

Design and Implementation of Alerting Drowsiness Using Smart Device

Submitted in partial fulfillment of the requirements for
the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

BOLLU SAIVIJAYAKUMAR (Reg. No. 37110105)

BATTULA RATNAKARGUPTHA (Reg. No. 37110082)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade “A” by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI
SALAI, CHENNAI – 600 119**

MARCH – 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with “A” grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this project report is the bonafide work of BOLLU SAI VIJAYA KUMAR (Reg.No.37110105) and BATTULA RATNAKAR GUPTHA (Reg.No.37110082) who carried out the project entitled “Design and Implementation of Alerting Drowsiness using smart device” under my supervision from August 2019 to March 2020.

Internal Guide

Mrs. R. Sathya Bama Krishna, M.E.,

Head of the Department

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I BOLLU SAIVIJAYAKUMAR and BATTULA RATNAKARGUPTHA hereby declare that the Project Report entitled “Design and Implementation of Alerting Drowsiness using smart device” is done by me under the guidance of Mrs. R. Sathya Bama Krishna, M.E., Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE :

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D., Dean**, School of Computing, **Dr. S. Vigneswari, M.E., Ph.D.,** and **Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Mrs. R. Sathya Bama Krishna, M.E., Assistant Professor**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Drowsiness of the drivers is the main cause of accidents in the world. Due to lack of sleep or tiredness, drowsiness can occur while driving. The best way to avoid accidents caused by drivers' drowsiness is to detect drowsiness of the driver and warn him before he/she fall into sleep. The main idea behind this project is to develop a device which can detect fatigue of any human and can issue a timely warning. Drivers who do not take regular breaks when driving long distances run a high risk of becoming drowsy state which they often fail to recognize early enough.

This device will monitor the drivers' eye using a camera and by developing an algorithm we can detect symptoms of driver fatigue early enough to avoid the person from sleeping. So, this device will be helpful in detecting driver fatigue in advance and will give warning output in form of alarm. In this device we mainly use Machine Learning algorithm (Neural Networks) to build a model that detects the state of eyes whether they are open or closed. This model is loaded into a raspberry pi to which a camera module is attached. From this camera module we take input and gives it to the model. This device can be used in any vehicle to identify the drowsiness state of the drivers and warn them. In monitoring employees also we can use this device.

TABLE OF CONTENTS

	ABSTRACT	v
	LIST OF FIGURES	viii
CHAPTER NO.	TITLE	PAGE NO.
1	INTRODUCTION	1
2	LITERATURE SURVEY	3
	2.1 INTRODUCTION TO MACHINE LEARNING	3
	2.2 MACHINE LEARNING VS TRADITIONAL PROGRAMMING	3
	2.3 HOW DOES MACHINE LEARNING WORK	4
	2.3.1 LEARNING PHASE	5
	2.3.2 INFERRING PHASE	5
	2.4 INTRODUCTION TO DEEP LEARNING	6
	2.5 DEEP LEARNING PROCESS	8
	2.6 HAAR CASCADE CLASSIFIERS	9
	2.7. CONNECTING RASPBERRY PI WITH CAMERA	9
	2.8. BOOT STRAPPING THE PYTHON SCRIPT IN RASPBERRY PI	10
3	AIM AND SCOPE OF PROJECT	11
	3.1 AIM	11
	3.2 SCOPE	11
4	ALGORITHM AND METHOD	12
	4.1 ALGORITHM	12

	4.2 CODING	12
	4.2.1 IMPORTING MODULES	12
	4.2.2 DEVELOPING A MODEL	13
	4.2.3 LOADING HAAR CASCADE CLASSIFIERS	13
	4.3 CODE	13
	4.3.1 MODEL BUILDING CODE	13
5	RESULT AND DISCUSSION	18
	5.1 RESULT	18
	5.2 PERFORMANCE ANALYSIS	19
6	SUMMARY AND CONCLUSION	20
	REFERENCES	21

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO.
2.1	Traditional programming block diagram	3
2.2	Machine Learning block Diagram	4
2.3	simple block diagram about Learning Phase	5
2.4	Inferring Phase block Diagram	5
2.5	Diagram of a Neuron	6
2.6	communication between	6
2.7	Diagram of a artificial neuron	7
2.8	Diagram of a complete neural network	7
2.9	types of deep learning networks	8
2.10	python script bootstrapped into raspbian os	10
4.1	code snippet of importing modules in model construction	11
4.2	code snippet of importing modules in main program	11
4.3	code snippet of the model	12
5.1	showing alarm.wav file in the project folder	17
5.2	showing driver's face with open eyes	17
5.3	showing driver's face with closed eyes	18
5.4	figure showing test accuracy of the model	18

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Drowsiness of the drivers is the main cause of accidents in the world. No one knows the exact moment when sleep comes over their body. Falling asleep at the wheel is clearly dangerous and it affects your ability to drive safely. Drowsiness makes you less able to pay attention to the road. Drowsiness slows reaction time if you have to brake or steer suddenly. Drowsiness affects your ability to make good decisions on the road. The National Highway Traffic Safety Administration of America estimates that drowsy driving was responsible for 72,000 crashes, 44,000 injuries and 800 deaths in 2013. By Using Machine Learning which is a part of AI one can detect the drowsiness of the drivers and warn them by using any alarm sound or with something else. By doing this one can save many people from these accidents caused by drowsiness. Nowadays, more and more professions require long term concentration.

