey Paper on Drowsiness Detection and Alarm System for Drivers**	*Prakash Choudhury, Rahul Sharma, Gautam Singh, Smarjeet Das, Sumedh G. Dhengre*	Used Neurons for Detection	High Accuracy		Lighting Condition Background Implement Calculation intensive	Paper Link 01
02	**Driver Drowsiness Detection by Applying Deep Learning Techniques to Sequences of Images**	*Elena Magán, M. Paz Sesmero, JuanAlonso-Weber Araceli Sanchis*	CNN	65%	Data Link	Too many false positives	Paper Link 02
03	**Driver Fatigue Detection based on Convolutional Neural Networks Using EM-CNN**	*Zuopeng Zhao, Nana Zhou, Lan Zhang, Hualin Yan, Yi Xu, Zhongxin Zhan*	CNN	AlexNet 16, GoogleLeNet and ResNet50, showing accuracy and sensitivity rates of 93.623% and 93.643% respectively	Information Technology company called Biteda, Driver Images Data Set	There is a high degree of randomness and contingency between driving state of vehicle and vehicle fatigue	Paper Link 03
04	**Deep CNN: A Machine Learning Approach for Driver Drowsiness Detection Based on Eye State**	*Venkata Rami Reddy, Srinivasulu Reddy Uyyala, VenkataKrishna Kishore Kolli*	CNN	SoftMax Layer in CNN with accuracy with 96.42%	NTHU Driver Dataset	Pose accuracy in regression is overcome purposed stalked CNN	Paper Link 04

S No.	NAME	AUTHOR	METHOD	ACCURACY	DATASET	LIMITATION	HYPERLINK
05	**Driver Drowsiness Detection Model Using Convolutional Neural Networks Techniques for Android Application**	*Rateb Jabbar, Mohammed Shinoy, Mohamed Kharbeche, Khalifa Al-Khalifa, Moez Krichen, Kamel Barkaoui*	CNN	83.33%	NTHU Dataset	Accuracy for all categories where the maximum size of the model did not exceed 75 KB	Paper Link 05
06	**CNN Based Driver Drowsiness Detection System Using Emotion Analysis**	*H. Varun Chand J. Karthikeyan*	CNN	80%	From the NCRB dataset and Time dataset	Trained Data are best in compared to real data	Paper Link 06

Chapter 2.2

Machine Learning Documentation

Face detection is one of the most fundamental aspects of computer vision. It is the base of many further studies like identifying specific people to marking key points on the face.

Haar Cascade were improved way back in 2001 by Paul Viola and Micheal Jones in their paper, “Rapid Object Detection using a Boosted Cascade of features from them. It is superfast to work with simple CNN, it extracts a lot of features from images.

```
import cv2classifier =
cv2.CascadeClassifier('models/haarcascade_frontalface2.
xml')
img = cv2.imread('test.jpg')
faces = classifier. result detectMultiScale(img)#
#to draw faces on image
for result in faces:
    x, y, w, h = result
    x1, y1 = x + w, y + h
    cv2.rectangle(img, (x, y), (x1, y1), (0, 0, 255), 2)
```

Also Haar Cascade requires a lot of positive and negative training images to train and their accuracy is also very poor. Hence, we choose **Dlib** instead of Haar Cascade.

The frontal face detector provided by dlib works using features extracted by Histogram of Oriented Gradients (HOG) which are then passed through an SVM. In the HOG feature descriptor, the distribution of the directions of gradients is used as features. Moreover, Dlib provides a more advanced CNN based face detector.

Dlib is a modern C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real world problems. Although it is written in C++ it has python bindings to run it in python. It also has a great facial landmark detector which I used here to detect the face. It is used in both industry and academia in a wide

range of domains including robotics, embedded devices, mobile phones, and large high performance computing environments. Dlib's open source licensing allows one to use it in any application, free of charge.

```
dlib.get_frontal_face_detector()
→  dlib::object_detector<dlib::fhog_pyrmid<dlib::pyra
mid_down<6u>, dlib::default_fhog_featureectractiour>
>
```

Return the Default Face Discover

The frontal face detector tool of the dlib library is a face detector tool which has 68 landmarks which It uses to detect various parts of the face i.e eyes, nose, mouth etc. Among these our point of interest is only the eyes.

