REAL TIME DRIVER DROWSINESS DETECTION

A PROJECT REPORT

Submitted by

DANUSH S 2017115518

LALITH PRASAD S 2017115544

PREMCHANDER S 2017115572

submitted to the Faculty of

INFORMATION AND COMMUNICATION ENGINEERING

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY

COLLEGE OF ENGINEERING, GUINDY

ANNA UNIVERSITY

CHENNAI 600 025

APRIL 2021

ANNA UNIVERSITY

CHENNAI - 600 025

BONA FIDE CERTIFICATE

Certified that this project report titled REAL TIME DRIVER DROWSINESS DETECTION is the bona fide work of DANUSH S (2017115518), LALITH PRASAD S (2017115544), PREMCHANDER S (2017115572) who carried out project work under my supervision. Certified further that to the best of my knowledge and belief, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or an award was conferred on an earlier occasion on this or any other candidate.

PLACE:
CHENNAI
DATE: 07/04/2021

Dr. ABIRAMI MURUGAPPAN
ASSISTANT PROFESSOR
PROJECT GUIDE
DEPARTMENT OF IST, CEG
ANNA UNIVERSITY
CHENNAI 600025

COUNTERSIGNED
Dr. SASWATI MUKHERJEE
HEAD OF THE DEPARTMENT
DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY
COLLEGE OF ENGINEERING, GUINDY
ANNA UNIVERSITY
CHENNAI 600025

ABSTRACT

One of the most powerful and compelling types of Deep Learning is Computer vision. Because of advances in Artificial Intelligence, Computer Vision has been able to take great leaps in recent years and has been able to surpass humans in some tasks related to detecting and labeling objects. One such task is detecting drowsiness of a driver in a vehicle. The proposed project attempts to monitor the driver continuously, check if the driver gets drowsy and alert the driver. In addition to detecting drowsiness after the driver falls asleep, it also identifies key patterns in the driver's face using image similarity and predicts whether the driver is going to fall asleep in a few seconds. It can also predict when the driver will fall asleep in the future based on his/her drowsiness history. This project aims to prevent on-road accidents due to drowsiness by extracting key features from the face using Deep Learning. The whole system is implemented using raspberry pi 4 which can run as a standalone system without requiring additional hardware resources besides USB camera for live video capturing.

This project involves three main modules. Drowsiness detection module consists of face detection and facial landmarks detection as sub modules. Face detection is implemented using HOG feature extraction and SVM classifier. Facial landmarks detection is implemented using an Ensemble of regression trees. The dataset used is from Dlib which provides a 300-W dataset of face images with annotations. Output of drowsiness detection module is whether or not the driver has fallen asleep. Image similarity analysis module is implemented using a Convolutional Neural Network. Custom dataset has been collected, captured using web camera. The dataset contains 401 images belonging to 2 classes - Awake and Going to sleep. Awake class has 124 images and Going to sleep class has 277 images. Output of image similarity analysis

module is whether or not the driver is going to fall asleep in a few seconds. Sleep prediction module is implemented using LSTM, Bidirectional LSTM and GRU models. Dataset used for this module is sleep data from Kaggle. It consists of 887 rows and 8 columns. Output of sleep prediction module is prediction of next day's timing at which the driver may fall asleep.

திட்டப்பணிச் சுருக்கம்

ஆழ்ந்த கற்றலின் மிகவும் சக்தி வாய்ந்த வகைகளில் ஒன்று கணினி பார்வை. செயற்கை நுண்ணறிவின் முன்னேற்றங்கள் காரணமாக, கணினி பார்வை சமீபத்திய ஆண்டுகளில் பெரும் பாய்ச்சல் எடுக்க முடிந்தது மற்றும் பொருட்களைக் கண்டறிதல் மற்றும் லேபிளிங் செய்வது தொடர்பான சில பணிகளில் மனிதர்களை மிஞ்ச முடிந்தது. அத்தகைய ஒரு பணி ஒரு வாகனத்தில் ஒரு ஓட்டுநரின் மயக்கத்தைக் கண்டறிவது. முன்மொழியப்பட்ட திட்டம் தொடர்ந்து டிரைவரை கண்காணிக்க முயற்சிக்கிறது, டிரைவர் மயக்கமடைகிறதா என்று சரிபார்த்து டிரைவரை எச்சரிக்க வேண்டும்.

டிரைவர் தூங்குவதற்கு சில நிமிடங்களுக்கு முன்பு அவரை எச்சரிக்கவும் கணினி முயற்சிக்கும். எதிர்காலத்தில் இயக்கி எப்போது தூங்குகிறது என்பதையும் இது கணிக்க முடியும். முகத்தில் இருந்து முக்கிய அம்சங்களை பிரித்தெடுப்பதன் மூலமும், ஓட்டுநர் சோர்வடைகிறாரா என்பதை அடையாளம் காண்பதன் மூலமும் மயக்கம் காரணமாக ஏற்படும் சாலை விபத்துகளைத் தடுப்பதை இந்த திட்டம் நோக்கமாகக் கொண்டுள்ளது.

ACKNOWLEDGEMENT

We express our profound sense of gratitude to our guide **Dr. ABIRAMI MURUGAPPAN**, Assistant Professor, Department of Information Science and Technology, College of Engineering Guindy, Anna University, for her invaluable support, guidance and encouragement for the successful completion of this project. Her vision and spirit will always inspire and enlighten us.

Our heartfelt thanks to **Dr. SASWATI MUKHERJEE**, Head, Department of Information Science and Technology, College of Engineering Guindy, Anna University, for the prompt and limitless help in providing the excellent computing facilities to do the project and to prepare the thesis.

We express our sincere gratitude to our review committee members, **Dr.S.SWAMYNATHAN**, Professor, **Dr.T.MALA**, Associate Professor, **Dr.K.VIDYA**, Assistant Professor, **Dr. D.NARASHIMAN**, Teaching Fellow, **Ms.T.SINDHU**, Teaching Fellow, and **Ms.G.MAHALAKSHMI**, Teaching Fellow Department of Information Science and Technology, College of Engineering Guindy, Anna University, for their endless support and understanding spirit during our project presentations.We express our heartiest thanks to all other teaching and non-teaching staffs those who have helped us to do the project and to prepare the thesis.

DANUSH S

LALITH PRASAD S

PREMCHANDER S

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

HOG	Histogram of Oriented Gradients
CNN	Convolutional Neural Networks
YOLO	You Only Look Once
PIL	Python Imaging Library

CHAPTER 1

INTRODUCTION

A driver who falls asleep at the wheel loses control of the vehicle, as a result of which accidents can occur. In order to prevent these devastating accidents, the state of drowsiness of the driver should be monitored. Accidents caused due to drivers getting drowsy can be prevented by using this driver drowsiness detection technology.

1.1 COMPUTER VISION

Computer vision is an interdisciplinary scientific field that deals with how computers can gain high-level understanding from digital images or videos. From the perspective of engineering, it seeks to understand and automate tasks that the human visual system can do. Computer vision tasks include methods for acquiring, processing, analyzing and understanding digital images, and extraction of high-dimensional data from the real world in order to produce numerical or symbolic information, e.g. in the forms of decisions. Understanding in this context means the transformation of visual images into descriptions of the world that make sense to thought processes and can elicit appropriate action. Consequently, computer vision is sometimes seen as a part of the artificial intelligence field or the computer science field in general. The advances on the computer vision field have been astounding. Accuracy rates for object identification and classification have gone from 50 percent to 99 percent in less than a decade and today's systems are more accurate than humans at quickly detecting and reacting to visual inputs.

1.2 MOTIVATION

According to available statistical data, over 1.3 million people die each year on the road and 20 to 50 million people suffer non-fatal injuries due to road accidents. This finding shows how motorists ignore the importance of taking adequate rest and end up endangering lives. In a country where road accidents claim nearly three lives every minute, this data clearly shows the need for a system which can detect drowsiness and prevent such accidents. The motivation for this work is that by implementing a cost effective system, such accidents can be prevented to a large extent.

1.3 NEED FOR PROPOSED WORK

- The existing systems mostly depend on a cloud based environment. The proposed system is built using raspberry pi, which enables it to function completely in a local environment, which makes it suitable for real time use.
- The proposed system uses computer vision technology in order to achieve the expected output so it does not require any sensors to be in contact with the driver, only a camera is enough. In contact approaches, drivers wear or touch some devices to get physiological parameters for detecting the level of their fatigue.
- Most of the accidents due to fatigue happen during night time. This system aims to perform with high accuracy even in low light conditions.

1.4 PROBLEM STATEMENT

Driver drowsiness leads to accidents and in some cases it has even proven to be fatal. By figuring out a solution which is cost effective as well as highly accurate, it can be taken into actual implementation and the number of accidents can be reduced. This project aims to alert the driver in case of drowsiness detection and also try to predict when the driver may fall asleep in the future. This project proposes to design a system to detect driver drowsiness using various machine learning and computer vision techniques.

1.5 OBJECTIVES OF PROPOSED WORK

- The main objective is to first design a system to detect driver's drowsiness by continuously monitoring the driver and extracting the facial features.
- Alert the driver on the detection of drowsiness by using a buzzer/alarm.
- To handle low light images using image illumination techniques.
- To inspect if driver goes out of frame and calculate the deviation ratio.
- To predict the next day's timings when the driver may fall asleep using time series prediction.
- Using image similarity to predict sleeping pattern and forewarn driver.
- Hardware implementation of the application using Raspberry Pi with real time prediction and alerts.

1.6 CHALLENGES OF THE PROPOSED WORK

The following challenges were faced during the implementation of this system

- Implementation of the entire system in Raspberry Pi.
- Capturing and Processing images under different light conditions.
- As drivers' heights are different, the positions of their faces in the video are different. Then, when the driver is driving, his or her head may be moving and turning. Hence, tracking the trajectory of the head in real time is important once the position of the head changes.
- Some drivers wear spectacles. The system should perform well and be capable of carrying out all the processes for those drivers too.
- To accurately predict the time in which the driver may fall asleep in the future.
- To keep track of the driver's sleep data in a real time database.
- Experimenting with and selecting algorithms and models so that they run efficiently in Raspberry Pi.

1.7 ORGANISATION OF THE THESIS

The project report is organised as follows,

Chapter 2 discusses the various existing methods for the modules of the proposed system.

Chapter 3 discusses the various concepts used in the proposed system along with the overall design.

Chapter 4 discusses the implementation details of the project.

Chapter 5 discusses the results and performance analysis of the project.

Chapter 6 concludes the report and discusses about the future work.

CHAPTER 2

LITERATURE SURVEY

A literature survey related to the project work has been done and the observations from relevant papers are discussed in this chapter. The knowledge gained from these papers has helped us to create a better system for movie certification.

2.1 REAL-TIME EMOTION RECOGNITION FROM FACIAL IMAGES USING RASPBERRY PI II

Suchitra et al., [1] proposed the system which detects real time emotion recognition from facial image.Emotions can understand by text,vocal,verbal and facial expressions.Facial expressions play big role in judging emotions of a person.It is found that limited work is done in field of real time emotion recognition using facial images.In the proposed method ,three steps face detection using Haar cascade,features extraction using Active shape Model(ASM),(26 facial points extracted) and Adaboost classifier for classification of five emotions anger, disgust, happiness, neutral and surprise.A Real-time emotion recognition system that recognizes basic emotions like anger, disgust, happiness, surprise and neutral using CMU MultiPIE database consisting 2D images with different illumination and poses.Active shape Model (ASM) for extracting facial points and AdaBoost classifier have been used for developing the emotion recognition software. The software system developed using this proposed method is deployed on Raspberry Pi II as it can be used with robots as the size of Raspberry Pi II is very small, light weighted and very less power supply is needed for it. The input image in real time is captured through webcam and fed to emotion recognition software as input. Emotion recognition

software is deployed in the Raspberry Pi II, which gives classified emotion as output. The recognized emotion is displayed in the monitor.

The key challenge in this paper was that most of the work is done on the frontal view images of the faces.More work need to be done on non-frontal images with different illumination conditions as in real time these global conditions are not uniform.

2.2 AN EMBEDDED REAL-TIME OBJECT DETECTION AND MEASUREMENT OF ITS SIZE

Nashwan Adnan Othman, Mehmet Umut Salur [2] proposed an enhanced technique for detecting objects and computing their measurements in real time from video streams. an object measurement technique for real-time video by utilizing OpenCV libraries and includes the canny edge detection, dilation, and erosion algorithms. In the offered system, Computer Vision used to detect and measure objects. The system can detect and measure objects in a real time video. After the object has been detected by using canny edge detector, the size is obtained for each object by using OpenCV functions. The system consists of two parts which are object detection and object measurement. In the first part, raspberry pi camera used to achieve the frames. In the second part, computer vision module will be applied to the captured frames to determine the objects, then measure each object. The detected object of the current frame immediately will be processed to extract dimensions of objects.Firstly, preprocessing of the image is performed.The camera will capture a frame and the frame will convert to grayscale to increase quickness and accuracy. Objects are detected via canny edge detector algorithm. It is used to detect only one object or multiple objects. By the help of canny edge detector, the converted image will be processed. The canny edge algorithm scans the entire image. After that, execute dilation and erosion algorithm to close holes among edges in the edge frame.The system

applies four operations such as record frames, find edges, find objects, and measure size for each objects and the output is displayed.

The key challenge in this paper was that the accuracy isn't perfect since this is due to the seeing angle and lens deformation.Calibration of camera and setting good width parameter can increase the accuracy.

2.3 DEEP CONVOLUTIONAL NETWORK CASCADE FOR FACIAL POINT DETECTION

Yi Son et al., [3] proposed a new approach for estimation of the positions of facial keypoints with three-level carefully designed convolutional networks. At each level, the outputs of multiple networks are fused for robust and accurate estimation.The deep structures of convolutional networks,global high-level features are extracted over the whole face region at the initialization stage,which help to locate high accuracy keypoints. There are two folds of advantage for this. First, the texture context information over the entire face is utilized to locate each keypoint. Second, since the networks are trained to predict all the keypoints simultaneously, the geometric constraints among keypoints are implicitly encoded. The method therefore can avoid local minimum caused by ambiguity and data corruption in difficult image samples due to occlusions, large pose variations, and extreme lightings. The networks at the following two level-s are trained to locally refine initial predictions and their inputs are limited to small regions around the initial predictions.