Drivers must keep a close eye on the road, so that they can react to sudden events immediately. Driver fatigue often becomes a direct cause of many traffic accidents. So, there is a need in developing such a system that will detect and notify a driver about his/her condition before causing any road accident. Now as internet become basic thing in daily life, technology has been developing day by day .There come many technologies into the world with wide range of real time applications examples like Internet of things, Machine Learning ,Artificial Intelligence. These technologies have wide range of applications in fields of medicine, security banking, real estate, shopping, e-commerce, and in some business applications. In now-a-days it is more important to do a survey or analysis before starting any business as it gives more important, some of these technologies has wide range in doing this, for example for starting any shopping mall or e-commerce one should have prior knowledge on what kind of products should be combined etc., this knowledge can be obtained from data mining. These mining is done through Machine Learning algorithms.

It is a necessary for each and every individual to know at least basics in this technologies as they play a very important role in the coming days and now. Machine Learning is the part of Artificial Intelligence is a scientific study that deals about algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. Internet of things is nothing but extending internet into physical components to communicate between themselves and to collect information and send it to cloud and process the results and send it to the user.

Block chain is a set of interconnected records which are protected using cryptography and which are unaltered. Augmented Reality is an interactive experience of a real-world environment where the objects that reside in the real-world are enhanced by computer-generated perceptual information, sometimes across multiple sensory modalities, including visual, auditory etc. Virtual Reality is an artificial environment that is created with software and presented to the user in such a way that the user suspends belief and accepts it as a real environment. All these are some of the presently evolving technologies. In this project we have mainly used Machine Learning .We will see in next section what this project is about in detail.

The beauty of this project it can run in any device or computer with minimum RAM it doesn't require any GPU's or any high performance system because model is previously compiled and stored in .h5 format. It can run on normal machine for example it can be used in way that take input from a normal camera that is live stream and we can monitor and warn the people continuously. Raspberry pi is nothing than a computer without any input and output devices and contains ports for connecting them and has small amount of ram generally 1GB,2GB,4GB for new raspberry pi 3.

We can also use this device to monitor employees in an organization and it can also be helpful for the students who are continuously spending their time on studies.

CHAPTER 2

LITERATURE SURVEY

2.1. INTRODUCTION TO MACHINE LEARNING

Machine Learning is a system that can learn from example through self-improvement and without being explicitly coded by programmer. The breakthrough comes with the idea that a machine can singularly learn from the data (i.e., example) to produce accurate results.

Machine learning combines data with statistical tools to predict an output. This output is then used by corporate to makes actionable insights. Machine learning is closely related to data mining and Bayesian predictive modeling. The machine receives data as input, use an algorithm to formulate answers.

A typical machine learning tasks are to provide a recommendation. For those who have a Netflix account, all recommendations of movies or series are based on the user's historical data. Tech companies are using unsupervised learning to improve the user experience with personalizing recommendation.

Machine learning is also used for a variety of task like fraud detection, predictive maintenance, portfolio optimization, automatize task and so on.

2.2. MACHINE LEARNING VS TRADITIONAL PROGRAMMING

Traditional programming differs significantly from machine learning. In traditional programming, a programmer code all the rules in consultation with an expert in the industry for which software is being developed. Each rule is based on a logical foundation; the machine will execute an output following the logical statement. When the system grows complex, more rules need to be written. It can quickly become unsustainable to maintain.

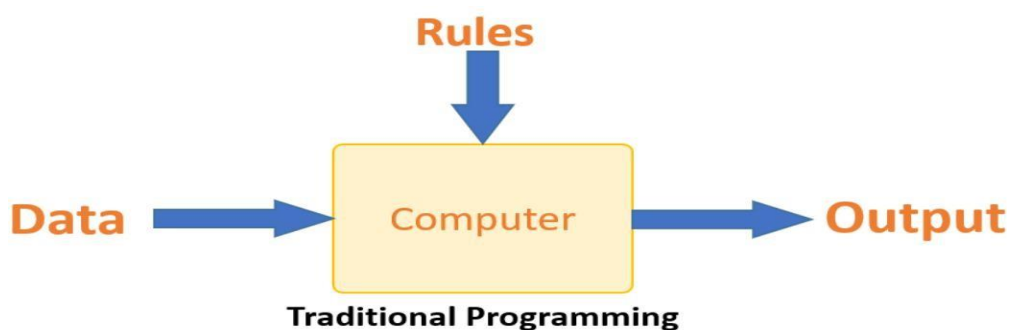


Fig 2.1 Traditional programming block diagram

Machine learning is supposed to overcome this issue. The machine learns how the input and output data are correlated and it writes a rule. The programmers do not need to write new rules each time there is new data. The algorithms adapt in response to new data and experiences to improve efficiency over time.