Chapter 2.3

Existing Work

There are different types of methodologies have been developed to find out drowsiness.

Physiological level approach: This technique is an intrusive method wherein electrodes are used to obtain pulse rate, heart rate and brain activity information. ECG is used to calculate the variations in heart rate and detect different conditions for drowsiness. The correlation between different signals such as ecg (electrocardiogram), EEG (electroencephalogram), and EMG (electromyogram) are made and then the output is generated whether the person is drowsy or not.

Behavioural based approach: In this technique eye blinking frequency, head pose, etc. of a person is monitored through a camera and the person is alerted if any of these drowsiness symptoms are detected.

The various technology that can be used are discussed as:

TensorFlow: It is an open-source software library for dataflow programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production.

TensorFlow computations are expressed as stateful dataflow graphs. The name TensorFlow derives from the operations that such neural networks perform on multidimensional data arrays. These arrays are referred to as "tensors".

Kivy: is an open source Python library for developing mobile apps and other multitouch application software with a natural user interface (NUI). It can run on Android, iOS, Linux, OS X, and Windows. Distributed under the terms of the MIT license, Kivy is free and open source software. Kivy is the main framework developed by the Kivy

organization, alongside Python for Android, Kivy iOS, and several other libraries meant to be used on all platforms.

Chapter 3

Dataset Preparation

Data preparation may be one of the most difficult steps in any machine learning project.

The reason is that each dataset is different and highly specific to the project. Nevertheless, there are enough commonalities across predictive modelling projects that we can define a loose sequence of steps and subtasks that you are likely to perform.

This process provides a context in which we can consider the data preparation required for the project, informed both by the definition of the project performed before data preparation and the evaluation of machine learning algorithms performed after.

Chapter 3.1

Dataset Description

The University of Texas at Arlington Real-Life Drowsiness Dataset (UTA-RLDD) was created for the task of multi-stage drowsiness detection, targeting not only extreme and easily visible cases, but also subtle cases when subtle micro-expressions are the discriminative actors. Detection of these subtle cases can be important for detecting drowsiness at an early stage, so as to activate drowsiness prevention mechanisms. Subtle micro-expressions of drowsiness have physiological and instinctive sources, so it can be difficult for actors who pretend to be drowsy to realistically simulate such expressions.

Our UTA-RLDD dataset is the largest to date realistic drowsiness dataset. Here this dataset is used.

In live detection, live input is fed into the program through the camera.

Chapter 3.2

Data Processing

In data processing, At first video data is collected then the the video data is cropped in small video which size is 10s.Then the video is converted into frames along with 30fps, then the frames are resized into frame size of (227× 227 × 3) which is the input format of AlexNet and then the methodology of AlexNet is applied.

Chapter 4

Methodology

Here, we get to see what algorithms are used to for Drowsiness detection using Deep Learning and the steps required to reach to the Final Conclusion.

We also get to see how Live Drowsiness Detection is acquired using Machine Learning.

