Artificial Intelligence Project

DRIVER DROWSINESS DETECTION

Submitted by

TEAM 9

Millennium-01301032017

A.Vaishnavi -05601032017

Rashmita Yadav- 05801032017

Rhea Prasad- 06101032017

Mahima Patel- 06201032017

Under the Supervision of

Mr. Rishabh Kaushal

Assistant Professor

Department of Information Technology

Department of Information Technology

Indira Gandhi Delhi Technical University for Women

Kashmere Gate, Delhi - 110006

Contents

ABSTRACT

In recent years driver fatigue is one of the major causes of vehicle accidents in the world. A direct way of measuring driver fatigue is measuring the state of the driver i.e. drowsiness. So it is very important to detect the drowsiness of the driver to save life and property. This project is aimed towards developing a prototype of drowsiness detection system. This system is a real time system which captures image continuously and measures the state of the eye according to the specified algorithm and gives warning if required.

Though there are several methods for measuring the drowsiness but this approach is completely non-intrusive which does not affect the driver in any way, hence giving the exact condition of the driver. This system works by monitoring the eyes of the driver and sounding an alarm when he/she is drowsy.

The priority is on improving the safety of the driver without being obtrusive. In this project the eye blink of the driver is detected. If the driver's eyes remain closed for more than a certain period of time, the driver is said to be drowsy and an alarm is sounded. The programming for this is done in OpenCV using the for the detection of facial features.

CHAPTER 1

INTRODUCTION

Driver drowsiness detection is a car safety technology which prevents accidents when the driver is getting drowsy. Various studies have suggested that around 20% of all road accidents are fatigue-related, up to 50% on certain roads. Driver fatigue is a significant factor in a large number of vehicle accidents. Recent statistics estimate that annually 1,200 deaths and 76,000 injuries can be attributed to fatigue related crashes. The development of technologies for detecting or preventing drowsiness at the wheel is a major challenge in the field of accident avoidance systems. Because of the hazard that drowsiness presents on the road, methods need to be developed for counteracting its affects. Driver inattention might be the result of a lack of alertness when driving due to driver drowsiness and distraction. Driver distraction occurs when an object or event draws a person's attention away from the driving task. Unlike driver distraction, driver drowsiness involves no triggering event but, instead, is characterized by a progressive withdrawal of attention from the road and traffic demands. Both driver drowsiness and distraction, however, might have the same effects, i.e., decreased driving performance, longer reaction time, and an increased risk of crash involvement. shows the block diagram of overall system. Based on Acquisition of video from the camera that is in front of driver perform real-time processing of an incoming video stream in order to infer the driver's level of fatigue if the drowsiness is Estimated then it will give the alert by sensing the eyes.

1.1 PROBLEM STATEMENT

1 in 4 vehicle accidents are caused by drowsy driving and 1 in 25 adult drivers report that they have fallen asleep at the wheel in the past 30 days. The scariest part is that drowsy driving isn't just falling asleep while driving. Drowsy driving can be as small as a brief state of unconsciousness when the driver is not paying full attention to the road. Drowsy driving results in over 71,000 injuries, 1,500 deaths, and $12.5 billion in monetary losses per year. Due to the relevance of this problem, we believe it is important to develop a solution for drowsiness detection, especially in the early stages to prevent accidents.

Additionally, we believe that drowsiness can negatively impact people in working and classroom environments as well. Although sleep deprivation and college go hand in hand, drowsiness in the workplace especially while working with heavy machinery may result in serious injuries similar to those that occur while driving drowsily.

1.1.1 OBJECTIVE

The objective is to measure physical changes (i.e. open/closed eyes and mouth to detect fatigue) is well suited for real world conditions since it is non-intrusive by using a video camera to detect changes. In addition, micro sleeps that are short period of sleeps lasting 2 to 3 minutes are good indicators of fatigue. Thus, by continuously monitoring the eyes and mouth of the driver one can detect the sleepy state of driver and a timely warning is issued.

Our solution to this problem is to build a detection system that identifies key attributes of drowsiness and triggers an alert when someone is drowsy before it is too late.

In this code we build a system of Driver drowsiness detection via eye and mouth monitoring being it closed or opened using Python, OpenCV, which will alert the driver when he feels sleepy.