The key challenge in this paper was that the facial keypoint detection causes problem when n face images are taken with extreme poses, lightings, expressions, and occlusions.

2.4 STOCK PRICE PREDICTION USING LSTM, RNN AND CNN-SLIDING WINDOW MODEL

Sreelekshmy Selvin et al., [4] proposed a deep learning based formalization for stock price prediction. It is seen that, deep neural network architectures are capable of capturing hidden dynamics and are able to make predictions.For the proposed methodology CNN is identified as the best model. It uses the information given at a particular instant for prediction. Even though the other two models are used in many other time dependent data analysis, it is not out performing the CNN architecture in this case. This is due to the sudden changes that occurs in stock markets. The changes occuring in the stock market may not always be in a regular pattern or may not always follow the same cycle. Based on the companies and the sectors, the existence of the trends and the period of their existence will differ. The analysis of these type of trends and cycles will give more profit for the investors. Convolutional networks use convolution instead of general matrix multiplication in at least one of their layers.The motivation behind using these three models is to identify whether there is any long term dependency existing in the given data.This shows that, the pattern or dynamics identified by the model is common to other companies also.

The key challenge in this paper was that the stock market is a highly dynamical system, the patterns and dynamics existing with in the system will not always be the same. This cause learning problems to LSTM and RNN architecture and hence the models fails to capture the dynamical changes accurately.

CHAPTER 3

SYSTEM ARCHITECTURE AND DESIGN

3.1 SYSTEM ARCHITECTURE

Figure 3.1 shows the overall architecture of the system. The entire system is implemented to run in Raspberry Pi. A Camera is set up to monitor the driver's face.

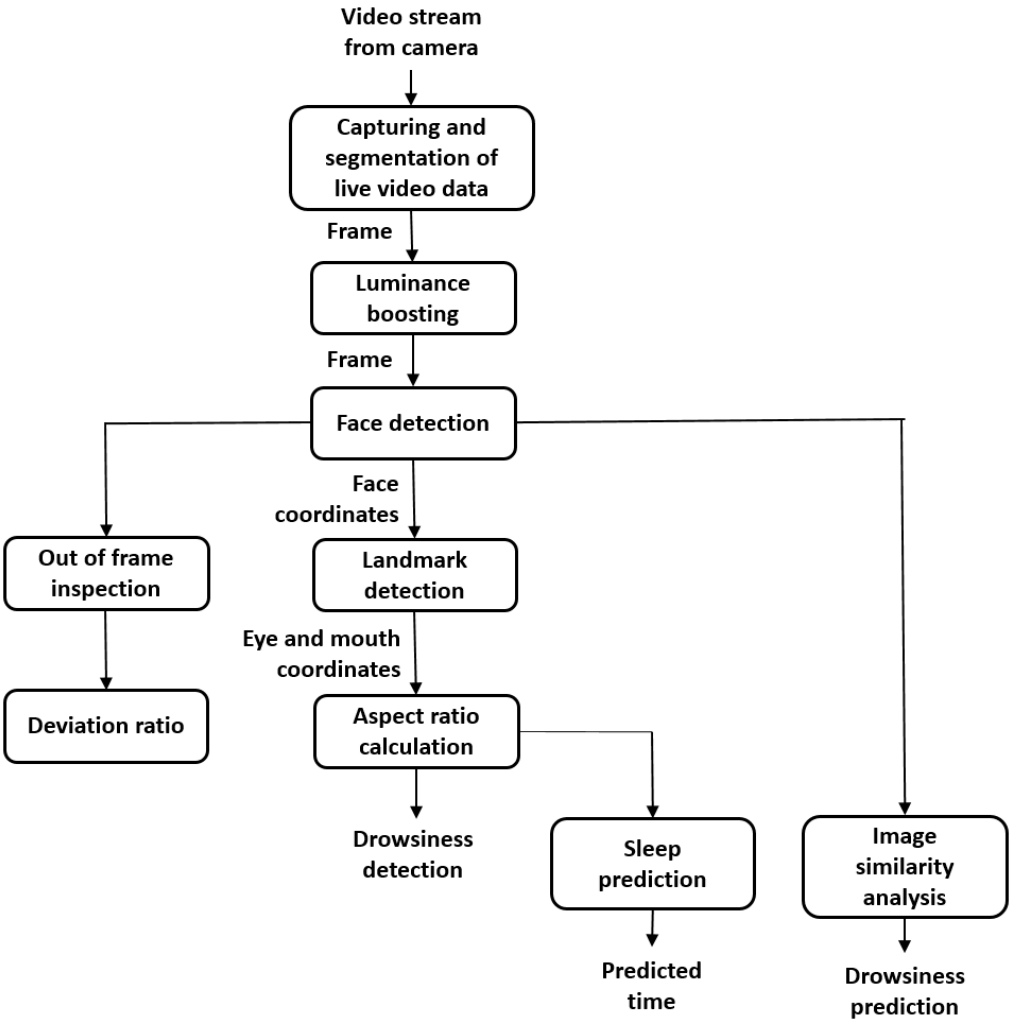

Figure 3.1: System Architecture

The video from the camera is first split into frames. Then, luminance boosting is applied on the frame. The frame then goes through face detection and landmark detection modules for drowsiness detection, sleep prediction module and image similarity analysis module for drowsiness prediction.

The proposed system follows a loosely coupled architecture comprising of the following modules.

- Drowsiness detection
- Sleep prediction
- Image similarity analysis

3.2 DROWSINESS DETECTION MODULE

The main objective of drowsiness detection module is to detect whether the driver has become drowsy and raise alerts. Other objectives include luminance boosting, out of frame inspection and deviation ratio calculation.

3.2.1 LUMINANCE BOOSTING

The frames which go through the computer vision methods are first preprocessed. Figure 3.2 gives the flow diagram of this module. First, the

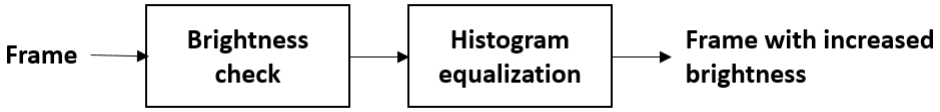

Figure 3.2: Luminance boosting

brightness of the frame is checked by converting the color space of the frame from bgr to ycbcr. The brightness of the frame is then calculated from the y channel which represents the luma characteristic of the image. If low brightness is detected, the b, g and r channels of the frame are equalized and the brightness of the frame is increased to provide as better input to the computer vision modules.

3.2.2 FACE DETECTION

Video stream from the camera is converted into frames and face detection is done on each of the separated frame. Face detection consists of the following sub modules:

- Image scaling
- HOG feature extraction
- Classification of HOG features

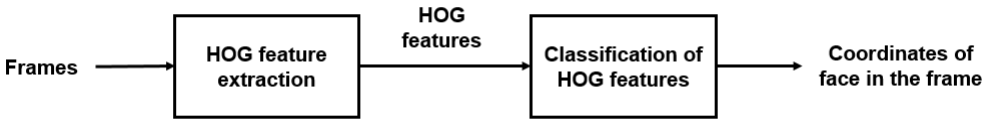

Figure 3.3: Face detection

Figure 3.3 gives the flow diagram of this module. The faces captured by various cameras will differ in size inside the frames. If the input to the classifier doesn't contain the face in correct proportions, it will be misclassified. So, the frame goes through Image scaling process, which will reduce the image size by scaling down repeatedly.

HOG counts occurrences of gradient orientation in localized portions of an image. The result of applying HOG to the frame will yield HOG feature vectors in the same shape and dimensions as the frame. Since images of the same class will have similar HOG features, classification can be done on HOG features.

The feature vectors will be split into 8 by 8 parts. Sliding window is applied which covers each part and the windows will then go through a classifier to decide whether that window contains a face or not. The output of this module is a bounding box specifying the top left coordinates, width and height of window in which face is detected.

3.2.3 FACIAL LANDMARKS DETECTION

Figure 3.4 shows the steps involved in detecting the facial keypoints - eyes and mouth in the frame.

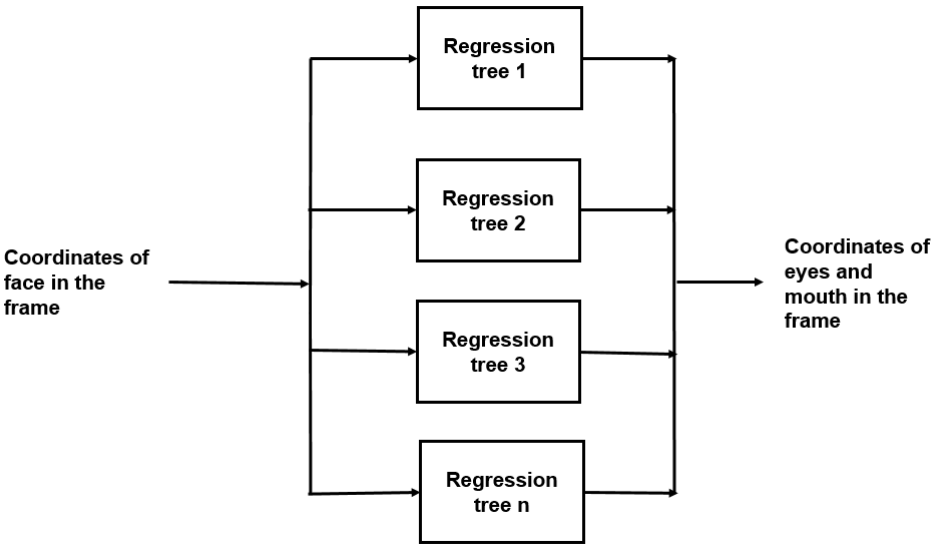

Figure 3.4: Facial Landmark Detection

The frame and bounding box from the face detection module are taken as inputs into the ensemble of regression trees, which will predict the locations of eyes and mouth inside the bounding box. A total of 18 landmark points are obtained as coordinate values around the eyes and mouth. Based on the coordinates of eyes and mouth, their respective aspect ratios are calculated. If the aspect ratio of the eyes are higher than a specific threshold, it indicates that the eyes of the driver are closing and the driver can be alerted. If the aspect ratio of the mouth is lower than a specific threshold, it indicates that the driver is yawning and a small indication can be given to the driver that he/she is getting drowsy.

3.2.4 OUT OF FRAME INSPECTION

Whenever the driver goes out of frame, the location of his/her face in the previous frame is tracked and a suggestion is given to tilt the camera along the direction of the exit direction. From the face detection module, the coordinates of face in the frame are obtained. Midpoint of the face is calculated from these coordinates. Based on the location of the midpoint in the frame, the location of face in the frame is identified and saved and updated consistently. When the driver goes out of frame, the suggestion for direction of tilt is given as this saved direction.

3.2.5 DEVIATION RATIO

Deviation ratio is the ratio of number of frames in which face was not detected to the number of frames in which face was detected. It gives a measure of how effective the system and the setup is for a specific driver. A value close to 0 indicates that the system is very effective and a value close to 1 indicates that the system is ineffective.

3.3 SLEEP PREDICTION MODULE

The main objective of drowsiness detection module is to predict the next day's timing during which the driver may fall asleep based on his/her drowsiness history. This module is driver specific. Figure 3.5 shows the flow diagram for sleep prediction module. Whenever drowsiness is detected for the

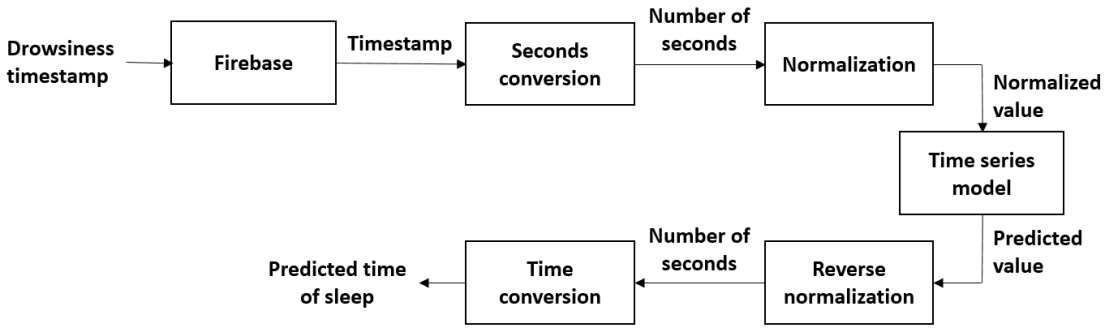

Figure 3.5: Sleep prediction

driver, the timestamp is logged in firebase. The database of logs act as training data for the model to learn the driver's specific drowsiness pattern. A single timestamp entry from the database is first converted from hours:minutes:seconds to number of seconds. It is then normalized so that the value falls between 0 and 1. 7 latest values in the database undergo this process and act as input to the model. The model predicts the next value in sequence to the input. The predicted value, which lies between 0 and 1 is then reverse normalized to obtain the number of seconds, which is converted into hours:minutes:seconds format to give the predicted time of sleep for the next day.

3.4 IMAGE SIMILARITY ANALYSIS MODULE

The main objective of drowsiness detection module is to predict whether the driver is going to fall asleep in a few seconds before he actually does based on analysing how his/her facial features look before falling asleep

and check whether the current frame is similar to those features. This module is driver specific. Figure 3.6 shows the flow diagram for image similarity analysis module.

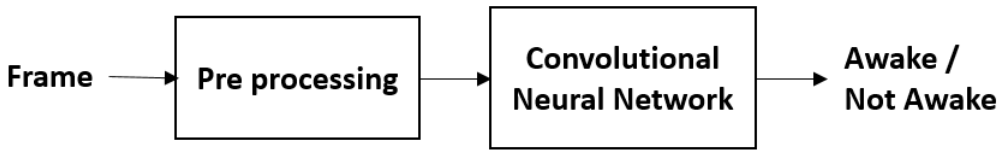

Figure 3.6: Image similarity analysis

There are 2 classes which the model outputs - Awake and going to sleep. Images of the driver's faces few seconds before he/she fell asleep are taken as dataset for 'going to sleep' class. Images of the driver's faces in which he/she is wide awake are taken as dataset for 'awake' class. From the video stream, every 15th frame is given as input to this model and a prediction is made to check whether the driver is going to fall asleep in a few seconds or not. If the prediction is 'going to sleep', an alert is raised to forewarn the driver.

CHAPTER 4

IMPLEMENTATION

4.1 TOOLS USED

The following are the list of tools and libraries used in this proposed work.