Chapter 4.1

Drowsiness Detection using Deep Learning

Here…

Chapter 4.2

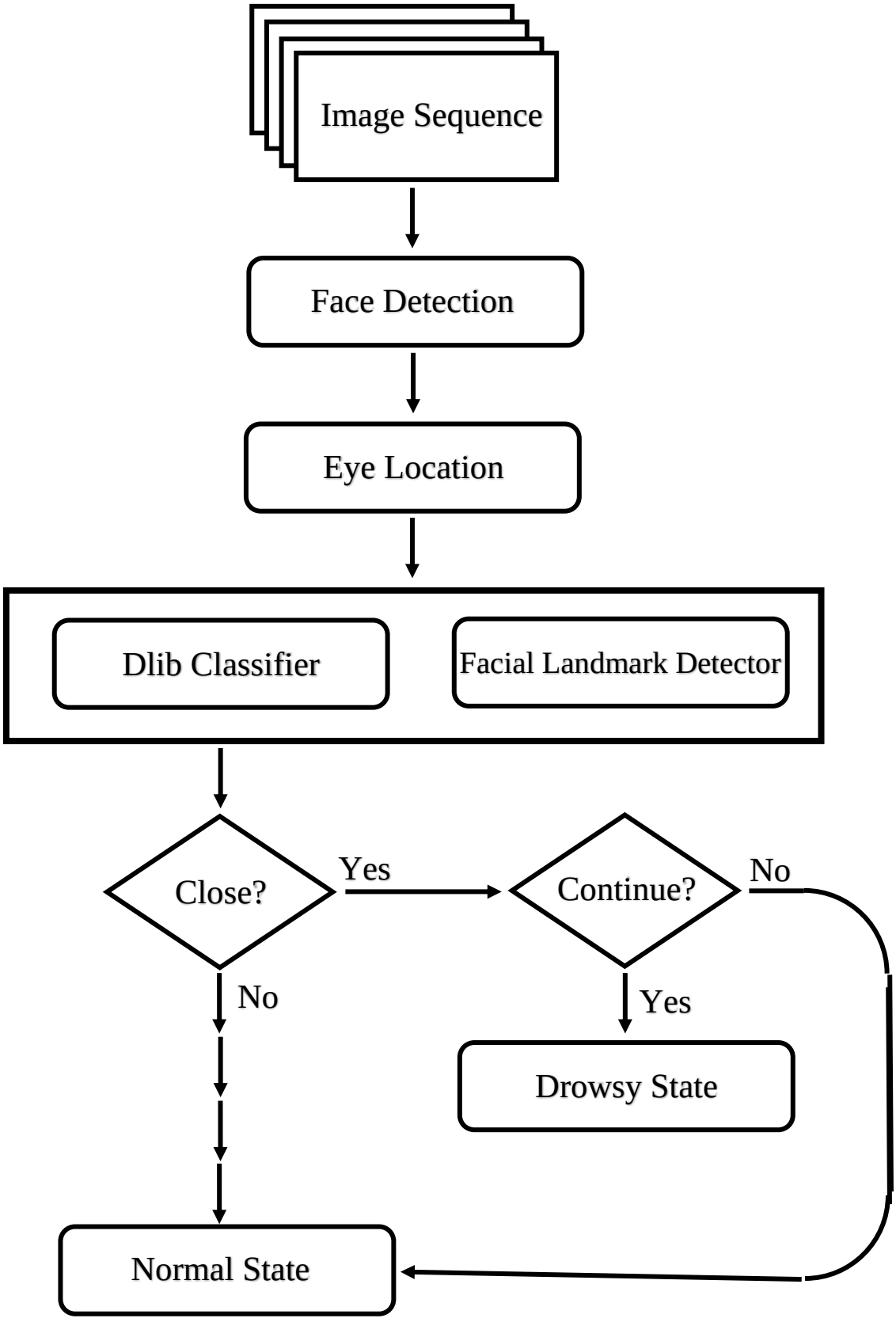
Live Drowsiness Detection using Machine Learning

Machine learning: Machine learning is the kind of programming which gives computers the capability to automatically learn from data without being explicitly programmed. This means in other words that these programs change their behavior by learning from data. Python is clearly one of the best languages for machine learning. Python does contain special libraries for machine learning namely scipy, pandas and numpy which great for linear algebra and getting to know kernel methods of machine learning. The language is great to use when working with machine learning algorithms and has easy syntax relatively.

OpenCV: OpenCV stands for Open Source Computer Vision. It's an Open Source BSD licensed library that includes hundreds of advanced Computer Vision algorithms that are optimized to use hardware acceleration. OpenCV is commonly used for machine learning, image processing, image manipulation, and much more. OpenCV has a modular structure. There are shared and static libraries and a CV Namespace.

In short, OpenCV is used in our application to easily load bitmap files that contain landscaping pictures and perform a blend operation between two pictures so that one picture can be seen in the background of another picture. This image manipulation is easily performed in a few lines of code using OpenCV versus other methods. OpenCV.org is a must if you want to explore and dive deeper into image processing and machine learning in general.