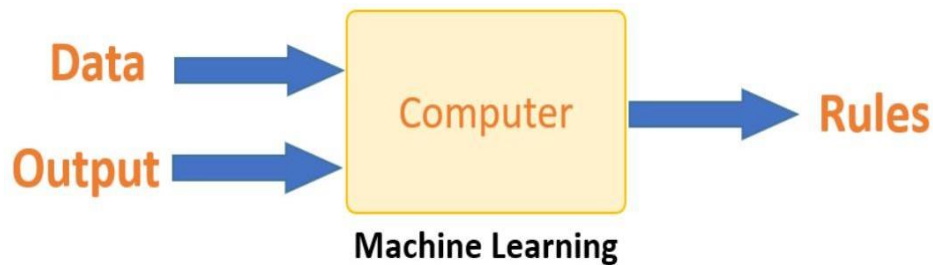


Fig 2.2 Machine Learning block Diagram

2.3. HOW DOES MACHINE LEARNING WORK

Machine learning is the brain where all the learning takes place. The way the machine learns is similar to the human being. Humans learn from experience. The more we know, the more easily we can predict. By analogy, when we face an unknown situation, the likelihood of success is lower than the known situation. Machines are trained the same. To make an accurate prediction, the machine sees an example. When we give the machine a similar example, it can figure out the outcome. However, like a human, if it's feed a previously unseen example, the machine has difficulties to predict.

The core objective of machine learning is the learning and inference. First of all, the machine learns through the discovery of patterns. This discovery is made thanks to the data. One crucial part of the data scientist is to choose carefully which data to provide to the machine. The list of attributes used to solve a problem is called a feature vector. You can think of a feature vector as a subset of data that is used to tackle a problem.

The machine uses some fancy algorithms to simplify the reality and transform this discovery into a model. Therefore, the learning stage is used to describe the data and summarize it into a model.

2.3.1. Learning phase

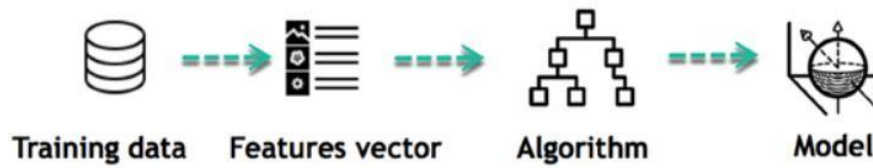


Fig 2.3 simple block diagram about Learning Phase

For instance, the machine is trying to understand the relationship between the wage of an individual and the likelihood to go to a fancy restaurant. It turns out the machine finds a positive relationship between wage and going to a high-end restaurant: This is the model.

2.3.2. Inferring Phase



Fig 2.4 Inferring Phase block Diagram

When the model is built, it is possible to test how powerful it is on never-seen-before data. The new data are transformed into a features vector, go through the model and give a prediction. This is all the beautiful part of machine learning. There is no need to update the rules or train again the model. You can use the model previously trained to make inference on new data.

2.4. INTRODUCTION TO DEEP LEARNING

Deep learning is a computer software that mimics the network of neurons in a brain. It is a subset of machine learning and is called deep learning because it makes use of deep neural networks. Deep learning algorithms are constructed with connected layers. A neural network mimics a neuron, which has dendrites, a nucleus, axon, and terminal axon. For a network, we need two neurons. These neurons transfer information via synapse between the dendrites of one and the terminal axon of another neurons.