CHAPTER 2

LITERATURE REVIEW

2.1 WHAT IS DROWSINESS?

Drowsiness is defined as a decreased level of awareness portrayed by sleepiness and trouble in staying alarm but the person awakes with simple excitement by stimuli. It might be caused by an absence of rest, medicine, substance misuse, or a cerebral issue. It is mostly the result of fatigue which can be both mental and physical. Physical fatigue, or muscle weariness, is the temporary physical failure of a muscle to perform ideally. Mental fatigue is a temporary failure to keep up ideal psychological execution. The onset of mental exhaustion amid any intellectual action is progressive, and relies on an individual's psychological capacity, furthermore upon different elements, for example, lack of sleep and general well-being. Mental exhaustion has additionally been appeared to diminish physical performance. It can show as sleepiness, dormancy, or coordinated consideration weakness. In the past years according to available data driver sleepiness has gotten to be one of the real reasons for street mishaps prompting demise and extreme physical injuries and loss of economy. A driver who falls asleep is in an edge of losing control over the vehicle prompting crash with other vehicle or stationary bodies. Keeping in mind to stop or reduce the number of accidents to a great extent the condition of sleepiness of the driver should be observed continuously.

2.2 QUESTIONARIES

Following information from the driver about more than one incident which may or may not be critical:

How much sleep did you get before this drive?

What were the road conditions?

What were the traffic conditions?

What type of road was it? E.g. curvy, straight, two-lane, etc.

Were you tired when you started?

What speed were you going?

What time of the day or night was it?

At what point during your drive did this happen?

Where were you going?

How long did you keep driving?

What did you do to try to become more alert?

What else would have helped you become more alert?

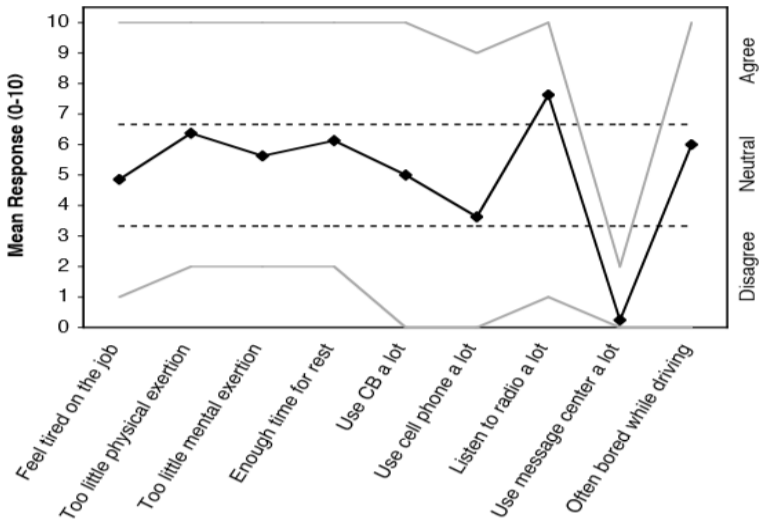
After the incident, were you more alert? How long did this “adrenaline effect” last?

Were you more tired once the adrenaline wore off?

When you stopped driving was it because you were drowsy or another reason?

Did you nap at all when you stopped?

Do you think that you drove at all with your eyes closed? If so, for how long?



Mean, maximum, and minimum driver responses to rating questions

CB: who do you speak with?	Frequency
Anyone	3
Company drivers	3
Don't use	2
Sometimes anyone	1
Cell phone: who do you speak with?	Frequency
Family	6
Work related	2
Friends	1
Radio: what do you listen to?	Frequency
Music	7
NPR	2
Talk	2
News	1

Driver responses to perceived value of a drowsy driver system

CHAPTER 3

COUNTER MEASURES

The study states that the reason for a mishap can be categorized as one of the accompanying primary classes:

(1) human

(2) vehicular

(3) surrounding factor

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

3.1 Vehicle based measures

Vehicle-based measures survey path position, which monitors the vehicle's position as it identifies with path markings, to determine driver weakness, and accumulate steering wheel movement information to characterize the fatigue from low level to high level.

Disadvantages:

-Vehicle based measures mostly affected by the geometry of road which sometimes unnecessarily activates the alarming system.

-The driving style of the current driver needs to be learned and modeled for the system to be efficient.

-The condition like micro sleeping which mostly happens in straight highways cannot be detected.