FLOWCHART AND ALGORITHM



ALGORITHM

The various detection stages are discussed as:

Face Detection: The facial landmark detector included in the dlib library is an implementation of the One Millisecond Face Alignment with an Ensemble of Regression Trees paper by Kazemi and Sullivan (2014). This method starts by using:

1. A training set of labelled facial landmarks on an image. These images are manually labelled, specifying specific (x, y)-coordinates of regions surrounding each facial structure.
2. Priors, of more specifically, the probability on distance between pairs of input pixels. The pre-trained facial landmark detector inside the dlib library is used to estimate the location of 68 (x, y)-coordinates that map to facial structures on the face. The indexes of the 68 coordinates can be visualized on the image below:

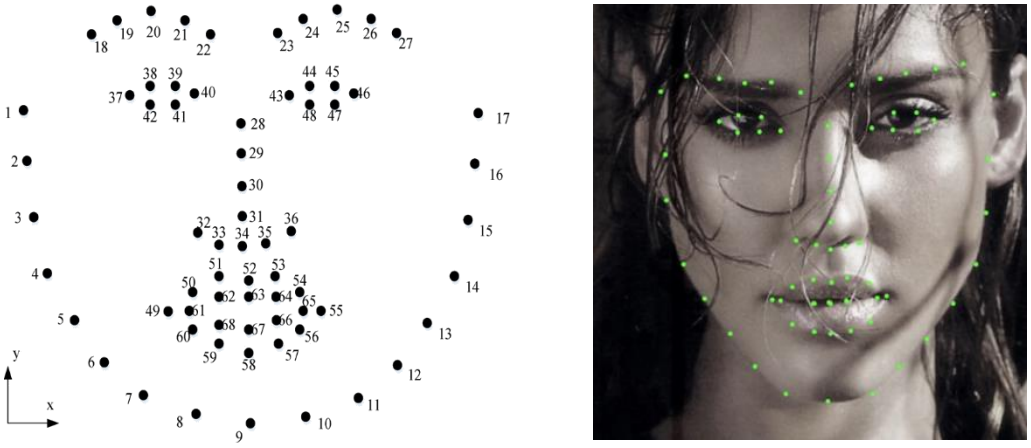


Fig 4.2.1 68 landmarks of "**get_frontal_face_detector**"

Eye detection: In the system we have used facial landmark prediction for eye detection Facial landmarks are used to localize and represent salient regions of the face, such as:

- Eyes
- Eyebrows
- Nose
- Mouth
- Jawline

Facial landmarks have been successfully applied to face alignment, head pose estimation, face swapping, blink detection and much more. In the context of facial landmarks, our goal is detecting important facial structures on the face using shape prediction methods.

The eyes are detected with the help of landmarks {37, 38, 39, 40, 41} which detects the left landmarks {43, 44, 45, 46, 47} detecting the right eye. (Figure 4.2.1)

Recognition of Eye State: Area of the eye is estimated from optical flow, by sparse tracking or by frame-to-frame intensity differencing and adaptive thresholding, and finally a decision is made whether the eye is open or close.

The status of the eye i.e open, closed or partially open are judged by calculating the distance.

For computing the distance Euclidian Distance is used.

$$d(\mathbf{p},\mathbf{q}) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$

P, q are two points on euclidian plane
p_i q_i are euclidian vectors
N = n~space

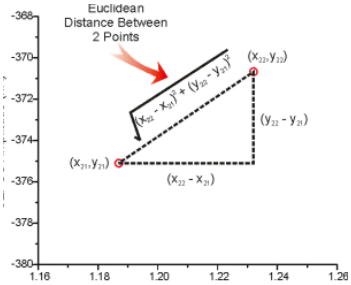


Fig 4.2.2 Euclidian Distance between two points

Eye Aspect Ratio Calculation: For every video frame, the eye landmarks are detected and the EAR is computed.

The Eye Aspect Ratio (EAR) is computed by the using the formula:

$$Ratio = \frac{Sum\ of\ both\ the\ short\ distance}{2 \times long\ distance}$$

After a lot of days of tiring research and experimentation, the ratio of **0.25** is a determined ratio for open eye failing which for a certain time, the individual is considered to be drowsy.

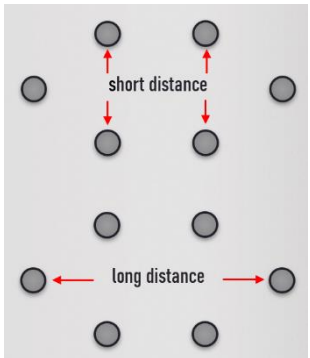


Fig 4.2.3 short and long distance between eyes

Drowsiness Detection: The last step of the algorithm is to determine the person's condition based on a pre-set condition for drowsiness. The average blink duration of a person is 100-400 milliseconds (i.e. 0.1-0.4 of a second). Hence if a person is drowsy his eye closure must be beyond this interval. We set a time frame of 6 seconds. If the eyes remain closed for five or more seconds, drowsiness is detected and alert pop regarding this is triggered.