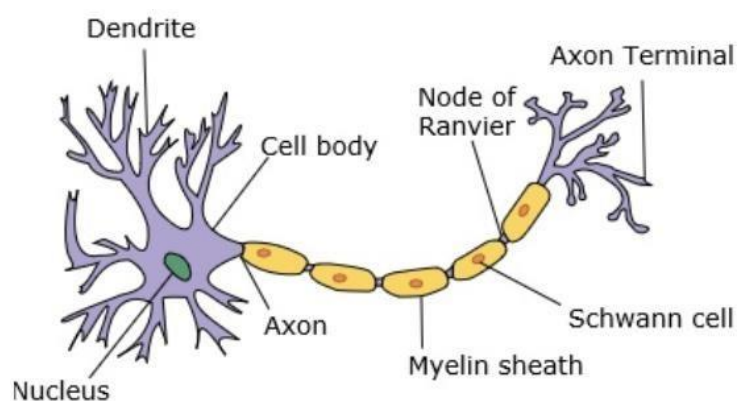


Fig 2.5 Diagram of a Neuron

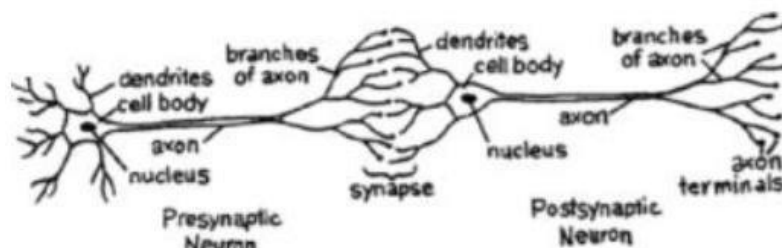


Fig 2.6 communication between

A probable model of an artificial neuron looks like this –

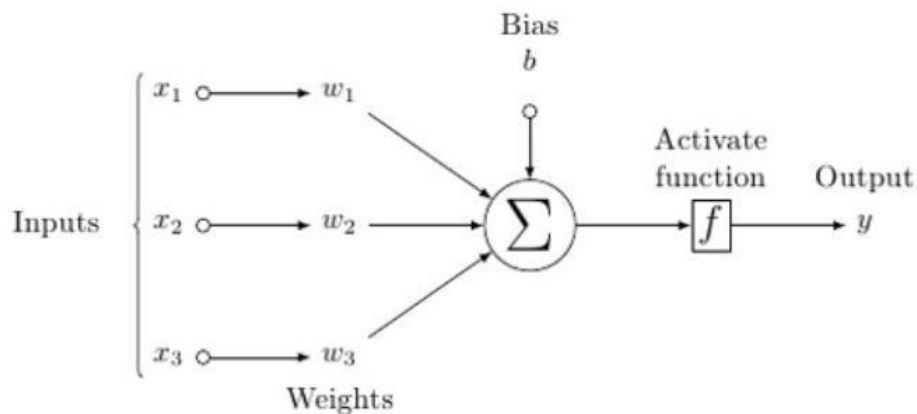


Fig 2.7 Diagram of a artificial neuron

A neural network will look like as shown below –

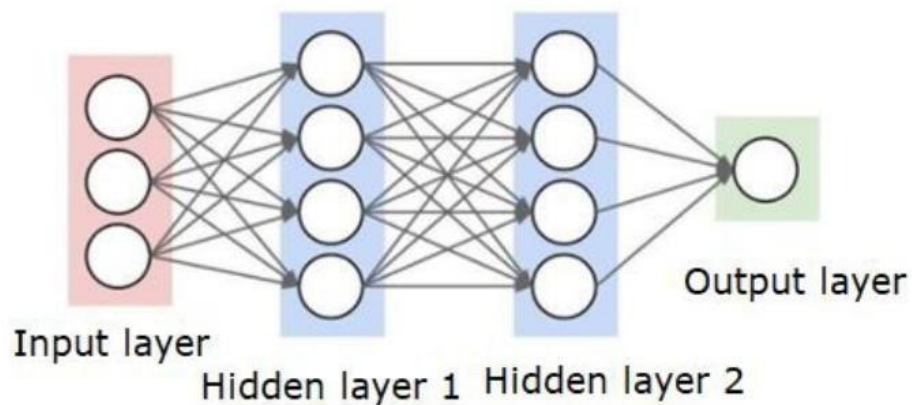


Fig 2.8 Diagram of a complete neural network

The circles are neurons or nodes, with their functions on the data and the lines/edges connecting them are the weights/information being passed along. Each column is a layer. The first layer of your data is the input layer. Then, all the layers between the input layer and the output layer are the hidden layers. If you have one or a few hidden layers, then you have a shallow neural network. If you have many hidden layers, then you have a deep neural network. In this model, you have input data, you weight it, and pass it through the function in the neuron that is called threshold function or activation function.

2.5. Deep Learning Process

A deep neural network provides state-of-the-art accuracy in many tasks, from object detection to speech recognition. They can learn automatically, without predefined knowledge explicitly coded by the programmers.

To grasp the idea of deep learning, imagine a family, with an infant and parents. The toddler points objects with his little finger and always says the word 'cat.' As its parents are concerned about his education, they keep telling him 'Yes, that is a cat' or 'No, that is not a cat.' The infant persists in pointing objects but becomes more accurate with 'cats.' The little kid, deep down, does not know why he can say it is a cat or not. He has just learned how to hierarchies complex features coming up with a cat by looking at the pet overall and continue to focus on details such as the tails or the nose before to make up his mind.

The learning occurs in two phases:

The first phase consists of applying a nonlinear transformation of the input and create a statistical model as output.

The second phase aims at improving the model with a mathematical method known as derivative.

The neural network repeats these two phases hundreds to thousands of time until it has reached a tolerable level of accuracy.

Types of Deep Learning Networks

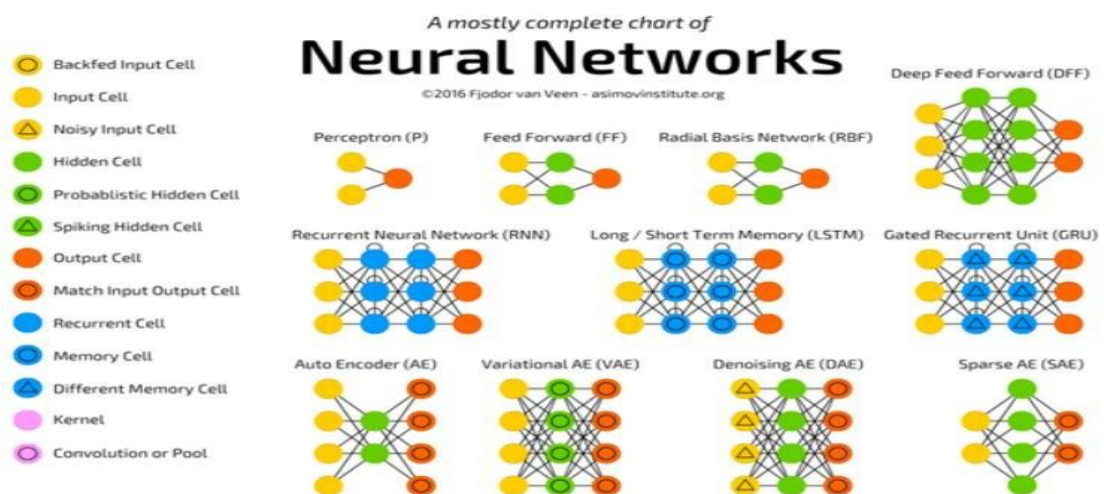


Fig 2.9 types of deep learning networks

2.6. Haar Cascade Classifiers:

Haar Cascade is a machine learning object detection algorithm used to identify objects in an image or video and based on the concept of features proposed by Paul Viola and Michael Jones in their paper "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001.