4.1.1 LABELIMG

LabelImg is a graphical image annotation tool used to draw bounding boxes in images and obtain their coordinates. It is written in Python and uses Qt for its graphical interface. Annotations are saved as XML files in PASCAL VOC format, the format used by ImageNet. Besides, it also supports YOLO format. The GUI can be used to open images and draw bounding boxes manually.

4.1.2 OPENCV

OpenCV is an open-source computer vision and machine learning software library in Python. cvtcolor method in OpenCV converts an image from one color space to another. There are more than 150 color-space conversion methods available in OpenCV. imshow method in OpenCV displays an image in a window which automatically fits to the image size.

4.1.3 NUMPY

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

4.1.4 PIL

Python Imaging Library is a free and open-source additional library for the Python programming language that has support for opening, manipulating, and saving many different image file formats. PIL can convert images across different color formats, extract individual color channels, modify the pixel information etc.

4.1.5 DLIB

Dlib is an open source library implementing a variety of machine learning algorithms, including classification, regression, clustering, data transformation, and structured prediction. Dlib library offers several functions for computer vision tasks such as face recognition, training custom landmark/shape predictors, object detection, object tracking.

4.1.6 SCIKIT-LEARN

Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy. It also provides datasets.

4.1.7 KERAS API

Keras is a deep learning API written in Python, running on top of the machine learning platform TensorFlow. Keras is the high-level API of TensorFlow 2.0: an approchable, highly-productive interface for solving machine learning problems, with a focus on modern deep learning. It provides essential abstractions and building blocks for developing and shipping machine learning solutions with high iteration velocity. Keras offers functions to build layers of a convoluted neural network.

4.1.8 TENSORFLOW LIBRARY

TensorFlow is a free and open-source software library for machine learning. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. It offers a wide range of datasets and also pre trained models which can be used for transfer learning.

4.1.9 MATPLOTLIB

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications.

4.1.10 FIREBASE DATABASE

Firebase is a Backend-as-a-Service (Baas). It provides with a variety of tools and services such as Realtime database and many more. It is built on Google's infrastructure. Firebase is categorized as a NoSQL database program, which stores data in JSON-like documents.

4.1.11 RASPBERRY PI

The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python.

4.1.12 RASPBIAN OS

Raspbian OS is a free operating system based on Debian, optimised for the Raspberry Pi hardware. Raspberry Pi OS comes with over 35,000 packages precompiled software bundled in a nice format for easy installation on the Raspberry Pi.

4.2 FACE DETECTION

Face detection was implemented using HOG and SVM approach where SVM classifies the HOG feature descriptors of the image. Following are the formulae [5] used to extract the HOG features which computes the norm and gradient respectively.

$$N_G(x,y) = \sqrt{G_H(x,y)^2 + G_V(x,y)^2}$$

$$O_G(x,y) = atan\left(\frac{G_H(x,y)}{G_V(x,y)}\right)$$

Dlib provides a custom object detector based on hog+svm approach. The model was configured to run on 36 threads, 80 by 80 sliding window size. Since 13 GB RAM in Colab wasn't enough, this model was trained in an AWS EC2 instance with 72 GB RAM. The model outputs the coordinates of bounding

box around the face. Algorithm 4.1 walks through the steps involved in face detection.

Algorithm 4.1 Face detection

Input: Video stream from webcam
Output: Bounding box of face in the frame

1. Video stream has been converted into frames
2. For each frame
   (a) Luminance of the frame has been checked
   (b) If luminance is low, boosting of luminance has been done
   (c) Conversion of image to grayscale takes place
   (d) Subsequently face has been identified

4.3 FACIAL LANDMARKS DETECTION

Custom shape predictor model provided by Dlib was trained for facial landmarks detection. The model was configured to train as follows

- Cascade depth: 15
- Tree depth: 4
- Oversampling amount: 5
- Number of threads: 36
- nu: 0.1

This model was trained in an AWS EC2 instance with 72 GB RAM. The model outputs the coordinates of six points around left eye, right eye and mouth individually. Algorithm 4.2 walks through the steps involved in facial landmarks detection. Algorithm 4.3 walks through the steps involved in calculating eye and mouth aspect ratios.

Algorithm 4.2 Facial landmarks detection

Input: Frame and bounding box of face
Output: Coordinates of areas around eyes and mouth

1. Window has been opened to display the image
2. For each frame
   (a) Eyes and mouth landmarks have been detected
   (b) Contours were drawn around the landmarks
   (c) The frame has been displayed with contours in the window
3. Window has been destroyed

Algorithm 4.3 Eye and mouth aspect ratios

Input: Coordinates of areas around eyes and mouth
Output: Drowsiness detection

1. For each eye
   (a) Distance between horizontal points has been calculated as width
   (b) Distance between vertical points has been calculated as height
   (c) Aspect ratio has been calculated as summation of heights by width times two
   (d) If Aspect ratio above threshold(0.3)
      i. If number of consecutive frames greater than 48
         A. Alert has been raised
2. For the mouth
   (a) Distance between horizontal points has been calculated as width
   (b) Distance between vertical points has been calculated as height
   (c) Aspect ratio has been calculated as summation of heights by width times two
   (d) If Aspect ratio below threshold(0.5)
      i. If number of consecutive frames greater than 30
         A. Alert has been raised

4.4 SLEEP PREDICTION

Whenever drowsiness is detected, the timestamps are recorded and uploaded to the firebase database.The timestamps are then preprocessed as suitable input to the LSTM model. The model consists of 3 layers - 2 LSTM layers and a dense layer. The model has a total of 30,651 parameters. Training of the model was done on Google Colab. Algorithm 4.4 walks through the steps involved in making inference using LSTM model for sleep prediction.

Algorithm 4.4 Sleep prediction

Input: Timestamps of instances when drowsiness was detected
Output: Timestamps for the next day when the driver may fall asleep

1. Extract latest entry from firebase
2. Load the saved model
3. Data preprocessing:
   (a) Convert timestamp to number of seconds
   (b) Normalize the value so that it falls between 0 and 1
   (c) Add the value to the existing list of previous days' history
   (d) Take the latest 7 values including new value and convert into array
4. Process the input through the loaded model for prediction
5. Reverse the normalization and obtain number of seconds
6. Convert seconds to time, which is the predicted time during which the driver may fall asleep

---

4.5 IMAGE SIMILARITY ANALYSIS

From the video stream, every 15th frame is chosen as input.The model is trained with two classes of images indicating - going to fall asleep and awake. Augmentation steps such as random horizontal flip and rotation are applied to the training images so that the model can generalize in a better way. The CNN model consists of 51 layers and 2,773,913 trainable parameters. Optimizer used for the model is Adam and the loss function is binary crossentropy. Algorithm 4.5 walks through the steps involved in making inference using LSTM model for image similarity analysis.

Algorithm 4.5 Image similarity analysis

Input: Segmented frames from the camera's video stream.
Output: Binary output indicating whether the driver is going to fall asleep or not

1. Load the saved CNN model
2. Turn camera on and start video stream
3. Image preprocessing:
   (a) Extract frame from stream
   (b) Convert color scheme from bgr to rgb
   (c) Reshape into 224,224 sized image, convert into array
   (d) Normalize all pixel values
4. Process the input through the loaded model for prediction
5. If the prediction infers that the driver is going to sleep, raise alert

4.6 RASPBERRY PI IMPLEMENTATION

Algorithm 4.6 walks through the steps of implementation in Raspberry Pi after integration of all the modules.

Algorithm 4.6 Raspberry Pi implementation

Input: Segmented frames from the camera's video stream.
Output: Alerts for the driver.

1. Turn camera on and start video stream
2. Extract frame from stream
3. Luminance boosting
   (a) Check luminance of frame
   (b) Apply histogram equalization on the frame
4. Face detection
   (a) Infer coordinates of face using Face detection model
5. Out of frame inspection
   (a) Calculate midpoint of face using coordinates
   (b) Save location of midpoint corresponding to the entire frame
6. Landmarks detection
   (a) Infer coordinates of eyes and mouth using Landmark detection model
   (b) Calculate eye and mouth aspect ratios
   (c) If aspect ratios cross thresholds, raise alert and log to Firebase
7. If face is not detected, suggest camera tilt to the latest saved location
8. Update deviation ratio
9. Image similarity analysis
   (a) For every 15th frame
      i. Infer whether the current frame is awake or going to sleep using Image similarity analysis model
      ii. If prediction is going to sleep, raise alert

The model of Raspberry Pi used for this work is Raspberry Pi 4 with 2 GB RAM. Additional hardware connected with Pi and used are a USB Camera for capturing live video stream, Micro SD card of 16GB for booting Raspbian OS and a USB power source for the Pi. Inference and processing are done using the Pi without using external computational sources.

CHAPTER 5

RESULTS AND PERFORMANCE ANALYSIS

5.1 DROWSINESS DETECTION

Results for drowsiness detection module have been analysed separately for face detection and facial landmark detection.

5.2 FACE DETECTION

The following are the results of the models used for face detection

5.2.1 CUSTOM CNN MODEL

A combination of images from LFW and Helen datasets were used for this model. A total of 12912 images for training and 2140 images for testing were used. This model had a total of 21,673,828 parameters and was trained for 100 epochs. The Custom CNN model gives training accuracy of 87.37% and validation accuracy of 50.33% . Figure 5.1 illustrates that though training accuracy increased with epochs, validation accuracy was poor.

5.2.2 TRANSFER LEARNING - MOBILENET

A combination of images from LFW and Helen datasets were used for this model. A total of 12912 images for training and 2140 images for testing were used. This model had a total of 5,146,180 parameters and was trained for 100 epochs. The Mobilenet model gives training accuracy of 89.67% and

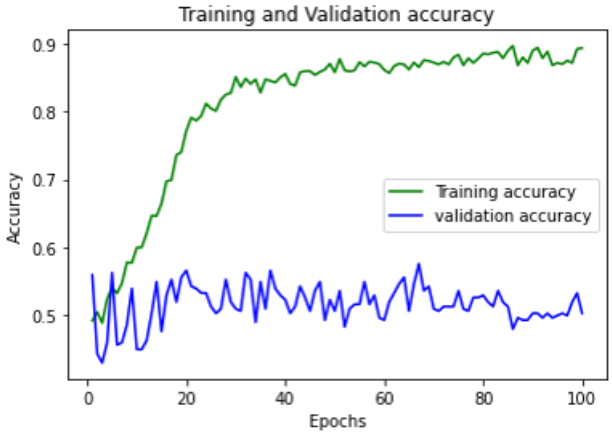

Figure 5.1: Custom CNN model

testing accuracy of 51.99% . Figure 5.2 illustrates that training accuracy had steady increase whereas validation accuracy didn't improve.

Figure 5.2: Mobilenet model

5.2.3 TRANSFER LEARNING - INCEPTION

A combination of images from LFW and Helen datasets were used for this model. A total of 12912 images for training and 2140 images for testing were used. This model had a total of 74,245,928 parameters and was trained for 100 epochs. The Inception model gives training accuracy of 84.67% and testing accuracy of 50.33%. Figure 5.3 illustrates that the model performed well for

training set and performed poor for the validation set.

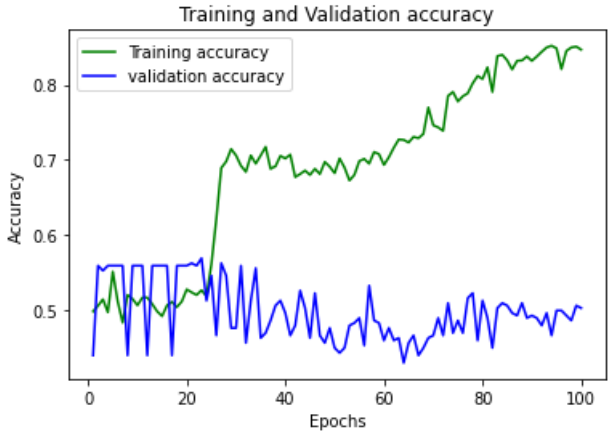

Figure 5.3: Inception model

5.2.4 CUSTOM HOG AND SVM

Scikit-image provides 13,233 images of face/not-face dataset was used as positive examples and 30,000 patches from 16 other classes which were used as negative examples. Scikit-learn provides hog feature extractor as well as SimpleSVM which were used in this model. The Custom HOG and SVM model had a best grid score of 98.8%.

5.2.5 YOLO

Custom dataset was collected for YOLO model. 615 images from 4 people were captured using webcam, preprocessed and used for training. LabelImg tool was used to manually annotate the 615 images and get the bounding boxes for each image in the format suitable for YOLO model. This model had a total of 7,255,092 parameters and was trained for 100 epochs. The YOLO model gives training accuracy of 97.7%. Figure 5.5 illustrates face detection using this model via a webcam. As illustrated by figure 5.4, YOLO model was able to achieve high precision as well as recall.

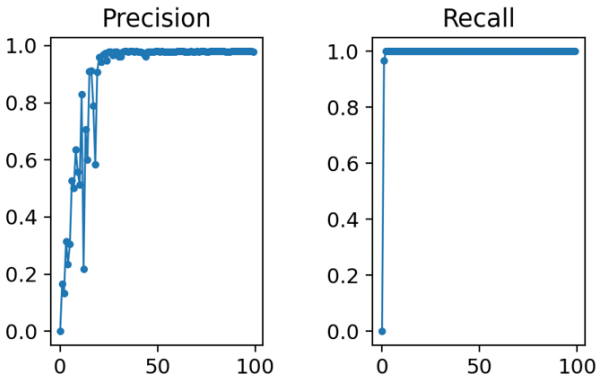

Figure 5.4: YOLO model

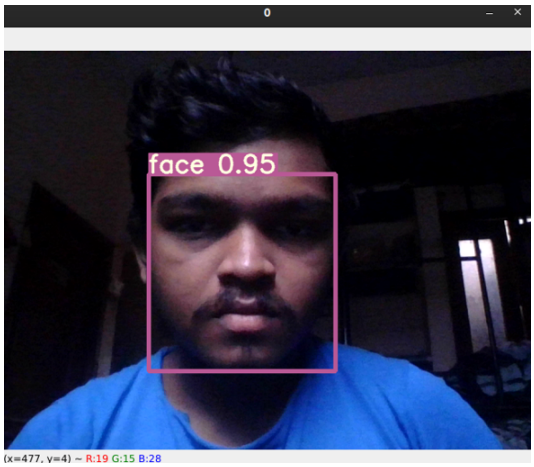

Figure 5.5: Face detection using YOLO

5.2.6 DLIB CUSTOM OBJECT DETECTOR

Dlib provides a 300-W dataset of face images with annotations which was used to train the custom object detector as a face detector. The Dlib custom object detector model gives training accuracy of 92.1%. Out of all the models, YOLO gave the highest accuracy when inferencing with real time data. But the average FPS of YOLO model was 2 to 3. For real time purposes, Dlib's custom object detector gave high FPS as well as high accuracy.