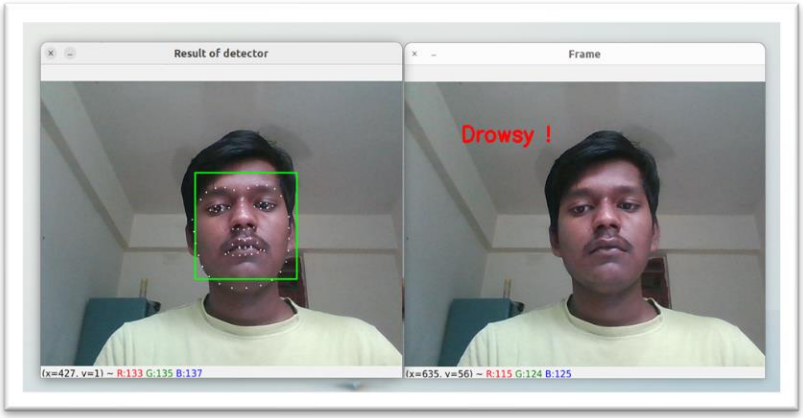
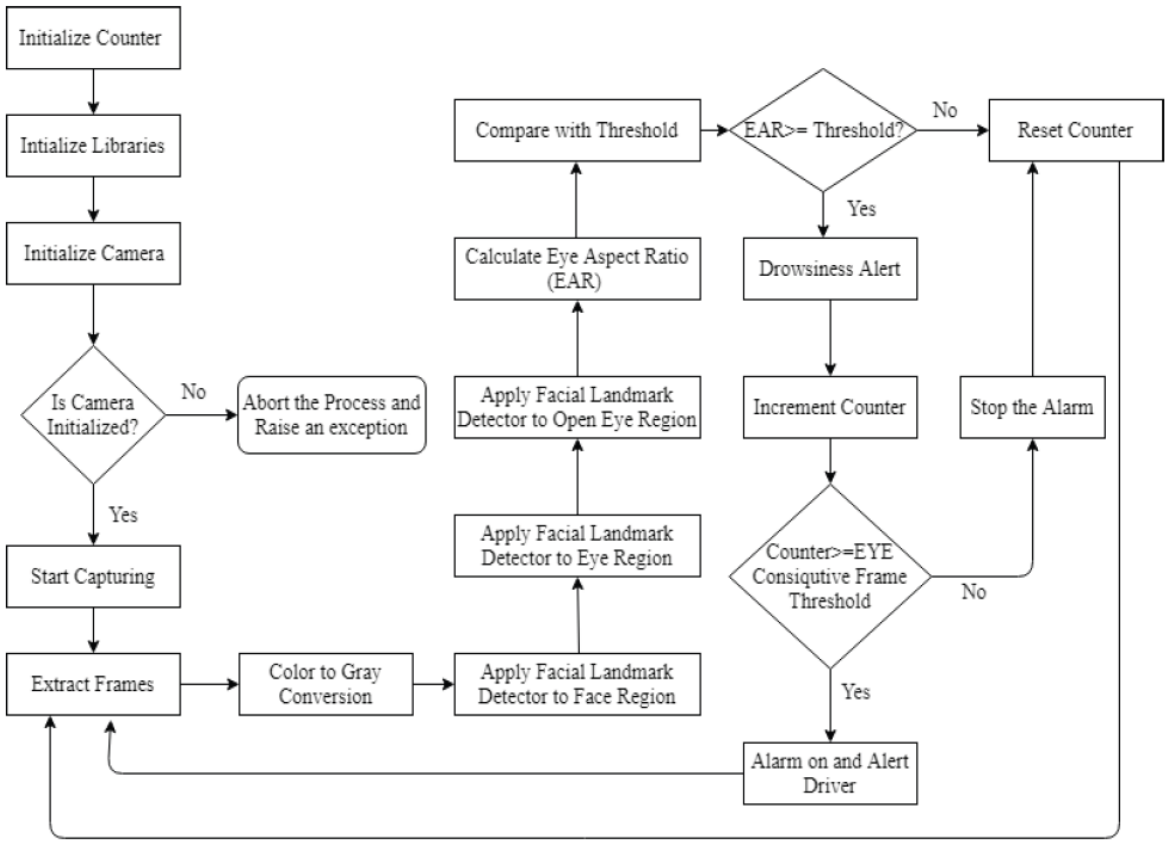