The algorithm has four stages:

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

It is well known for being able to detect faces and body parts in an image, but can be trained to identify almost any object. Initially, the algorithm needs a lot of positive images of faces and negative images without faces to train the classifier. Then we need to extract features from it.

We can download them from the below link:

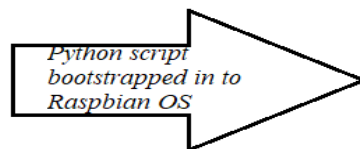
<https://github.com/opencv/opencv/tree/master/data/haarcascades>

2.7.Connecting Raspberry pi with camera:

- Raspberry Pi is initially connected to a camera module with the help of camera port allocated to Raspi
- Then initial testing of the camera module is done by changing the configurations in the Raspi in order to detect the image or a video
- As soon the testing is successful then we can make sure that the Raspi is ready for capturing a video or image.

2.8. Boot Strapping the Python script in Raspberry Pi:

- The python file which is used to detect the eyes and produce an alarm will be boot strapped in to the Raspberry Pi .
- In order to do this we need to set the path of the python script in the boot part of OS by using crontab
- Now the device need to be rebooted in order to reset the configurations .
- Hence as soon as the Raspi is powered on we get the access to camera and video can be capture



2.10 python script bootstrapped into raspbian os

CHAPTER 3

AIM AND SCOPE OF PROJECT

3.1. AIM:

The problem is to classify whether an eye is open or closed. If the eyes are closed based on the score and threshold value we have to warn the driver. Before going into main topic, first we discuss about some prerequisites and jump into the process. One should have a basic knowledge in Machine Learning and and an idea about Deep Learning.

Later we will see about those module and its installation in any system.

3.2. SCOPE:

People who's more likely to drive drowsy are drivers who don't get enough sleep, commercial drivers who work a lot, shift workers, drivers with untreated sleep disorders and drivers who use medications that make them sleepy. The idea behind the project is capture the face of the driver using a camera module and process it frame by frame.

While processing we detect the face of the driver using haarcascades classifier and we identify left and right eyes and we pass these eye pictures to Machine Learning model which is saved in the format of .h5 file. This Machine Learning model classifies a picture of eye whether it is closed or open. For each and every frame we classify the status of the both eyes and if they are closed we increment a score value. After the certain number of frames, if the score value is greater than a threshold value we can warn the driver by generating an alarm. The Machine Learning model which we have used is a neural network model which is trained with 4000 images of eyes (both open and closed). The dataset consists of around 5000 images for both testing and training. It consists of both closed and open eye images.

CHAPTER 4

ALGORITHM AND METHOD

4.1. Algorithm:

1. Finding the dataset and preprocessing the data.
2. Designing and developing a model
3. Training and testing the model
4. Capture the video and recognize the eyes
5. Classify the eyes and develop a score
6. Based on the score warn the diver.

4.2. Coding

And consider the above algorithm as a basic reference and we go on explaining each and every step.

Before going into coding part in any project make sure that you have installed all the libraries required and properly import them.

4.2.1. Importing modules

```
1 import os
2 from keras.preprocessing import image
3 import matplotlib.pyplot as plt
4 import numpy as np
5 from keras.utils.np_utils import to_categorical
6 import random,shutil
7 from keras.models import Sequential
8 from keras.layers import Dropout,Conv2D,Flatten,Dense, MaxPooling2D, BatchNormalization
9 from keras.models import load_model
```

Fig 4.1 code snippet of importing modules in model construction

```
1 import cv2
2 import os
3 from keras.models import load_model
4 import numpy as np
5 from pygame import mixer
6 import time
```

Fig 4.2 code snippet of importing modules in main program

Opencv(CV2) is used to capture the video.

Pygame module is used to beep the alarm.

Numpy module is used to perform calculations on the arrays.

Keras is used to develop a deep learning model.

4.2.2. Developing a model:

```
model = Sequential([
    Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(24,24,1)),
    MaxPooling2D(pool_size=(1,1)),
    Conv2D(32,(3,3),activation='relu'),
    MaxPooling2D(pool_size=(1,1)),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D(pool_size=(1,1)),
    Dropout(0.25),
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(2, activation='softmax')
])

model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])
model.fit_generator(train_batch, validation_data=valid_batch,epochs=15,steps_per_epoch=SPE ,validation_steps=VS)
model.save('models/rak.h5', overwrite=True)
```

Fig 4.3 code snippet of the model

The list of the layers we have used are convolutional layer,MaxPooling,Dropout and flatten layer.

4.2.3. Loading Haar Cascade classifiers:

```
face = cv2.CascadeClassifier('haar cascade files\haarcascade_frontalface_alt.xml')
leye = cv2.CascadeClassifier('haar cascade files\haarcascade_lefteye_2splits.xml')
reye = cv2.CascadeClassifier('haar cascade files\haarcascade_righteye_2splits.xml')
```

First, we have to download these files from the above given link (mentioned in 2nd chapter) and we have to load them using CascadeClassifier method available in OpenCV module.