The accuracy of face detection by the object detector is measured

with the following confusion matrix.

	Predicted as face	Predicted as not a face
Actually contains face	2669	20
Actually does not contain face	4	5

Precision of the model is 99.8% and Recall is 99.2%

5.3 FACIAL LANDMARK DETECTION

300-W dataset of face images with annotations was used to train the custom shape predictor as a facial landmarks detector.The annotations of eyes and mouth were manually selected for the model to train on. The Dlib shape

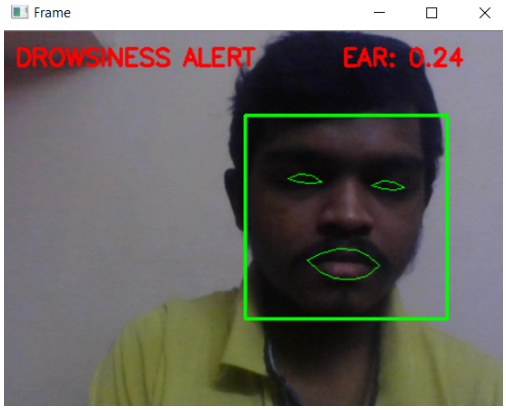

Figure 5.6: Drowsiness detection from eyes

predictor model gives training accuracy of 96.4%. If the aspect ratio of eyes goes below 30%, the driver is considered to be drowsy, as illustrated in Figure 5.6. If the mouth aspect ratio goes above 50%, the driver is considered to be

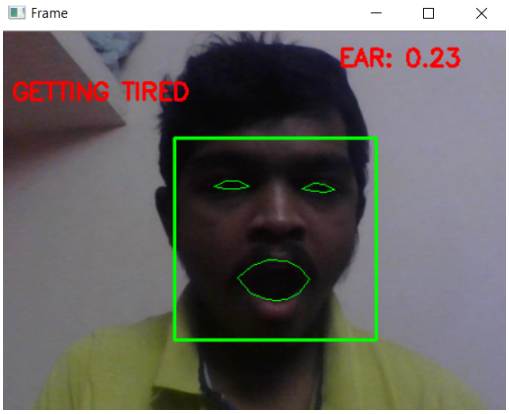

Figure 5.7: Yawn detection

getting drowsy, as illustrated in Figure 5.7. Dlib's custom object detector model and shape predictor model prove to be more suitable for real time detection of drowsiness with high FPS as well as high accuracy.

5.4 IMAGE SIMILARITY ANALYSIS

Custom dataset has been collected, captured using web camera. The dataset contains 401 images belonging to 2 classes - Awake and Going to sleep.

Figure 5.8: Going to sleep

Awake class has 124 images and Going to sleep class has 277 images. The model is a custom CNN model, trained on Google Colab. The model has

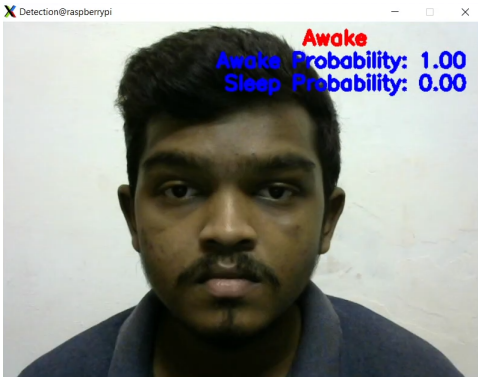

Figure 5.9: Awake

a training loss of 0.0027, validation loss of 0.0003 as shown in Figure 5.11, training accuracy of 100% and validation accuracy of 100% as shown in Figure 5.10. Figure 5.8 shows that the driver is going to fall asleep in a few seconds and Figure 5.9 shows that the driver is awake.

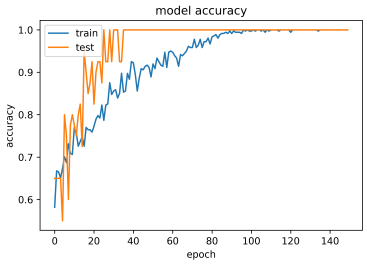

Figure 5.10: Training and validation accuracy of CNN model

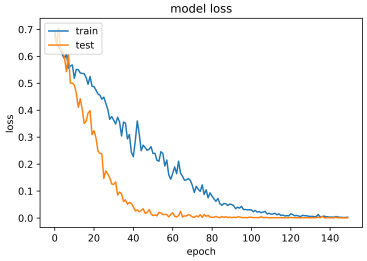

Figure 5.11: Training and validation loss of CNN model

5.5 SLEEP PREDICTION

Dataset used for this module is sleep_data from Kaggle. It consists of 887 rows and 8 columns. After preprocessing, the dataset for training the time series model contained 882 rows and 1 column - Time of sleep. Here LSTM is used as the time series model. Figure 5.12 shows the comparison between actual

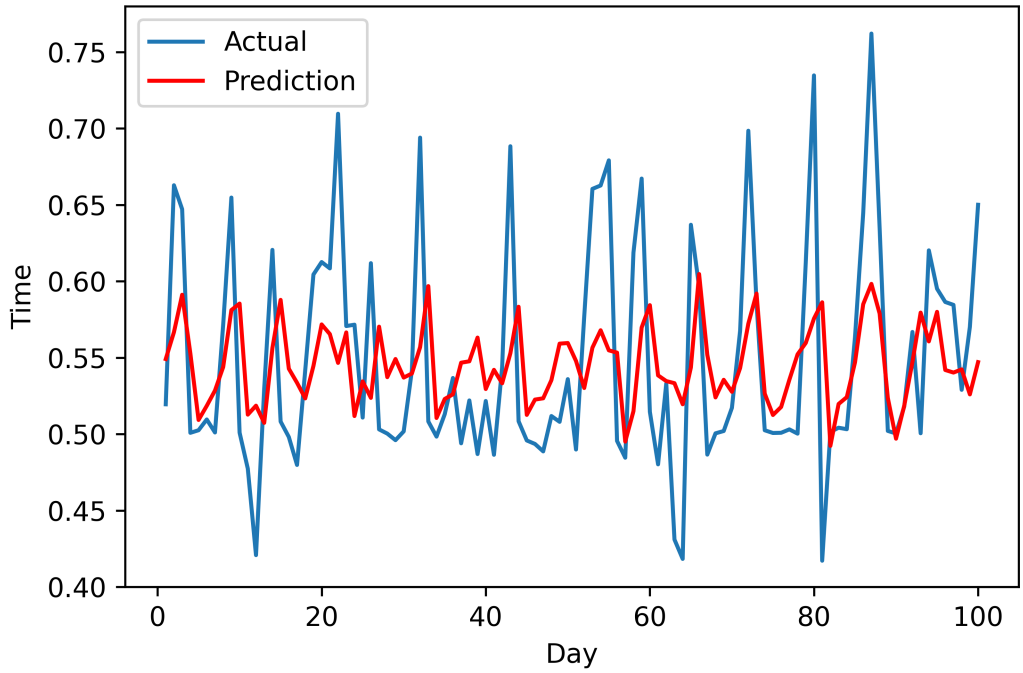

Figure 5.12: Actual vs predicted values

data versus the values which the LSTM model predicted. X-axis represent the consecutive days and the Y-axis represent time of sleep in normalized range. Each peak represents time in Y-axis to the corresponding day in X-axis. The model doesn't perform well for outliers. When seven days' values are given as input, the model predicts the 8th day's value. The loss type used when training the model was mean squared error and the loss is 0.005. Other models which were trained for sleep prediction are specified in the following subsections.

5.5.1 BIDIRECTIONAL LSTM

The Bidirectional LSTM model consists of 1 input layer, 2 BiLSTM layers and 1 dense layer. The model contains 132,737 parameters. The loss type used for this model is mean squared error and the loss is 0.0446.

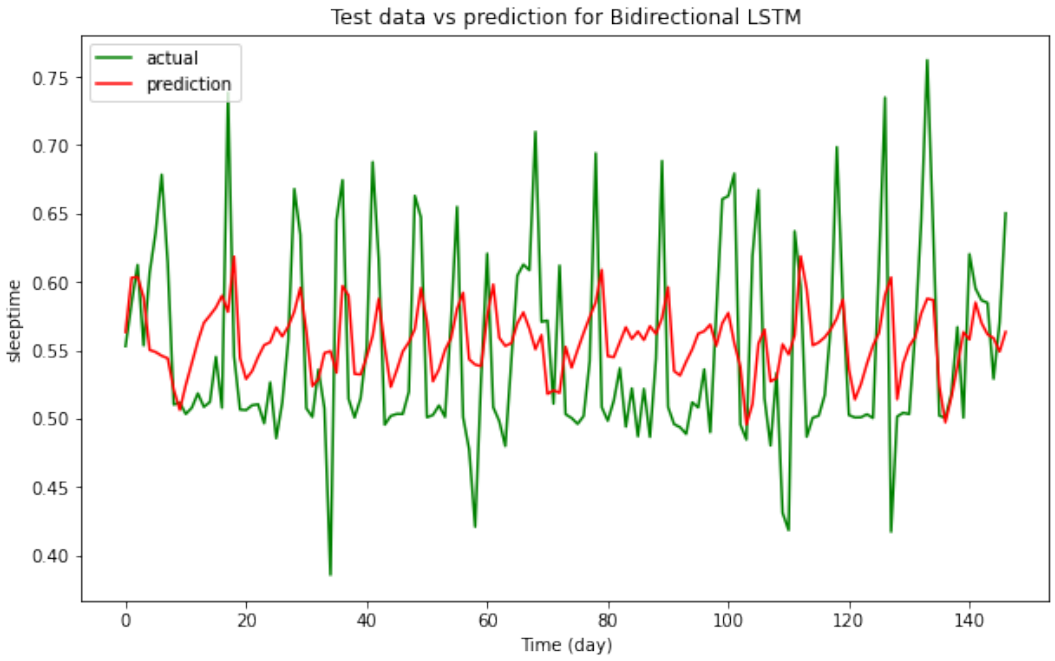

Figure 5.13: Actual vs predicted values of Bidirectional LSTM model

Figure 5.13 shows the comparison actual versus predicted value of the Bidirectional LSTM model. Here the X-axis represents consecutive days and Y-axis represents time of sleep in normalized range.

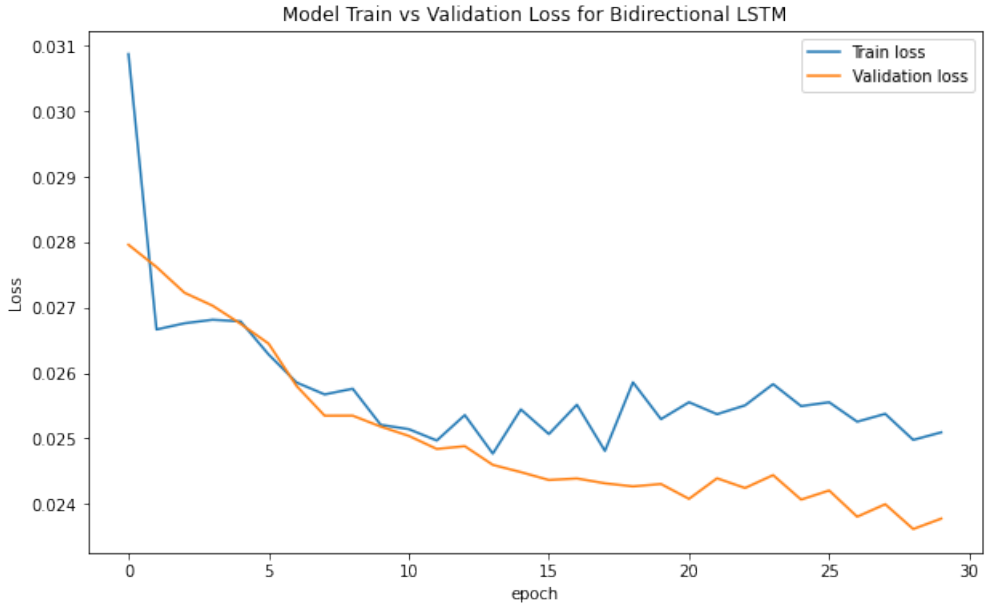

Figure 5.14: Training and validation loss of Bidirectional LSTM model

Figure 5.14 shows the training and validation loss of the model. In Figure 5.14 the X-axis represents the epoch and the Y-axis represents loss. By using this model we were able to predict when the driver will fall asleep the next day based on the previous 7 days.

5.5.2 GATED RECURRENT UNIT

The GRU model consists of 1 input layer, 2 GRU layers, 2 Dropout layers and 1 dense layer. The model contains 37,889 parameters. The loss type used for this model is mean squared error and the loss is 0.0293.

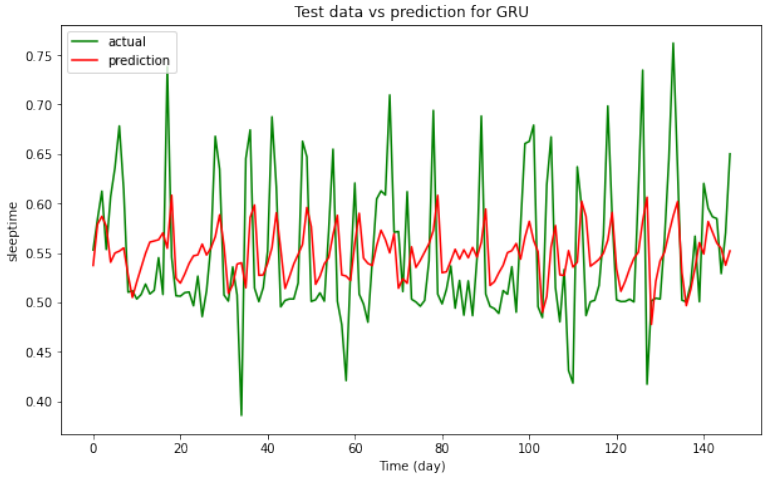

Figure 5.15: Actual vs predicted values of GRU model

Figure 5.15 shows the comparison actual versus predicted value of the GRU model. Here the X-axis represents consecutive days and Y-axis represents time of sleep in normalized range.