Fig 4.2.4 Drowsy State

MODELLING DIAGRAM



Chapter 5

Experimental Results

IMPORTING LIBRARIES

```
[ ]  1 #Importing OpenCV Library for basic image processing functions
     2 import cv2
     3 # Numpy for array related functions
     4 import numpy as np
     5 # Dlib for deep learning based Modules and face landmark detection
     6 import dlib
     7 #face_utils for basic operations of conversion
     8 from imutils import face_utils
```

INITIALIZE CAMERA

```
[ ]  1 #Initializing the camera and taking the instance
     2 cap = cv2.VideoCapture(0)
```

INITIALIZING FACE DETECTOR

```
     4 #Initializing the face detector and landmark detector
     5 detector = dlib.get_frontal_face_detector()
     6
     7 predictor = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")
```

STATUS MARKING FOR CURRENT STATE

```
     9 #status marking for current state
    10 sleep = 0
    11 drowsy = 0
    12 active = 0
    13 status=""
    14 color=(0,0,0)
```

COMPUTING EUCLIDIAN DISTANCE

```
    16 def compute(ptA,ptB):
    17     dist = np.linalg.norm(ptA - ptB)
    18     return dist
```

Chapter 5

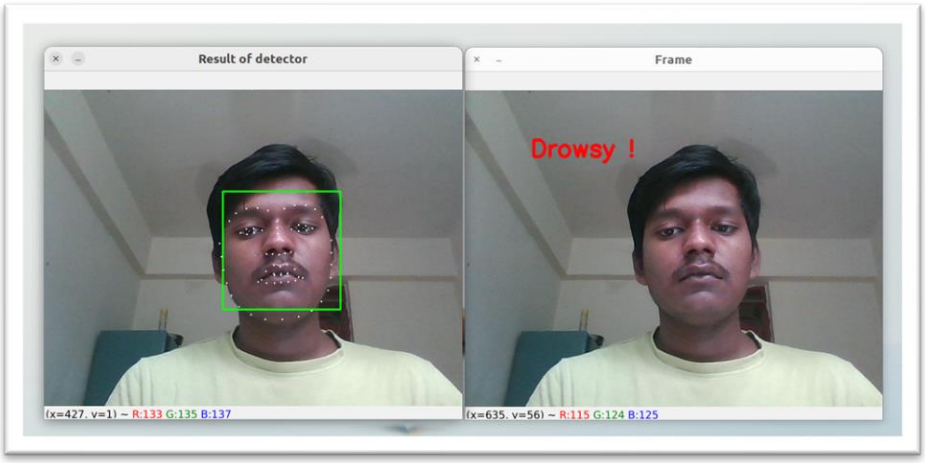
Experimental Results

APPENDIX

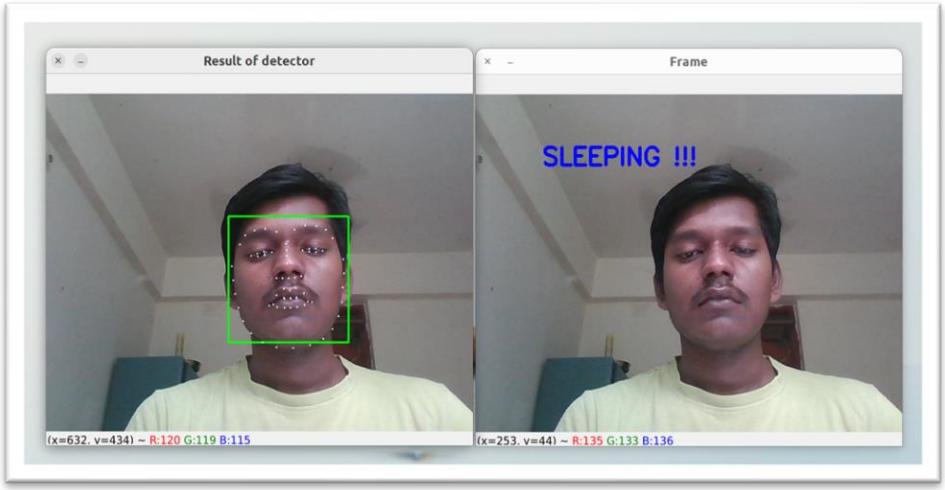
```
[ ]
#Importing OpenCV Library for basic image processing functions
import cv2
# Numpy for array related functions
import numpy as np
# Dlib for deep learning based Modules and face landmark detection
import dlib
#face_utils for basic operations of conversion
from imutils import face_utils
```

```
#Initializing the camera and taking the instance
cap = cv2.VideoCapture(0)

#Initializing the face detector and landmark detector
detector = dlib.get_frontal_face_detector()

predictor = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")

#status marking for current state
sleep = 0
drowsy = 0
active = 0
status=""
color=(0,0,0)

def compute(ptA,ptB):
    dist = np.linalg.norm(ptA - ptB)
    return dist

def blinked(a,b,c,d,e,f):
    up = compute(b,d) + compute(c,e)
    down = compute(a,f)