4.3. Code

4.3.1 Model Building Code:

```
import os
from keras.preprocessing import image
import matplotlib.pyplot as plt
import numpy as np
from keras.utils.np_utils import to_categorical
import random,shutil
from keras.models import Sequential
```

```

from keras.layers import Dropout,Conv2D,Flatten,Dense, MaxPooling2D,
BatchNormalization
from keras.models import load_model
def generator(dir, gen=image.ImageDataGenerator(rescale=1./255),
shuffle=True,batch_size=1,target_size=(24,24),class_mode='categorical' ):
    return
gen.flow_from_directory(dir,batch_size=batch_size,shuffle=shuffle,color_mode='gr
ayscale',class_mode=class_mode,target_size=target_size)
BS= 32
TS=(24,24)
train_batch= generator('data/train',shuffle=True, batch_size=BS,target_size=TS)
valid_batch= generator('data/valid',shuffle=True, batch_size=BS,target_size=TS)
SPE= len(train_batch.classes)//BS
VS = len(valid_batch.classes)//BS
print(SPE,VS)
model = Sequential([
    Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(24,24,1)),
    MaxPooling2D(pool_size=(1,1)),
    Conv2D(32,(3,3),activation='relu'),
    MaxPooling2D(pool_size=(1,1)),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D(pool_size=(1,1)),
    Dropout(0.25),
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(2, activation='softmax')
])
model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])
model.fit_generator(train_batch,
validation_data=valid_batch,epochs=15,steps_per_epoch=SPE
,validation_steps=VS)
model.save('rak.h5', overwrite=True)Main Program

```

```

import cv2
import os
from keras.models import load_model
import numpy as np
from pygame import mixer
import time
mixer.init()
sound = mixer.Sound('alarm.wav')
face = cv2.CascadeClassifier('F:/Main Project/haar cascade
                             files/haarcascade_frontalface_alt.xml')
leye = cv2.CascadeClassifier('F:/Main Project/haar cascade
                             files/haarcascade_lefteye_2splits.xml')
reye = cv2.CascadeClassifier('F:/Main Project/haar cascade
                             files/haarcascade_righteye_2splits.xml')
lbl=['Close','Open']
model = load_model("F:/Main Project/rakmodel.h5")
path = os.getcwd()
cap = cv2.VideoCapture(0)
font = cv2.FONT_HERSHEY_COMPLEX_SMALL
count=0
score=0
thicc=2
rpred=[99]
lpred=[99]
while(True):ret,
frame = cap.read()
height,width = frame.shape[:2]
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces =
face.detectMultiScale(gray,minNeighbors=5,scaleFactor=1.1,minSize=(25,25
))
left_eye = leye.detectMultiScale(gra
right_eye = reye.detectMultiScale(gray)
cv2.rectangle(frame, (0,height-50) , (200,height) , (0,0,0) ,

```

```

thickness=cv2.FILLED )
for (x,y,w,h) in faces:
    cv2.rectangle(frame, (x,y) , (x+w,y+h) , (100,100,100) , 1 )
for (x,y,w,h) in right_eye:
    r_eye=frame[y:y+h,x:x+w]
    count=count+1
    r_eye = cv2.cvtColor(r_eye,cv2.COLOR_BGR2GRAY)
    r_eye = cv2.resize(r_eye,(24,24))
    r_eye= r_eye/255
    r_eye= r_eye.reshape(24,24,-1)
    r_eye = np.expand_dims(r_eye,axis=0)
    rpred = model.predict_classes(r_eye)
    if(rpred[0]==1):
        lbl='Open'
    if(rpred[0]==0):
        lbl='Closed'
    break
for (x,y,w,h) in left_eye:
    l_eye=frame[y:y+h,x:x+w]
    count=count+1
    l_eye = cv2.cvtColor(l_eye,cv2.COLOR_BGR2GRAY)
    l_eye = cv2.resize(l_eye,(24,24))
    l_eye= l_eye/255
    l_eye=l_eye.reshape(24,24,-1)
    l_eye = np.expand_dims(l_eye,axis=0)
    lpred = model.predict_classes(l_eye)
    if(lpred[0]==1):
        lbl='Open'
    if(lpred[0]==0):
        lbl='Closed'
    break

if(rpred[0]==0 and lpred[0]==0):
    score=score+1

```

```

cv2.putText(frame,"Closed",(10,height-20), font,
1,(255,255,255),1,cv2.LINE_AA)
# if(rpred[0]==1 or lpred[0]==1):
else:
score=score-1
cv2.putText(frame,"Open",(10,height-20), font,
1,(255,255,255),1,cv2.LINE_AA)
if(score<0):
score=0
cv2.putText(frame,'Score:'+str(score),(100,height-20), font,
1,(255,255,255),1,cv2.LINE_AA)
if(score>15):

cv2.imwrite(os.path.join(path,'image.jpg'),frame)
try:
sound.play()
except:
pass
if(thicc<16):
thicc= thicc+2
else:
thicc=thicc-2
if(thicc<2):
thicc=2
cv2.rectangle(frame,(0,0),(width,height),(0,0,255),thicc)
cv2.imshow('frame',frame)
if cv2.waitKey(1) & 0xFF == ord('q'):
break
cap.release()
cv2.destroyAllWindows()

```

CHAPTER 5

RESULT AND DISCUSSION

5.1. RESULT

- The alarm made by the device is the output of the device. As we can't place a physical object here, we will include a screenshot of the sound file in the projec folder.

haar cascade files	9/25/2019 12:20 PM	File folder	
models	1/21/2020 7:23 PM	File folder	
alarm.wav	4/16/2019 9:27 PM	WAV File	996 KB
drowsiness detection.py	1/21/2020 7:35 PM	PY File	4 KB

Fig 5.1 showing alarm.wav file in the project folder

- But for the convenience we can see the output by running these files in a desktop/laptop.