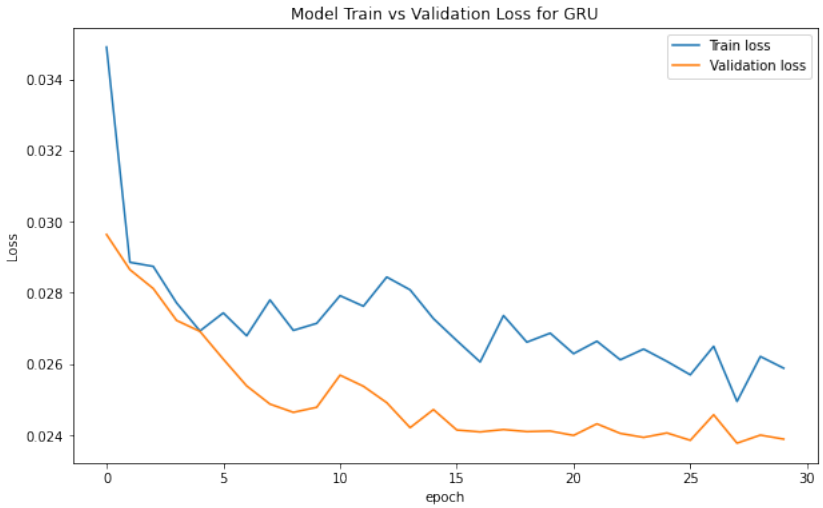

Figure 5.16: Training and validation loss of GRU model

Figure 5.16 shows the training and validation loss of the model. In Figure 5.16 the X-axis represents the epoch and the Y-axis represents loss.

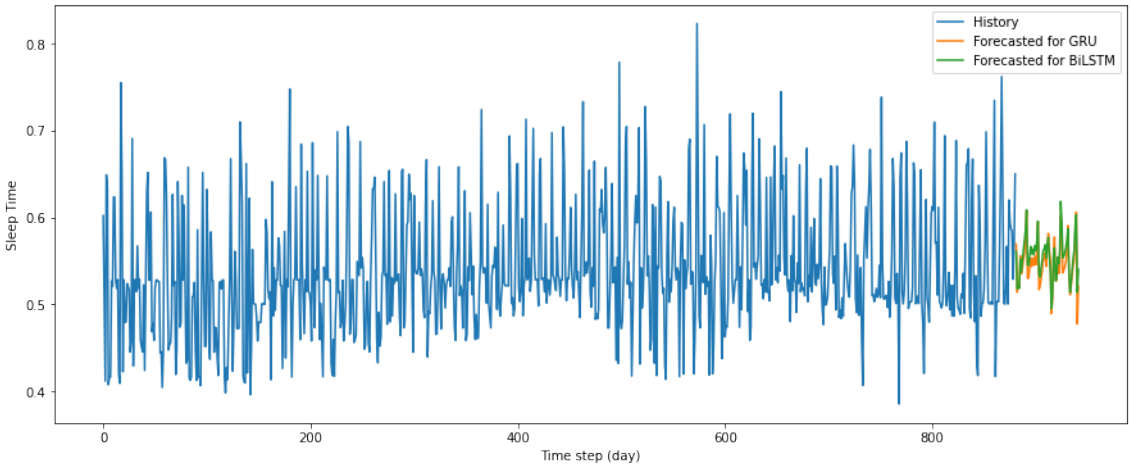

Figure 5.17: Future predictions by Bidirectional LSTM and GRU models

Figure 5.17 represents the future predictions by both Bidirectional LSTM and GRU models. By using the history of sleep data of a driver these two models predicted the sleep time of the driver for the next 30 days.

5.6 LUMINANCE BOOSTING

Figure 5.18 shows an image whose brightness is below the required threshold on the left and the same image after boosting brightness on the right.

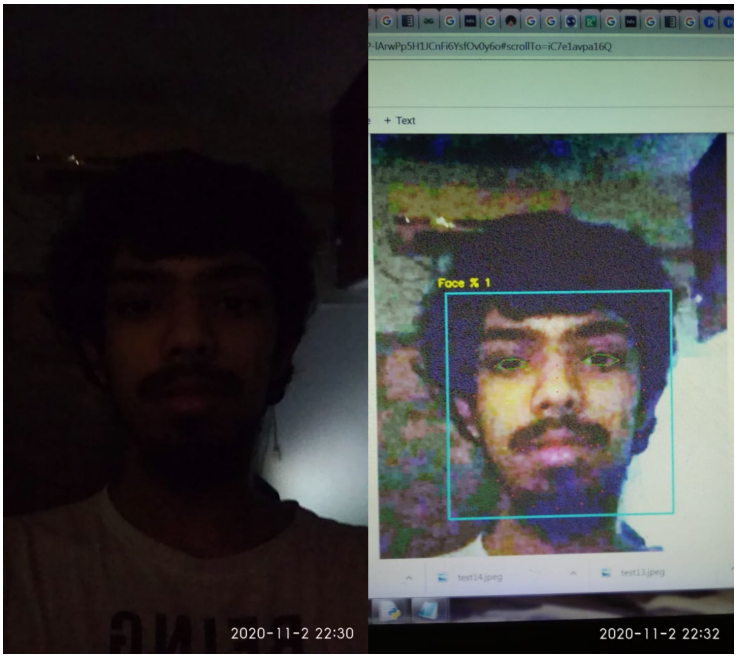

Figure 5.18: Image before and after processing

From Figure 5.18, it is observable that the face detection module was unable to detect a face from the image on the left and that the model was able to detect a face from the same image after brightness was increased. Luminance boosting is applied on a frame whenever it's brightness falls below a specific threshold, and the modified frame has proven to be a better input to the model than when boosting is not applied.

5.7 Out of frame inspection and deviation ratio

Figure 5.19 shows that the face has gone out of frame to the left side and gives suggestion that the camera can be tilted to the left.

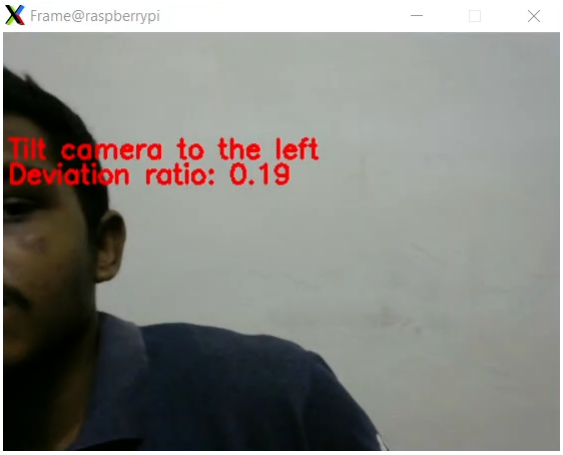

Figure 5.19: Image before processing

Figure 5.20 shows that the face has gone out of frame to the right side and gives suggestion that the camera can be tilted to the right.

Figure 5.20: Image post processing

In addition to left and right, top and bottom suggestions are also given if the face goes out of frame in the respective directions. Out of frame images and blurry images contribute to the deviation ratio which is used as a statistic for analytic purposes.

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this project, drowsiness of the driver has been detected using facial features extracted using HOG and SVM and was successfully implemented in Raspberry Pi 4. In addition to this, alerting the driver before he falls asleep using CNN has also implemented in Raspberry Pi 4. Then, by using the timestamps collected when the driver fell asleep previously, the time at which the driver may fall asleep the next day has been accurately predicted using LSTM, Bidirectional LSTM and GRU models. Because of the implementation of luminance boosting, the system will be able to perform well even in low light conditions. Using pixel coordinates, the movement of the driver has been captured effectively and using this, the system will indicate when the driver deviates from the camera and also gives accurate suggestions for tilting the camera to the appropriate direction to get the face inside camera's field of view.

The future work of this project involves implementing the system in a more powerful hardware module than Raspberry Pi 4 as in the current system the number of frames per second is very less. The accuracy of the current system can also be increased by experimenting with different models with more number of parameters which is limited by the current system's limited hardware resources.

REFERENCES

[1] P. Suja Suchitra and Shikha Tripathi. "Real-Time Emotion Recognition from Facial Images using Raspberry Pi II". In *Signal Processing and Integrated Networks (SPIN), 2016 3rd International Conference*, 2016.

[2] Nashwan Adnan Othman and Mehmet Umut Salur, Mehmet Karakose, and Ilhan Aydin. "An Embedded Real-Time Object Detection and Measurement of its Size". In *2018 International Conference on Artificial Intelligence and Data Processing (IDAP)*, 2018.

[3] Yi Sun , Xiaogang Wang, and Xiaoou Tang. "Deep Convolutional Network Cascade for Facial Point Detection". In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[4] Sreelekshmy Selvin , R. Vinayakumar, E.A. Gopalakrishna, Vijay Krishna Menon, and K.P. Soman. "Stock price prediction using Lstm, Rnn and Cnn-sliding window model". In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2017.

[5] R. Cerna and Lourdes. "Face Detection: Histogram of Oriented Gradients and Bag of Feature Method". volume 1, pages 523–638, 2013.

[6] C. Yashwanth and J. S. Kirar. "Driver's Drowsiness Detection". In *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*, pages 1622–1625, 2019.

[7] B. Alshaqaqi, A. S. Baquhaizel, M. E. Amine Ouis, M. Boumehed, A. Ouamri, and M. Keche. "Driver Drowsiness Detection System". In *2013 8th International Workshop on Systems, Signal Processing and their Applications*, pages 151–155, 2013.

[8] M. Khan, S. Chakraborty, R. Astya, and S. Khepra. "Face Detection And Recognition Using Opencv". In *2019 International Conference on Computing, Communication, and Intelligent Systems*, pages 116–119, 2019.

[9] C. Tang, Y. Feng, X. Yang, C. Zheng, and Y. Zhou. "Object Detection Based On Deep Learning". In *2017 4th International Conference on Information Science and Control Engineering*, pages 723–728, 2017.

[10] W. Deng and R. Wu. "Real-time Driver-drowsiness Detection System Using Facial Features", ieee access, vol. 7. pages 4344–4347, 2019.

Curiginal

Document Information

Analyzed document	Final report.pdf (D106365028)
Submitted	5/25/2021 2:43:00 PM
Submitted by	Abirami
Submitter email	abirami_mr@yahoo.com
Similarity	2%
Analysis address	abirami_mr.annauniv@analysis.urkund.com

Sources included in the report

W	URL: https://rc.library.uta.edu/uta-ir/bitstream/handle/10106/27411/NAYAK-THESIS-2018.pdf?sequence=1&isAllowed=y Fetched: 5/15/2020 4:58:34 AM	1
W	URL: https://uhcl-ir.tdl.org/bitstream/handle/10657.1/1413/WANG-MASTERSTHESIS-2018.pdf?sequence=1&isAllowed=y Fetched: 10/7/2019 10:12:50 AM	2
W	URL: http://www.ijitee.org/wp-content/uploads/papers/v9i9/I7033079920.pdf Fetched: 4/24/2021 7:56:35 PM	1
W	URL: https://www.ripublication.com/ijaer19/ijaerv14n4_01.pdf Fetched: 7/6/2020 10:55:20 AM	1
W	URL: https://www.ijrem.in/Downloads/Docs/IJREM%20-%20902.pdf Fetched: 10/30/2019 7:30:45 AM	1
W	URL: http://www.iieta.org/download/file/fid/22716 Fetched: 12/14/2020 10:30:20 AM	1

Curiginal

Entire Document

REAL TIME DRIVER DROWSINESS DETECTION A PROJECT REPORT Submitted by DANUSH S 2017115518 LALITH PRASAD S 2017115544 PREMCHANDER S 2017115572 submitted to the Faculty of INFORMATION AND COMMUNICATION ENGINEERING in partial fulfillment for the award of the degree of BACHELOR OF TECHNOLOGY in INFORMATION TECHNOLOGY DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY COLLEGE OF ENGINEERING, GUINDY ANNA UNIVERSITY CHENNAI 600 025 APRIL 2021

ii ANNA UNIVERSITY CHENNAI - 600 025 BONA FIDE CERTIFICATE Certified that this project report titled REAL TIME DRIVER DROWSINESS DETECTION is the bona fide work of DANUSH S (2017115518), LALITH PRASAD S (2017115544), PREMCHANDER S (2017115572) who carried out project work under my supervision. Certified further that to the best of my knowledge and belief, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or an award was conferred on an earlier occasion on this or any other candidate. PLACE: CHENNAI DATE: 07/04/2021 Dr. ABIRAMI MURUGAPPAN ASSISTANT PROFESSOR PROJECT GUIDE DEPARTMENT OF IST, CEG ANNA UNIVERSITY CHENNAI 600025 COUNTERSIGNED Dr. SASWATI MUKHERJEE HEAD OF THE DEPARTMENT DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY COLLEGE OF ENGINEERING, GUINDY ANNA UNIVERSITY CHENNAI 600025

iii ABSTRACT One of the most powerful and compelling types of Deep Learning is Computer vision. Because of advances in Artificial Intelligence, Computer Vision has been able to take great leaps in recent years and has been able to surpass humans in some tasks related to detecting and labeling objects. One such task is detecting drowsiness of a driver in a vehicle. The proposed project attempts to monitor the driver continuously, check if the driver gets drowsy and alert the driver. In addition to detecting drowsiness after the driver falls asleep, it also identifies key patterns in the driver's face using image similarity and predicts whether the driver is going to fall asleep in a few seconds. It can also predict when the driver will fall asleep in the future based on his/her drowsiness history. This project aims to prevent on-road accidents due to drowsiness by extracting key features from the face using Deep Learning. The whole system is implemented using raspberry pi 4 which can run as a standalone system without requiring additional hardware resources besides USB camera for live video capturing. This project involves three main modules. Drowsiness detection module consists of face detection and facial landmarks detection as sub modules. Face detection is implemented using HOG feature extraction and SVM classifier. Facial landmarks detection is implemented using an Ensemble of regression trees. The dataset used is from Dlib which provides a 300-W dataset of face images with annotations. Output of drowsiness detection module is whether or not the driver has fallen asleep. Image similarity analysis module is implemented using a Convolutional Neural Network. Custom dataset has been collected, captured using web camera. The dataset contains 401 images belonging to 2 classes - Awake and Going to sleep. Awake class has 124 images and Going to sleep class has 277 images. Output of image similarity analysis

iv module is whether or not the driver is going to fall asleep in a few seconds. Sleep prediction module is implemented using LSTM, Bidirectional LSTM and GRU models. Dataset used for this module is sleep data from Kaggle. It consists of 887 rows and 8 columns. Output of sleep prediction module is prediction of next day's timing at which the driver may fall asleep.