    ratio = up/(2.0*down)

    #Checking if it is blinked
    if(ratio>0.25):
        return 2
    elif(ratio>0.21 and ratio<=0.25):
        return 1
    else:
        return 0
```

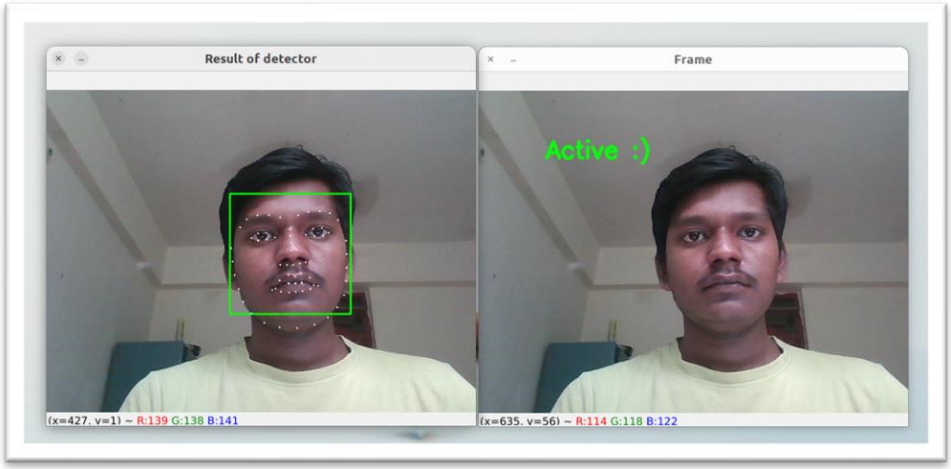
```python
while True:
    _, frame = cap.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    faces = detector(gray)
    #detected face in faces array
    for face in faces:
        x1 = face.left()
        y1 = face.top()
        x2 = face.right()
        y2 = face.bottom()

        face_frame = frame.copy()
        cv2.rectangle(face_frame, (x1, y1), (x2, y2), (0, 255, 0), 2)

        landmarks = predictor(gray, face)
        landmarks = face_utils.shape_to_np(landmarks)

        #The numbers are actually the landmarks which will show eye
        left_blink = blinked(landmarks[36],landmarks[37],
            landmarks[38], landmarks[41], landmarks[40], landmarks[39])
        right_blink = blinked(landmarks[42],landmarks[43],
            landmarks[44], landmarks[47], landmarks[46], landmarks[45])

        #Now judge what to do for the eye blinks
        if(left_blink==0 or right_blink==0):
            sleep+=1
            drowsy=0
            active=0
            if(sleep>6):
                status="SLEEPING !!!"
                color = (255,0,0)

        elif(left_blink==1 or right_blink==1):
            sleep=0
            active=0
            drowsy+=1
            if(drowsy>6):
                status="Drowsy !"
                color = (0,0,255)

        else:
            drowsy=0
            sleep=0
            active+=1
            if(active>6):
                status="Active :)"
                color = (0,255,0)

        cv2.putText(frame, status, (100,100), cv2.FONT_HERSHEY_SIMPLEX, 1.2, colo

        for n in range(0, 68):
            (x,y) = landmarks[n]
            cv2.circle(face_frame, (x, y), 1, (255, 255, 255), -1)

    cv2.imshow("Frame", frame)
    cv2.imshow("Result of detector", face_frame)
    key = cv2.waitKey(1)
    if key == 27:
        break
```