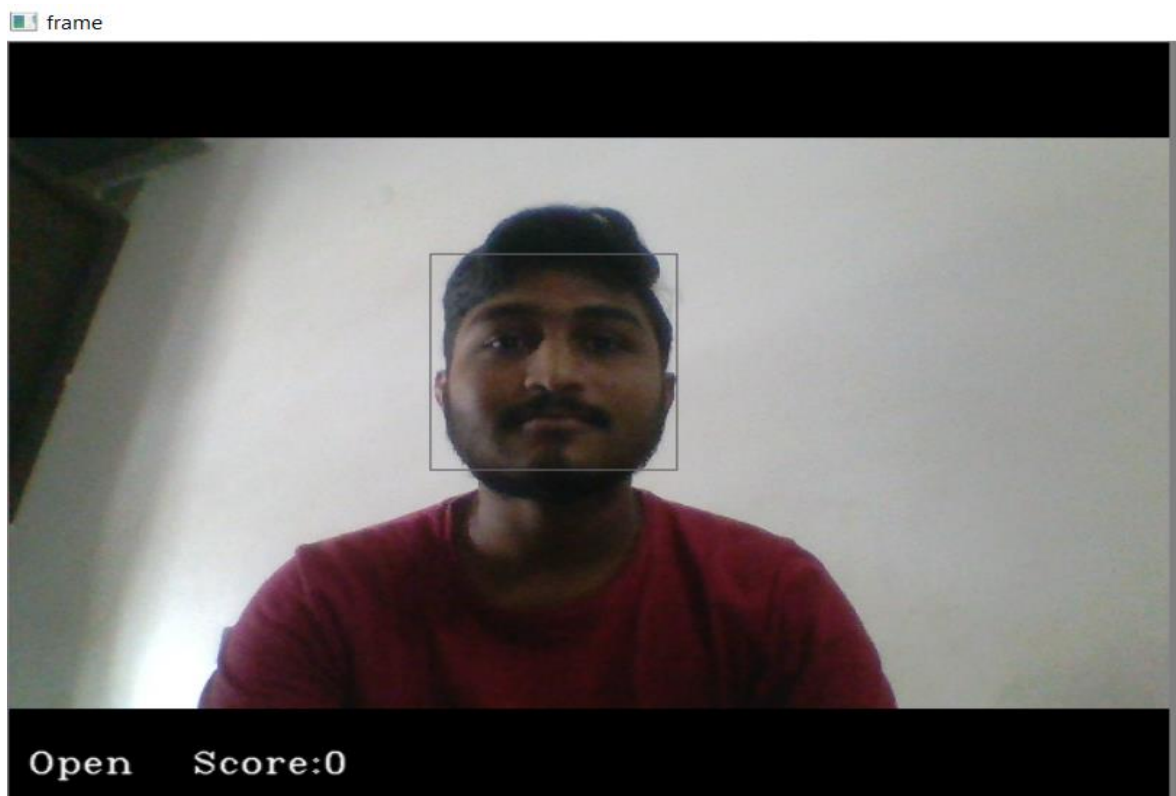


Fig 5.2 showing driver's face with open eyes

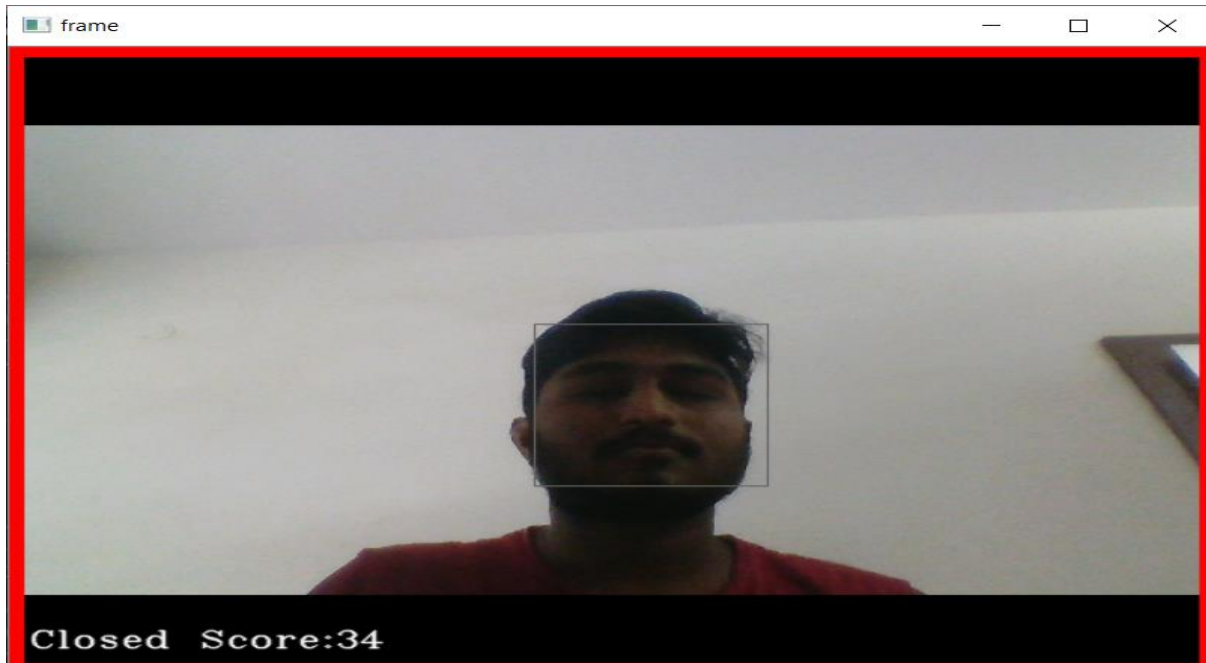


Fig 5.3 showing driver's face with closed eyes

5.2. PERFORMANCE ANALYSIS

The software for detecting the drowsiness of the driver is developed by using python, keras and opencv module which uses tensor flow backend. The model which is developed to classify the eyes gives 94.4% test

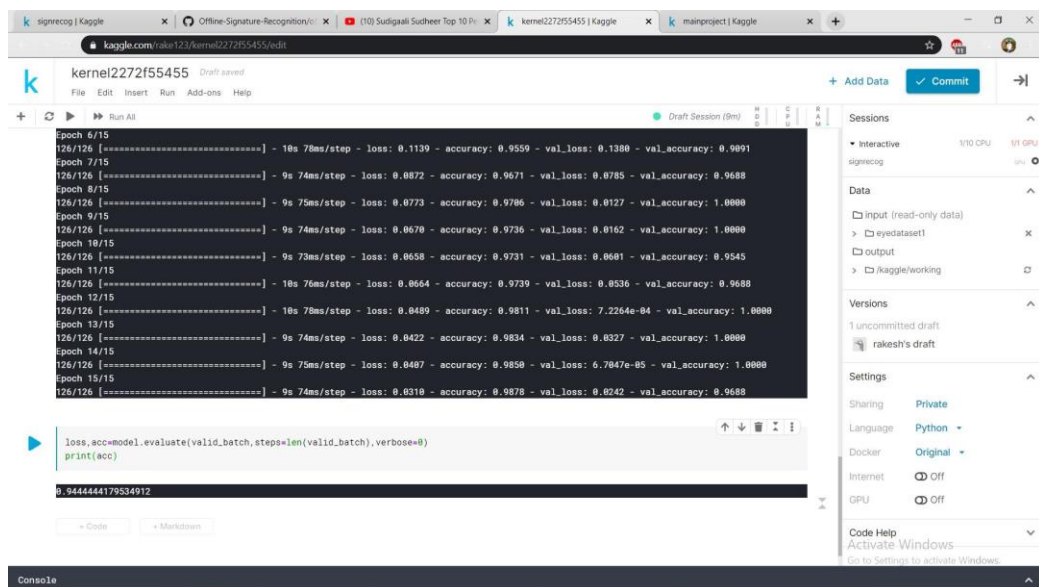


Fig 5.4 figure showing test accuracy of the model

CHAPTER 6

SUMMARY AND CONCLUSION

The device built was capturing the faces and then eyes and blowing the alarm when the eyes are closed. There are many factors that play a major role while detecting the faces. Some of them are light and place where camera module is placed. If we use a basic camera module of 5MP it will be very difficult to identify a face in low light and if we place camera in a low light then it will be difficult to identify the face. The device should be placed in such a position that there should be some light on the driver's face then it will be more effective in identifying the face.

This device has many real time applications. It can also be used in monitoring the jobs which are risky to do and in the jobs where the employee's attention has given the top most priority. Basically, we are using a 5MP