v [

vi ACKNOWLEDGEMENT We express our profound sense of gratitude to our guide Dr. ABIRAMI MURUGAPPAN, Assistant Professor, Department of Information Science and Technology, College of Engineering Guindy, Anna University, for her invaluable support, guidance and encouragement for the successful completion of this project. Her vision and spirit will always inspire and enlighten us. Our heartfelt thanks to Dr. SASWATI MUKHERJEE, Head, Department of Information Science and Technology, College of Engineering Guindy, Anna University, for the prompt and limitless help in providing the excellent computing facilities to do the project and to prepare the thesis. We express our sincere gratitude to our review committee members, Dr.S.SWAMYNATHAN, Professor, Dr.T.MALA, Associate Professor, Dr.K.VIDYA, Assistant Professor, Dr. D.NARASHIMAN, Teaching Fellow, Ms.T.SINDHU, Teaching Fellow, and Ms.G.MAHALAKSHMI, Teaching Fellow Department of Information Science and Technology, College of Engineering Guindy, Anna University, for their endless support and understanding spirit during our project presentations.We express our heartiest thanks to all other teaching and non-teaching staffs those who have helped us to do the project and to prepare the thesis. DANUSH S LALITH PRASAD S PREMCHANDER S

vii TABLE OF CONTENTS ABSTRACT iii ABSTRACT TAMIL v LIST OF FIGURES x LIST OF SYMBOLS AND ABBREVIATIONS xi 1 INTRODUCTION 1 1.1 COMPUTER VISION 1 1.2 MOTIVATION 2 1.3 NEED FOR PROPOSED WORK 2 1.4 PROBLEM STATEMENT 3 1.5 OBJECTIVES OF PROPOSED WORK 3 1.6 CHALLENGES OF THE PROPOSED WORK 3 1.7 ORGANISATION OF THE THESIS 4 2 LITERATURE SURVEY 6 2.1 REAL-TIME EMOTION RECOGNITION FROM FACIAL IMAGES USING RASPBERRY PI II 6 2.2 AN EMBEDDED REAL-TIME OBJECT DETECTION AND MEASUREMENT OF ITS SIZE 7 2.3 DEEP CONVOLUTIONAL NETWORK CASCADE FOR FACIAL POINT DETECTION 8 2.4 STOCK PRICE PREDICTION USING LSTM, RNN AND CNN-SLIDING WINDOW MODEL 8 3 SYSTEM ARCHITECTURE AND DESIGN 10 3.1 SYSTEM ARCHITECTURE 10 3.2 DROWSINESS DETECTION MODULE 11 3.2.1 LUMINANCE BOOSTING 11 3.2.2 FACE DETECTION 12 3.2.3 FACIAL LANDMARKS DETECTION 13 3.2.4 OUT OF FRAME INSPECTION 14 3.2.5 DEVIATION RATIO 14 3.3 SLEEP PREDICTION MODULE 14 3.4 IMAGE SIMILARITY ANALYSIS MODULE 15

viii 4 IMPLEMENTATION 17 4.1 TOOLS USED 17 4.1.1 LABELIMG 17 4.1.2 OPENCV 17 4.1.3 NUMPY 17 4.1.4 PIL 18 4.1.5 DLIB 18 4.1.6 SCIKIT-LEARN 18 4.1.7 KERAS API 18 4.1.8 TENSORFLOW LIBRARY 19 4.1.9 MATPLOTLIB 19 4.1.10FIREBASE DATABASE 19 4.1.11RASPBERRY PI 19 4.1.12RASPBIAN OS 20 4.2 FACE DETECTION 20 4.3 FACIAL LANDMARKS DETECTION 21 4.4 SLEEP PREDICTION 23 4.5 IMAGE SIMILARITY ANALYSIS 24 4.6 RASPBERRY PI IMPLEMENTATION 24 5 RESULTS AND PERFORMANCE ANALYSIS 27 5.1 DROWSINESS DETECTION 27 5.2 FACE DETECTION 27 5.2.1 CUSTOM CNN MODEL 27 5.2.2 TRANSFER LEARNING - MOBILENET 27 5.2.3 TRANSFER LEARNING - INCEPTION 28 5.2.4 CUSTOM HOG AND SVM 29 5.2.5 YOLO 29 5.2.6 DLIB CUSTOM OBJECT DETECTOR 29 5.3 FACIAL LANDMARK DETECTION 31 5.4 IMAGE SIMILARITY ANALYSIS 32 5.5 SLEEP PREDICTION 34 5.5.1 BIDIRECTIONAL LSTM 35 5.5.2 GATED RECURRENT UNIT 36 5.6 LUMINANCE BOOSTING 39 5.7 Out of frame inspection and deviation ratio 39

ix 6 CONCLUSION AND FUTURE WORK 41 REFERENCES 42

x LIST OF FIGURES 3.1 System Architecture 10 3.2 Luminance boosting 11 3.3 Face detection 12 3.4 Facial Landmark Detection 13 3.5 Sleep prediction 15 3.6 Image similarity analysis 16 5.1 Custom CNN model 28 5.2 Mobilenet model 28 5.3 Inception model 29 5.4 YOLO model 30 5.5 Face detection using YOLO 30 5.6 Drowsiness detection from eyes 31 5.7 Yawn detection 32 5.8 Going to sleep 32 5.9 Awake 33 5.10 Training and validation accuracy of CNN model 33 5.11 Training and validation loss of CNN model 33 5.12 Actual vs predicted values 34 5.13 Actual vs predicted values of Bidirectional LSTM model 35 5.14 Training and validation loss of Bidirectional LSTM model 36 5.15 Actual vs predicted values of GRU model 37 5.16 Training and validation loss of GRU model 37 5.17 Future predictions by Bidirectional LSTM and GRU models 38 5.18 Image before and after processing 39 5.19 Image before processing 40 5.20 Image post processing 40

xi LIST OF ABBREVIATIONS HOG Histogram of Oriented Gradients CNN Convolutional Neural Networks YOLO You Only Look Once PIL Python Imaging Library

1 CHAPTER 1 INTRODUCTION A driver who falls asleep at the wheel loses control of the vehicle, as a result of which accidents can occur. In order to prevent these devastating accidents, the state of drowsiness of the driver should be monitored. Accidents caused due to drivers getting drowsy can be prevented by using this driver drowsiness detection technology. 1.1

how computers can gain high-level understanding from digital images or videos. From the perspective of engineering, it seeks to understand and automate tasks that the human visual system can do. Computer vision tasks include methods for acquiring, processing, analyzing and understanding digital images, and extraction of high-dimensional data from the real world in order to produce numerical or symbolic information, e.g. in the forms of decisions. Understanding in this context means the transformation of visual images into descriptions of the world that make sense to thought processes and can elicit appropriate action. Consequently, computer vision is sometimes seen as a part of the artificial intelligence field or the computer science field in general. The advances on the computer vision field have been astounding. Accuracy rates for object identification and classification have gone from 50 percent to 99 percent in less than a decade and today's systems are more accurate than humans at quickly detecting and reacting to visual inputs.

2 1.2 MOTIVATION According to available statistical data, over 1.3 million people die each year on the road and 20 to 50 million people suffer non-fatal injuries due to road accidents. This finding shows how motorists ignore the importance of

taking adequate rest and end up endangering lives. In a country where road accidents claim nearly three lives every minute, this data clearly shows the need for a system which can detect drowsiness and prevent such accidents. The motivation for this work is that by implementing a cost effective system, such accidents can be prevented to a large extent. 1.3 NEED FOR PROPOSED WORK • The existing systems mostly depend on a cloud based environment. The proposed system is built using raspberry pi, which enables it to function completely in a local environment, which makes it suitable for real time use. • The proposed system uses computer vision technology in order to achieve the expected output so it does not require any sensors to be in contact with the driver, only a camera is enough. In contact approaches, drivers wear or touch some devices to get physiological parameters for detecting the level of their fatigue. • Most of the accidents due to fatigue happen during night time. This system aims to perform with high accuracy even in low light conditions.

3 1.4 PROBLEM STATEMENT Driver drowsiness leads to accidents and in some cases it has even proven to be fatal. By figuring out a solution which is cost effective as well as highly accurate, it can be taken into actual implementation and the number of accidents can be reduced. This project aims to alert the driver in case of drowsiness detection and also try to predict when the driver may fall asleep in the future. This project proposes to design a system to detect driver drowsiness using various machine learning and computer vision techniques. 1.5 OBJECTIVES OF PROPOSED WORK • The main objective is to first design a system to detect driver's drowsiness by continuously monitoring the driver and extracting the facial features. • Alert the driver on the detection of drowsiness by using a buzzer/alarm. • To handle low light images using image illumination techniques. • To inspect if driver goes out of frame and calculate the deviation ratio. • To predict the next day's timings when the driver may fall asleep using time series prediction. • Using image similarity to predict sleeping pattern and forewarn driver. • Hardware implementation of the application using Raspberry Pi with real time prediction and alerts.

4 1.6 CHALLENGES OF THE PROPOSED WORK The following challenges were faced during the implementation of this system • Implementation of the entire system in Raspberry Pi. • Capturing and Processing images under different light conditions. • As drivers' heights are different, the positions of their faces in the video are different. Then, when the driver is driving, his or her head may be moving and turning. Hence, tracking the trajectory of the head in real time is important once the position of the head changes. • Some drivers wear spectacles. The system should perform well and be capable of carrying out all the processes for those drivers too. • To accurately predict the time in which the driver may fall asleep in the future. • To keep track of the driver's sleep data in a real time database. • Experimenting with and selecting algorithms and models so that they run efficiently in Raspberry Pi. 1.7 ORGANISATION OF THE THESIS The project report is organised as follows, Chapter 2 discusses the various existing methods for the modules of the proposed system.

5 Chapter 3 discusses the various concepts used in the proposed system along with the overall design. Chapter 4 discusses the implementation details of the project. Chapter 5 discusses the results and performance analysis of the project. Chapter 6 concludes the report and discusses about the future work.

6 CHAPTER 2 LITERATURE SURVEY A literature survey related to the project work has been done and the observations from relevant papers are discussed in this chapter. The knowledge gained from these papers has helped us to create a better system for movie certification. 2.1 REAL-TIME EMOTION RECOGNITION FROM FACIAL IMAGES USING RASPBERRY PI II Suchitra et al., [1] proposed the system which detects real time emotion recognition from facial image.Emotions can understand by text,vocal,verbal and facial expressions.Facial expressions play big role in judging emotions of a person.It is found that limited work is done in field of real time emotion recognition using facial images.In the proposed method ,three steps face detection using Haar cascade,features extraction using Active shape Model(ASM),(26 facial points extracted) and Adaboost classifier for classification of five emotions anger, disgust, happiness, neutral and surprise.A Real-time emotion recognition system that recognizes basic emotions like anger, disgust, happiness, surprise and neutral using CMU MultiPIE database consisting 2D images with different illumination and poses.Active shape Model (ASM) for extracting facial points and AdaBoost classifier have been used for developing the emotion recognition software. The software system developed using this proposed method is deployed on Raspberry Pi II as it can be used with robots as the size of Raspberry Pi II is very small, light weighted and very less power supply is needed for it. The input image in real time is captured through webcam and fed to emotion recognition software as input. Emotion recognition

7 software is deployed in the Raspberry Pi II, which gives classified emotion as output. The recognized emotion is displayed in the monitor. The key challenge in this paper was that most of the work is done on the frontal view images of the

faces.More work need to be done on non-frontal images with different illumination conditions as in real time these global conditions are not uniform. 2.2

AN EMBEDDED REAL-TIME OBJECT DETECTION AND MEASUREMENT OF ITS SIZE Nashwan Adnan Othman, Mehmet Umut Salur [2] proposed an enhanced technique for detecting objects and computing their measurements in real time from video streams. an object measurement technique for real-time video by utilizing OpenCV libraries and includes the canny edge detection, dilation, and erosion algorithms. In the offered system, Computer Vision used to detect and measure objects. The system can detect and measure objects in a real time video. After the object has been detected by using canny edge detector, the size is obtained for each object by using OpenCV functions. The system consists of two parts which are object detection and object measurement. In the first part, raspberry pi camera used to achieve the frames. In the second part, computer vision module will be applied to the captured frames to determine the objects, then measure each object. The detected object of the current frame immediately will be processed to extract dimensions of objects.Firstly, preprocessing of the image is performed.The camera will capture a frame and the frame will convert to grayscale to increase quickness and accuracy. Objects are detected via canny edge detector algorithm. It is used to detect only one object or multiple objects. By the help of canny edge detector, the converted image will be processed. The canny edge algorithm scans the entire image. After that, execute dilation and erosion algorithm to close holes among edges in the edge frame.The system

8 applies four operations such as record frames, find edges, find objects, and measure size for each objects and the output is displayed. The key challenge in this paper was that the accuracy isn't perfect since this is due to the seeing angle and lens deformation.Calibration of camera and setting good width parameter can increase the accuracy. 2.3 DEEP CONVOLUTIONAL NETWORK CASCADE FOR FACIAL POINT DETECTION Yi Son et al., [3] proposed a new approach for estimation of the positions of facial keypoints with three-level carefully designed convolutional networks. At each level, the outputs of multiple networks are fused for robust and accurate estimation.The deep structures of convolutional networks,global high-level features are extracted over the whole face region at the initialization stage,which help to locate high accuracy keypoints. There are two folds of advantage for this. First, the texture context information over the entire face is utilized to locate each keypoint. Second, since the networks are trained to predict all the keypoints simultaneously, the geometric constraints among keypoints are implicitly encoded. The method therefore can avoid local minimum caused by ambiguity and data corruption in difficult image samples due to occlusions, large pose variations, and extreme lightings. The networks at the following two level-s are trained to locally refine initial predictions and their inputs are limited to small regions around the initial predictions. The key challenge in this paper was that the facial keypoint detection causes problem when n face images are taken with extreme poses, lightings, expressions, and occlusions. 2.4 STOCK PRICE PREDICTION USING LSTM, RNN AND CNN-SLIDING WINDOW MODEL

9 Sreelekshmy Selvin et al., [4] proposed a deep learning based formalization for stock price prediction. It is seen that, deep neural network architectures are capable of capturing hidden dynamics and are able to make predictions.For the proposed methodology CNN is identified as the best model. It uses the information given at a particular instant for prediction. Even though the other two models are used in many other time dependent data analysis, it is not out performing the CNN architecture in this case. This is due to the sudden changes that occurs in stock markets. The changes occuring in the stock market may not always be in a regular pattern or may not always follow the same cycle. Based on the companies and the sectors, the existence of the trends and the period of their existence will differ. The analysis of these type of trends and cycles will give more profit for the investors. Convolutional networks use convolution instead of general matrix multiplication in at least one of their layers.The motivation behind using these three models is to identify whether there is any long term dependency existing in the given data.This shows that, the pattern or dynamics identified by the model is common to other companies also. The key challenge in this paper was that the stock market is a highly dynamical system, the patterns and dynamics existing with in the system will not always be the same. This cause learning problems to LSTM and RNN architecture and hence the models fails to capture the dynamical changes accurately.