DROWSY STATE



ACTIVE STATE



SLEEPING STATE

References

- http://dlib.net/
  https://pyimagesearch.com/2021/04/19/face-detection-with-dlib-hog-and-cnn/
- https://www.slidescarnival.com/quintus-free-presentation-template/1405#preview
- chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/viewer.html?pdfurl=https%3A%2F%2Fd1wqtxts1xzle7.cloudfront.net%2F54342040%2FIRJET-V3I12315-with-cover-page-v2.pdf%3FExpires%3D1649339813%26Signature%3DXR9mujjB0zNpvtBoB4rDNU4pSGJ~S8JKw2WharRRIiQXpn2Fo6g7kpgNnKrWbrTx6af5EK1CBOH18xbaTdoxyZ9CYSDwvDz1GOcBvcXbXsRQyxfzdlMB00QOQbje9GFii7OtXKYDz83TI03UyT-rFVTdrhdZ4Jfbi0ovPRz9gaVhOW6e2eZqKviZph5Od0voMPQtGByy3Q8nMj57529gVTD3PNLahjLcfhdubI30VEkgX6o3i6A~k8yS-1XlVLm~6YpfIsS3V1hk8PRTEWfDHNaMA9yz2BFabrgXtyhZU8mdGI5DMvFB0aYJ3RvPmu6qeZX4iGuiLIUR-rdwOVgWSw__%26Key-Pair-Id%3DAPKAJLOHF5GGSLRBV4ZA&clen=983719&chunk=true
- https://www.mdpi.com/2076-3417/12/3/1145/pdf
- https://www.hindawi.com/journals/cin/2020/7251280/
- https://www.researchgate.net/publication/338251837_Deep_CNN_A_Machine_Learning_Approach_for_Driver_Drowsiness_Detection_Based_on_Eye_State
- chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/viewer.html?pdfurl=https%3A%2F%2Farxiv.org%2Fpdf%2F2002.03728.pdf&clen=954778&chunk=true
- https://www.techscience.com/iasc/v31n2/44533/pdf
- https://www.slideshare.net/vigneshwarvs/driver-drowsiness-detection?qid=78ba9d2b-3734-4e16-b681-1985074d1507&v=&b=&from_search=2
- https://www.istockphoto.com/video/stressed-worried-driver-gm1217195460-355205820
- https://www.istockphoto.com/video/tired-exhausted-businessman-driving-a-car-at-evening-gm618279894-107704979
- https://www.istockphoto.com/search/2/film?phrase=Sleepy
- https://en.wikipedia.org/wiki/Euclidean_distance
- https://www.researchgate.net/figure/The-68-landmarks-detected-by-dlib-library-This-image-was-created-by-Brandon-Amos-of-CMU_fig2_329392737
- https://www.google.com/url?sa=i&url=https%3A%2F%2Fvitalflux.com%2Fdifferent-types-of-distance-measures-in-machine-learning%2F&psig=AOvVaw00kly8QN81k6oJC5B-CRiB&ust=1656345975826000&source=images&cd=vfe&ved=0CAkQjRxqFwoTCLihlde_y_gCFQAAAAAdAAAAABAD
- https://towardsdatascience.com/face-detection-with-haar-cascade-727f68dafd08
- https://openaccess.thecvf.com/content_cvpr_2014/html/Kazemi_One_Millisecond_Face_2014_CVPR_paper.html
- https://ieeexplore.ieee.org/document/6909637
- https://machinelearningmastery.com/what-is-data-preparation-in-machine-learning/
- https://openaccess.thecvf.com/content_cvpr_2014/papers/Kazemi_One_Millisecond_Face_2014_CVPR_paper.pdf
- https://medium.com/analytics-vidhya/eye-aspect-ratio-ear-and-drowsiness-detector-using-dlib-a0b2c292d706