3.2 Physiological measures

Physiological measures are the objective measures of the physical changes that occur in our body because of fatigue. These physiological changes are measured by their respective instruments as follows: ECG, EMG, EOG & EEG.

Disadvantages:

-It requires placement of several electrodes to be placed on head, chest and face which is not at all a convenient and annoying for a driver.

-They need to be very carefully placed on respective places for perfect result.

3.3 Subjective Methods

Subjective methods involve assessing the drivers' current level of drowsiness by subjecting them to ratings in the form of questionnaires. These ratings are usually self-evaluated or evaluated by experts watching the driver in action. To detect the changes in a driver's drowsiness state, conducted a pre-experimental, mid-experimental, and postexperimental Karolinska Sleepiness Scale (KSS) exercise. These ratings were the keys to define their drowsiness ground truth , methods like Stanford Sleepiness Scale and Karolinska Sleepiness Scale are the two most widely utilized subjective measures. SSS is a 7-point measurement scale describing the current state of drowsiness of an individual which is further most likely be used to categorize driver drowsiness into only two states. This is because of the close relation of each scale. KSS, on the other hand, is a 9-point scale. A contrast to SSS, this scale is considered a robust scale capable of categorizing driver's drowsiness into different levels.

SSS

Value	Description
1	Feeling active, vital, alert, or wide awake
2	Functioning at high levels, but not at peak; able to concentrate
3	Awake, but relaxed; responsive but not fully alert
4	Little foggy; not at peak
5	Foggy; losing interest in remaining awake; slowed down
6	Sleepy; woozy; fighting sleep; prefer to lie down
7	No longer fighting sleep; sleep onset soon; cannot stay awake

KSS

Value	Sleepiness Level
1	Extremely alert
2	Very alert
3	Alert
4	Rather alert
5	Neither alert nor sleepy
6	Some signs of sleepiness
7	Sleepy but no difficulty staying awake
8	Sleepy with some effort to keep alert
9	Extremely sleepy, fighting sleep

CHAPTER 4

PROPOSED METHODOLOGY

4.1 BEHAVIORAL MEASURES

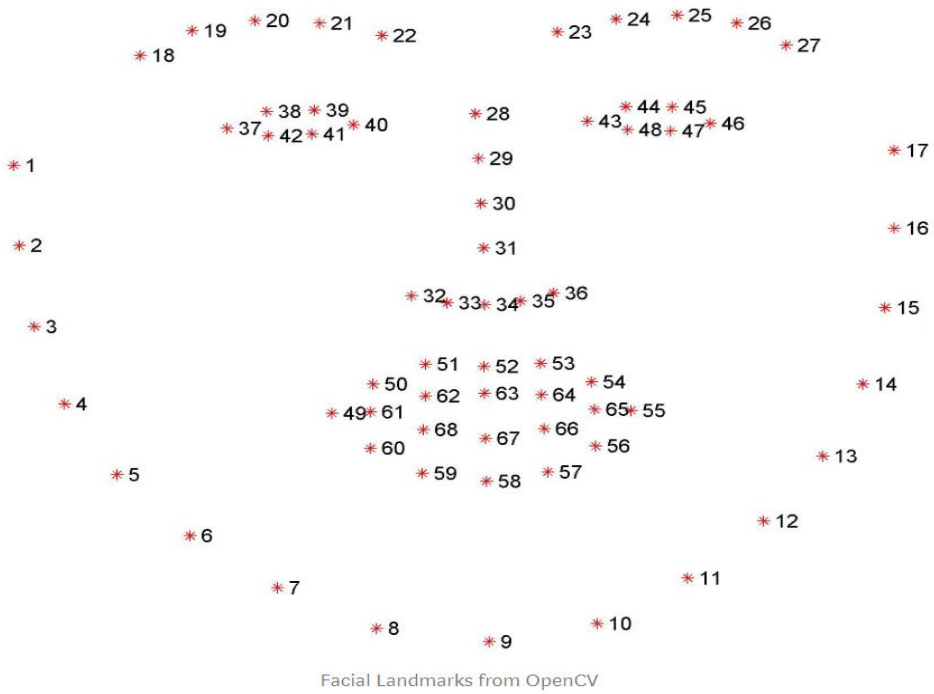
The objective is to measure physical changes (i.e. open/closed eyes and mouth to detect fatigue) is well suited for real world conditions. Under this technique eye aspect ratio, mouth aspect ratio, etc. of a person is monitored through a camera and the person is alerted if any of these drowsiness symptoms are detected.