10 CHAPTER 3 SYSTEM ARCHITECTURE AND DESIGN 3.1 SYSTEM ARCHITECTURE Figure 3.1 shows the overall architecture of the system. The entire system is implemented to run in Raspberry Pi. A Camera is set up to monitor the driver's face. Figure 3.1: System Architecture

11 The video from the camera is first split into frames. Then, luminance boosting is applied on the frame. The frame then goes through face detection and landmark detection modules for drowsiness detection, sleep prediction module and image similarity analysis module for drowsiness prediction. The proposed system follows a loosely coupled architecture comprising of the following modules. • Drowsiness detection • Sleep prediction • Image similarity analysis 3.2 DROWSINESS DETECTION MODULE The main objective of drowsiness detection module is to detect whether the driver has become drowsy and raise alerts. Other objectives include luminance boosting, out of frame inspection and deviation ratio calculation. 3.2.1 LUMINANCE BOOSTING The frames which go through the computer vision methods are first preprocessed. Figure 3.2 gives the flow diagram of this module. First, the Figure 3.2: Luminance boosting

12 brightness of the frame is checked by converting the color space of the frame from bgr to ycbcr. The brightness of the frame is then calculated from the y channel which represents the luma characteristic of the image. If low brightness is detected, the b, g and r channels of the frame are equalized and the brightness of the frame is increased to provide as better input to the computer vision modules. 3.2.2 FACE DETECTION Video stream from the camera is converted into frames and face detection is done on each of the separated frame. Face detection consists of the following sub modules: • Image scaling • HOG feature extraction • Classification of HOG features Figure 3.3: Face detection Figure 3.3 gives the flow diagram of this module. The faces captured by various cameras will differ in size inside the frames. If the input to the classifier doesn't contain the face in correct proportions, it will be misclassified. So, the frame goes through Image scaling process, which will reduce the image size by scaling down repeatedly.

13 HOG counts occurrences of gradient orientation in localized portions of an image. The result of applying HOG to the frame will yield HOG feature vectors in the same shape and dimensions as the frame. Since images of the same class will have similar HOG features, classification can be done on HOG features. The feature vectors will be split into 8 by 8 parts. Sliding window is applied which covers each part and the windows will then go through a classifier to decide whether that window contains a face or not. The output of this module is a bounding box specifying the top left coordinates, width and height of window in which face is detected. 3.2.3 FACIAL LANDMARKS DETECTION Figure 3.4 shows the steps involved in detecting the facial keypoints - eyes and mouth in the frame. Figure 3.4: Facial Landmark Detection

14 The frame and bounding box from the face detection module are taken as inputs into the ensemble of regression trees, which will predict the locations of eyes and mouth inside the bounding box. A total of 18 landmark points are obtained as coordinate values around the eyes and mouth. Based on the coordinates of eyes and mouth, their respective aspect ratios are calculated. If the aspect ratio of the eyes are higher than a specific threshold, it indicates that the eyes of the driver are closing and the driver can be alerted. If the aspect ratio of the mouth is lower than a specific threshold, it indicates that the driver is yawning and a small indication can be given to the driver that he/she is getting drowsy. 3.2.4 OUT OF FRAME INSPECTION Whenever the driver goes out of frame, the location of his/her face in the previous frame is tracked and a suggestion is given to tilt the camera along the direction of the exit direction. From the face detection module, the coordinates of face in the frame are obtained. Midpoint of the face is calculated from these coordinates. Based on the location of the midpoint in the frame, the location of face in the frame is identified and saved and updated consistently. When the driver goes out of frame, the suggestion for direction of tilt is given as this saved direction. 3.2.5 DEVIATION RATIO Deviation ratio is the ratio of number of frames in which face was not detected to the number of frames in which face was detected. It gives a measure of how effective the system and the setup is for a specific driver. A value close to 0 indicates that the system is very effective and a value close to 1 indicates that the system is ineffective.

15 3.3 SLEEP PREDICTION MODULE The main objective of drowsiness detection module is to predict the next day's timing during which the driver may fall asleep based on his/her drowsiness history. This module is driver specific. Figure 3.5 shows the flow diagram for sleep prediction module. Whenever drowsiness is detected for the Figure 3.5: Sleep prediction driver, the timestamp is logged in firebase. The database of logs act as training data for the model to learn the driver's specific drowsiness pattern. A single timestamp entry from the database is first converted from hours:minutes:seconds to number of seconds. It is then normalized so that the value falls between 0 and 1. 7 latest values in the database undergo this process and act as input to the model. The model predicts the next value in sequence to the input. The predicted value, which lies between 0 and 1 is then reverse normalized to obtain the number of seconds, which is converted into hours:minutes:seconds format to give the predicted time of sleep for the next day. 3.4 IMAGE SIMILARITY ANALYSIS MODULE The main objective of drowsiness detection module is to predict whether the driver is going to fall asleep in a few seconds before he actually does based on analysing how his/her facial features look before falling asleep

16 and check whether the current frame is similar to those features. This module is driver specific. Figure 3.6 shows the flow diagram for image similarity analysis module. Figure 3.6: Image similarity analysis There are 2 classes which the

model outputs - Awake and going to sleep. Images of the driver's faces few seconds before he/she fell asleep are taken as dataset for 'going to sleep' class. Images of the driver's faces in which he/she is wide awake are taken as dataset for 'awake' class. From the video stream, every 15th frame is given as input to this model and a prediction is made to check whether the driver is going to fall asleep in a few seconds or not. If the prediction is 'going to sleep', an alert is raised to forewarn the driver.

17 CHAPTER 4 IMPLEMENTATION 4.1 TOOLS USED The following are the list of tools and libraries used in this proposed work. 4.1.1 LABELIMG LabelImg is a graphical image annotation tool used to draw bounding boxes in images and obtain their coordinates. It is written in Python and uses Qt for its graphical interface. Annotations are saved as XML files in PASCAL VOC format, the format used by ImageNet. Besides, it also supports YOLO format. The GUI can be used to open images and draw bounding boxes manually. 4.1.2 OPENCV

Python. cvtcolor method in OpenCV converts an image from one color space to another. There are more than 150 color-space conversion methods available in OpenCV. imshow method in OpenCV displays an image in a window which automatically fits to the image size. 4.1.3

PIL Python Imaging Library is a free and open-source additional library for the Python programming language that has support for opening, manipulating, and saving many different image file formats. PIL can convert images across different color formats, extract individual color channels, modify the pixel information etc. 4.1.5 DLIB Dlib is an open source library implementing a variety of machine learning algorithms, including classification, regression, clustering, data transformation, and structured prediction. Dlib library offers several functions for computer vision tasks such as face recognition, training custom landmark/shape predictors, object detection, object tracking. 4.1.6 SCIKIT-LEARN Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy. It also provides datasets.

19 4.1.7 KERAS API Keras is a deep learning API written in Python, running on top of the machine learning platform TensorFlow. Keras is the high-level API of TensorFlow 2.0: an approchable, highly-productive interface for solving machine learning problems, with a focus on modern deep learning. It provides essential abstractions and building blocks for developing and shipping machine learning solutions with high iteration velocity. Keras offers functions to build layers of a convoluted neural network. 4.1.8 TENSORFLOW LIBRARY

machine learning. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. It offers a wide range of datasets and also pre trained models which can be used for transfer learning. 4.1.9 MATPLOTLIB Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications. 4.1.10 FIREBASE DATABASE Firebase is a Backend-as-a-Service (Baas). It provides with a variety of tools and services such as Realtime database and many more. It is built on Google's infrastructure. Firebase is categorized as a NoSQL database program, which stores data in JSON-like documents.

20 4.1.11 RASPBERRY PI The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore

computing, and to learn how to program in languages like Scratch and Python. 4.1.12 RASPBIAN OS Raspbian OS is a free operating system based on Debian, optimised for the Raspberry Pi hardware. Raspberry Pi OS comes with over 35,000 packages precompiled software bundled in a nice format for easy installation on the Raspberry Pi. 4.2 FACE DETECTION Face detection was implemented using HOG and SVM approach where SVM classifies the HOG feature descriptors of the image. Following are the formulae [5] used to extract the HOG features which computes the norm and gradient respectively. Dlib provides a custom object detector based on hog+svm approach. The model was configured to run on 36 threads, 80 by 80 sliding window size. Since 13 GB RAM in Colab wasn't enough, this model was trained in an AWS EC2 instance with 72 GB RAM. The model outputs the coordinates of bounding

21 box around the face. Algorithm 4.1 walks through the steps involved in face detection. Algorithm 4.1 Face detection Input: Video stream from webcam Output: Bounding box of face in the frame 1. Video stream has been converted into frames 2. For each frame (a) Luminance of the frame has been checked (b) If luminance is low, boosting of luminance has been done (c) Conversion of image to grayscale takes place (d) Subsequently face has been identified 4.3 FACIAL LANDMARKS DETECTION Custom shape predictor model provided by Dlib was trained for facial landmarks detection. The model was configured to train as follows • Cascade depth: 15 • Tree depth: 4 • Oversampling amount: 5 • Number of threads: 36 • nu: 0.1 This model was trained in an AWS EC2 instance with 72 GB RAM. The model outputs the coordinates of six points around left eye, right eye and mouth individually. Algorithm 4.2 walks through the steps involved in facial landmarks detection. Algorithm 4.3 walks through the steps involved in calculating eye and mouth aspect ratios.

22 Algorithm 4.2 Facial landmarks detection Input: Frame and bounding box of face Output: Coordinates of areas around eyes and mouth 1. Window has been opened to display the image 2. For each frame (a) Eyes and mouth landmarks have been detected (b) Contours were drawn around the landmarks (c) The frame has been displayed with contours in the window 3. Window has been destroyed Algorithm 4.3 Eye and mouth aspect ratios Input: Coordinates of areas around eyes and mouth Output: Drowsiness detection 1. For each eye (a) Distance between horizontal points has been calculated as width (b) Distance between vertical points has been calculated as height (c) Aspect ratio has been calculated as summation of heights by width times two (d) If Aspect ratio above threshold(0.3) i. If number of consecutive frames greater than 48 A. Alert has been raised 2. For the mouth (a) Distance between horizontal points has been calculated as width (b) Distance between vertical points has been calculated as height (c) Aspect ratio has been calculated as summation of heights by width times two (d) If Aspect ratio below threshold(0.5) i. If number of consecutive frames greater than 30 A. Alert has been raised

23 4.4 SLEEP PREDICTION Whenever drowsiness is detected, the timestamps are recorded and uploaded to the firebase database.The timestamps are then preprocessed as suitable input to the LSTM model. The model consists of 3 layers - 2 LSTM layers and a dense layer. The model has a total of 30,651 parameters. Training of the model was done on Google Colab. Algorithm 4.4 walks through the steps involved in making inference using LSTM model for sleep prediction. Algorithm 4.4 Sleep prediction Input: Timestamps of instances when drowsiness was detected Output: Timestamps for the next day when the driver may fall asleep 1. Extract latest entry from firebase 2. Load the saved model 3. Data preprocessing: (a) Convert timestamp to number of seconds (b) Normalize the value so that it falls between 0 and 1 (c) Add the value to the existing list of previous days' history (d) Take the latest 7 values including new value and convert into array 4. Process the input through the loaded model for prediction 5. Reverse the normalization and obtain number of seconds 6. Convert seconds to time, which is the predicted time during which the driver may fall asleep

24 4.5 IMAGE SIMILARITY ANALYSIS From the video stream, every 15th frame is chosen as input.The model is trained with two classes of images indicating - going to fall asleep and awake. Augmentation steps such as random horizontal flip and rotation are applied to the training images so that the model can generalize in a better way. The CNN model consists of 51 layers and 2,773,913 trainable parameters. Optimizer used for the model is Adam and the loss function is binary crossentropy. Algorithm 4.5 walks through the steps involved in making inference using LSTM model for image similarity analysis. Algorithm 4.5 Image similarity analysis Input: Segmented frames from the camera's video stream. Output: Binary output indicating whether the driver is going to fall asleep or not 1. Load the saved CNN model 2. Turn camera on and start video stream 3. Image preprocessing: (a) Extract frame from stream (b) Convert color scheme from bgr to rgb (c) Reshape into 224,224 sized image, convert into array (d) Normalize all pixel values 4. Process the input through the loaded model for prediction 5. If the prediction infers that the driver is going to sleep, raise alert 4.6 RASPBERRY PI IMPLEMENTATION Algorithm 4.6 walks through the steps of implementation in Raspberry Pi after integration of all the modules.