normal camera so, sometimes it can't capture the face correctly in low light or at night times but by using a good camera or night vision camera we can capture these images correctly. It can be very helpful for the students who are preparing or for some institutions who are training the students for competitive exams in their preparation. Because they have to concentrate for long hours on their studies. It can also be used in monitoring pilots who are on duty in the cockpit. We can also note the timings at which the driver is feeling tired and drowsy and we can take necessary actions. We can also develop an auto braking system for any vehicle based on the driver's tiredness and drowsiness.

REFERENCES

- . Sagonas, E. Antonakos, G. Tzimiropoulos, S.Zafeiriou and M. Pantic, "300 Faces in-The-Wild Challenge: database and results," Image and Vision Computing, vol.47, pp 3-18, 2016.
- Jie Yi Wong and Yee Lau,"Real-Time Driver Alert System",ECTI TRANSACTIONS ON ELECTRICAL ENG., ELECTRONICS, AND COMMUNICATIONS VOL.17, NO.2 August 2019
- Y.Surya Kumar and V.Shiva Narasimha,"IOT BASED DRIVER DROWSINESS DETECTION AND ALERT SYSTEM", IJRAR, Volume 7, Issue 2, April 2020.
- Python documentation:
<https://docs.python.org/3/>
- Opencv documentation :
https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_setup/py_intro/pyintro.html