25 Algorithm 4.6 Raspberry Pi implementation Input: Segmented frames from the camera's video stream. Output: Alerts for the driver. 1. Turn camera on and start video stream 2. Extract frame from stream 3. Luminance boosting (a) Check

luminance of frame (b) Apply histogram equalization on the frame 4. Face detection (a) Infer coordinates of face using Face detection model 5. Out of frame inspection (a) Calculate midpoint of face using coordinates (b) Save location of midpoint corresponding to the entire frame 6. Landmarks detection (a) Infer coordinates of eyes and mouth using Landmark detection model (b) Calculate eye and mouth aspect ratios (c) If aspect ratios cross thresholds, raise alert and log to Firebase 7. If face is not detected, suggest camera tilt to the latest saved location 8. Update deviation ratio 9. Image similarity analysis (a) For every 15th frame i. Infer whether the current frame is awake or going to sleep using Image similarity analysis model ii. If prediction is going to sleep, raise alert

26 The model of Raspberry Pi used for this work is Raspberry Pi 4 with 2 GB RAM. Additional hardware connected with Pi and used are a USB Camera for capturing live video stream, Micro SD card of 16GB for booting Raspbian OS and a USB power source for the Pi. Inference and processing are done using the Pi without using external computational sources.

27 CHAPTER 5 RESULTS AND PERFORMANCE ANALYSIS 5.1 DROWSINESS DETECTION Results for drowsiness detection module have been analysed separately for face detection and facial landmark detection. 5.2 FACE DETECTION The following are the results of the models used for face detection 5.2.1 CUSTOM CNN MODEL A combination of images from LFW and Helen datasets were used for this model. A total of 12912 images for training and 2140 images for testing were used. This model had a total of 21,673,828 parameters and was trained for 100 epochs. The Custom CNN model gives training accuracy of 87.37% and validation accuracy of 50.33% . Figure 5.1 illustrates that though training accuracy increased with epochs, validation accuracy was poor. 5.2.2 TRANSFER LEARNING - MOBILENET A combination of images from LFW and Helen datasets were used for this model. A total of 12912 images for training and 2140 images for testing were used. This model had a total of 5,146,180 parameters and was trained for 100 epochs. The Mobilenet model gives training accuracy of 89.67% and

28 Figure 5.1: Custom CNN model testing accuracy of 51.99% . Figure 5.2 illustrates that training accuracy had steady increase whereas validation accuracy didn't improve. Figure 5.2: Mobilenet model 5.2.3 TRANSFER LEARNING - INCEPTION A combination of images from LFW and Helen datasets were used for this model. A total of 12912 images for training and 2140 images for testing were used. This model had a total of 74,245,928 parameters and was trained for 100 epochs. The Inception model gives training accuracy of 84.67% and testing accuracy of 50.33%. Figure 5.3 illustrates that the model performed well for

29 training set and performed poor for the validation set. Figure 5.3: Inception model 5.2.4 CUSTOM HOG AND SVM Scikit-image provides 13,233 images of face/not-face dataset was used as positive examples and 30,000 patches from 16 other classes which were used as negative examples. Scikit-learn provides hog feature extractor as well as SimpleSVM which were used in this model. The Custom HOG and SVM model had a best grid score of 98.8%. 5.2.5 YOLO Custom dataset was collected for YOLO model. 615 images from 4 people were captured using webcam, preprocessed and used for training. LabelImg tool was used to manually annotate the 615 images and get the bounding boxes for each image in the format suitable for YOLO model. This model had a total of 7,255,092 parameters and was trained for 100 epochs. The YOLO model gives training accuracy of 97.7%. Figure 5.5 illustrates face detection using this model via a webcam. As illustrated by figure 5.4, YOLO model was able to achieve high precision as well as recall.

30 Figure 5.4: YOLO model Figure 5.5: Face detection using YOLO 5.2.6 DLIB CUSTOM OBJECT DETECTOR Dlib provides a 300-W dataset of face images with annotations which was used to train the custom object detector as a face detector. The Dlib custom object detector model gives training accuracy of 92.1%. Out of all the models, YOLO gave the highest accuracy when inferencing with real time data. But the average FPS of YOLO model was 2 to 3. For real time purposes, Dlib's custom object detector gave high FPS as well as high accuracy. The accuracy of face detection by the object detector is measured

31 with the following confusion matrix. Predicted as face Predicted as not a face Actually contains face 2669 20 Actually does not contain face 4 5 Precision of the model is 99.8% and Recall is 99.2% 5.3 FACIAL LANDMARK DETECTION 300-W dataset of face images with annotations was used to train the custom shape predictor as a facial landmarks detector.The annotations of eyes and mouth were manually selected for the model to train on. The Dlib shape Figure 5.6: Drowsiness detection from eyes predictor model gives training accuracy of 96.4%. If the aspect ratio of eyes goes below 30%, the driver is considered to be drowsy, as illustrated in Figure 5.6. If the mouth aspect ratio goes above 50%, the driver is considered to be

32 Figure 5.7: Yawn detection getting drowsy, as illustrated in Figure 5.7. Dlib's custom object detector model and shape predictor model prove to be more suitable for real time detection of drowsiness with high FPS as well as high accuracy. 5.4 IMAGE SIMILARITY ANALYSIS Custom dataset has been collected, captured using web camera. The dataset contains

401 images belonging to 2 classes - Awake and Going to sleep. Figure 5.8: Going to sleep Awake class has 124 images and Going to sleep class has 277 images. The model is a custom CNN model, trained on Google Colab. The model has

33 Figure 5.9: Awake a training loss of 0.0027, validation loss of 0.0003 as shown in Figure 5.11, training accuracy of 100% and validation accuracy of 100% as shown in Figure 5.10. Figure 5.8 shows that the driver is going to fall asleep in a few seconds and Figure 5.9 shows that the driver is awake. Figure 5.10: Training and validation accuracy of CNN model Figure 5.11: Training and validation loss of CNN model

34 5.5 SLEEP PREDICTION Dataset used for this module is sleep data from Kaggle. It consists of 887 rows and 8 columns. After preprocessing, the dataset for training the time series model contained 882 rows and 1 column - Time of sleep. Here LSTM is used as the time series model. Figure 5.12 shows the comparison between actual Figure 5.12: Actual vs predicted values data versus the values which the LSTM model predicted. X-axis represent the consecutive days and the Y-axis represent time of sleep in normalized range. Each peak represents time in Y-axis to the corresponding day in X-axis. The model doesn't perform well for outliers. When seven days' values are given as input, the model predicts the 8th day's value. The loss type used when training the model was mean squared error and the loss is 0.005. Other models which were trained for sleep prediction are specified in the following subsections.

35 5.5.1 BIDIRECTIONAL LSTM The Bidirectional LSTM model consists of 1 input layer, 2 BiLSTM layers and 1 dense layer. The model contains 132,737 parameters. The loss type used for this model is mean squared error and the loss is 0.0446. Figure 5.13: Actual vs predicted values of Bidirectional LSTM model Figure 5.13 shows the comparison actual versus predicted value of the Bidirectional LSTM model. Here the X-axis represents consecutive days and Y-axis represents time of sleep in normalized range.

36 Figure 5.14: Training and validation loss of Bidirectional LSTM model Figure 5.14 shows the training and validation loss of the model. In Figure 5.14 the X-axis represents the epoch and the Y-axis represents loss. By using this model we were able to predict when the driver will fall asleep the next day based on the previous 7 days. 5.5.2 GATED RECURRENT UNIT The GRU model consists of 1 input layer, 2 GRU layers, 2 Dropout layers and 1 dense layer. The model contains 37,889 parameters. The loss type used for this model is mean squared error and the loss is 0.0293.

37 Figure 5.15: Actual vs predicted values of GRU model Figure 5.15 shows the comparison actual versus predicted value of the GRU model. Here the X-axis represents consecutive days and Y-axis represents time of sleep in normalized range. Figure 5.16: Training and validation loss of GRU model Figure 5.16 shows the training and validation loss of the model. In Figure 5.16 the X-axis represents the epoch and the Y-axis represents loss.

38 Figure 5.17: Future predictions by Bidirectional LSTM and GRU models Figure 5.17 represents the future predictions by both Bidirectional LSTM and GRU models. By using the history of sleep data of a driver these two models predicted the sleep time of the driver for the next 30 days.

39 5.6 LUMINANCE BOOSTING Figure 5.18 shows an image whose brightness is below the required threshold on the left and the same image after boosting brightness on the right. Figure 5.18: Image before and after processing From Figure 5.18, it is observable that the face detection module was unable to detect a face from the image on the left and that the model was able to detect a face from the same image after brightness was increased. Luminance boosting is applied on a frame whenever it's brightness falls below a specific threshold, and the modified frame has proven to be a better input to the model than when boosting is not applied. 5.7 Out of frame inspection and deviation ratio Figure 5.19 shows that the face has gone out of frame to the left side and gives suggestion that the camera can be tilted to the left.

40 Figure 5.19: Image before processing Figure 5.20 shows that the face has gone out of frame to the right side and gives suggestion that the camera can be tilted to the right. Figure 5.20: Image post processing In addition to left and right, top and bottom suggestions are also given if the face goes out of frame in the respective directions. Out of frame images and blurry images contribute to the deviation ratio which is used as a statistic for analytic purposes.

41 CHAPTER 6 CONCLUSION AND FUTURE WORK In this project, drowsiness of the driver has been detected using facial features extracted using HOG and SVM and was successfully implemented in Raspberry Pi 4. In addition to this, alerting the driver before he falls asleep using CNN has also implemented in Raspberry Pi 4. Then, by using the timestamps collected when the driver fell asleep previously, the time at which the driver may fall asleep the next day has been accurately predicted using LSTM, Bidirectional LSTM and GRU models. Because of the implementation of luminance boosting, the system will be able to perform well even in low light conditions. Using pixel coordinates, the movement of the driver has been captured effectively and using this, the system will indicate when the driver deviates from the camera and also gives accurate suggestions for tilting the camera to the appropriate direction to get the face inside camera's field

of view. The future work of this project involves implementing the system in a more powerful hardware module than Raspberry Pi 4 as in the current system the number of frames per second is very less. The accuracy of the current system can also be increased by experimenting with different models with more number of parameters which is limited by the current system's limited hardware resources.

42 REFERENCES [1] P. Suja Suchitra and Shikha Tripathi. "Real-Time Emotion Recognition from Facial Images using Raspberry Pi II". In Signal Processing and Integrated Networks (SPIN), 2016 3rd International Conference, 2016. [2] Nashwan Adnan Othman and Mehmet Umut Salur, Mehmet Karakose, and Ilhan Aydin. "An Embedded Real-Time Object Detection and Measurement of its Size". In 2018 International Conference on Artificial Intelligence and Data Processing (IDAP), 2018. [3] Yi Sun , Xiaogang Wang, and Xiaoou Tang. "Deep Convolutional Network Cascade for Facial Point Detection". In 2013 IEEE Conference on Computer Vision and Pattern Recognition, 2013. [4] Sreelekshmy Selvin , R. Vinayakumar, E.A. Gopalakrishna, Vijay Krishna Menon, and K.P. Soman. "Stock price prediction using Lstm, Rnn and Cnn-sliding window model". In 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2017. [5] R. Cerna and Lourdes. "Face Detection: Histogram of Oriented Gradients and Bag of Feature Method". volume 1, pages 523–638, 2013. [6] C. Yashwanth and J. S. Kirar. "Driver's Drowsiness Detection". In TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON), pages 1622–1625, 2019. [7]

In 2013 8

pages 151–155, 2013. [8] M. Khan, S. Chakraborty, R. Astya, and S. Khepra. "Face Detection And Recognition Using Opencv". In 2019 International Conference on Computing, Communication, and Intelligent Systems, pages 116–119, 2019. [9] C. Tang, Y. Feng, X. Yang, C. Zheng, and Y. Zhou. "Object Detection Based On Deep Learning". In 2017 4th International Conference on Information Science and Control Engineering, pages 723–728, 2017. [10] W. Deng and R. Wu. "Real-time Driver-drowsiness Detection System Using Facial Features", ieee access, vol. 7. pages 4344–4347, 2019.

Curiginal

Hit and source - focused comparison, Side by Side

Submitted text As student entered the text in the submitted document.

Matching text As the text appears in the source.

1/7	SUBMITTED TEXT	11 WORDS	87%	MATCHING TEXT	11 WORDS

COMPUTER VISION Computer vision is an interdisciplinary scientific field that deals with

Computer Vision Computer Vision is an interdisciplinary field that deals with

W https://rc.library.uta.edu/uta-ir/bitstream/handle/10106/27411/NAYAK-THESIS-2018.pdf?sequence=1&i ...

2/7	SUBMITTED TEXT	22 WORDS	78%	MATCHING TEXT	22 WORDS

faces.More work need to be done on non-frontal images with different illumination conditions as in real time these global conditions are not uniform. 2.2

faces. Future work needs to be focused on non-frontal facial images with different illumination conditions as in real-time processing these global conditions are not uniform. 56

W https://uhcl-ir.tdl.org/bitstream/handle/10657.1/1413/WANG-MASTERSTHESIS-2018.pdf?sequence=1&isAl ...

3/7	SUBMITTED TEXT	12 WORDS	100%	MATCHING TEXT	12 WORDS

OpenCV is an open-source computer vision and machine learning software library in

OpenCV is an open source computer vision and machine learning software library. In

W https://uhcl-ir.tdl.org/bitstream/handle/10657.1/1413/WANG-MASTERSTHESIS-2018.pdf?sequence=1&isAl ...

6/7	SUBMITTED TEXT	31 WORDS	100%	MATCHING TEXT	31 WORDS

NUMPY 18 NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. 4.1.4

Numpy - NumPy is a library for the Python programming language, adding support for large, multi- dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays ?

W https://www.ijrem.in/Downloads/Docs/IJREM%20-%20902.pdf

4/7	SUBMITTED TEXT	10 WORDS	100%	MATCHING TEXT	10 WORDS

TensorFlow is a free and open-source software library for

TensorFlow is a free and open-source software library for

W http://www.ijitee.org/wp-content/uploads/papers/v9i9/I7033079920.pdf

Curiginal

5/7	SUBMITTED TEXT	20 WORDS	100%	MATCHING TEXT	20 WORDS

B. Alshaqaqi, A. S. Baquhaizel, M. E. Amine Ouis, M. Boumehed, A. Ouamri, and M. Keche. "Driver Drowsiness Detection System".

B. Alshaqaqi, A. S. Baquhaizel, M. E. Amine Ouis, M. Boumehed, A. Ouamri and M. Keche, "Driver drowsiness detection system," 2013 8

W https://www.ripublication.com/ijaer19/ijaerv14n4_01.pdf

7/7	SUBMITTED TEXT	10 WORDS	100%	MATCHING TEXT	10 WORDS

th International Workshop on Systems, Signal Processing and their Applications,

th International Workshop on Systems, Signal Processing and Their Applications (

W http://www.iieta.org/download/file/fid/22716