4.2 DATA SOURCE AND PREPROCESSING

For our training and test data, we use REAL LIFE DROWSINESS DATASET created by a research team from the University of Texas at Arlington specifically for detecting multi-stage drowsiness. The end goal is to detect not only extreme and visible cases of drowsiness but allow our system to detect softer signals of drowsiness as well. The dataset consists of around 30 hours of videos of 60 unique participants. From the dataset, we were able to extract facial landmarks from 44 videos of 22 participants. This allowed us to obtain a sufficient amount of data for both the alert and drowsy state.

For each video, we used OpenCV to extract 1 frame per second starting at the 3-minute mark until the end of the video.
Each video was approximately 10 minutes long, so we extracted around 240 frames per video, resulting in 10560 frames for the entire dataset.

There were 68 total landmarks per frame but we decided to keep the landmarks for the eyes and mouth only (Points 37–68). These were the important data points we used to extract the features for our model.



Facial Landmarks from OpenCV

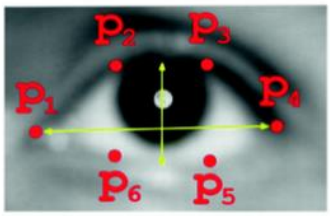
4.3 FEATURE COMPUTATION

4.3.1 FEATURE EXTRACTION

We ventured into developing suitable features for our classification model. While we hypothesized and tested several features like mouth aspect ratio, pupil circularity, and finally, mouth aspect ratio over eye aspect ratio, the core features that we concluded on for our final models were:

Eye Aspect Ratio (EAR)

EAR, as the name suggests, is the ratio of the length of the eyes to the width of the eyes. The length of the eyes is calculated by averaging over two distinct vertical lines across the eyes as illustrated in the figure below.

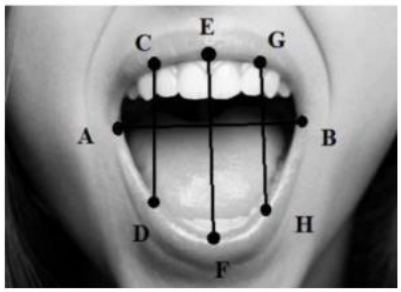


$$\text{EAR} = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

Our hypothesis was that when an individual is drowsy, their eyes are likely to get smaller and they are likely to blink more. Based on this hypothesis, we expected our model to predict the class as drowsy if the eye aspect ratio for an individual over successive frames started to decline i.e. their eyes started to be more closed or they were blinking faster.

Mouth Aspect Ratio (MAR)

Computationally similar to the EAR, the MAR, as you would expect, measures the ratio of the length of the mouth to the width of the mouth. Our hypothesis was that as an individual becomes drowsy, they are likely to yawn and lose control over their mouth, making their MAR to be higher than usual in this state.



$$\text{MAR} = \frac{|\text{EF}|}{|\text{AB}|}$$

Pupil Circularity (PUC)

PUC is a measure complementary to EAR, but it places a greater emphasis on the pupil instead of the entire eye.

$$Circularity = \frac{4 * \pi * Area}{perimeter^2} \qquad Area = \left(\frac{\text{Distance(p2, p5)}}{2}\right)^2 * \pi$$

$$Perimeter = Distance(p1, p2) + Distance(p2, p3) + Distance(p3, p4) + Distance(p4, p5) + Distance(p5, p6) + Distance(p6, p1)$$

For example, someone who has their eyes half-open or almost closed will have a much lower pupil circularity value versus someone who has their eyes fully open due to the squared term in the denominator. Similar to the EAR, the expectation was that when an individual is drowsy, their pupil circularity is likely to decline.

Mouth Over Eye Ratio (MOE)

MOE is simply the ratio of the MAR to the EAR.

The benefit of using this feature is that EAR and MAR are expected to move in opposite directions if the state of the individual changes. As opposed to both EAR and MAR, MOE as a measure will be more responsive to these changes as it will capture the subtle changes in both EAR and MAR and will exaggerate the changes as the denominator and numerator move in opposite directions. Because the MOE takes MAR as the numerator and EAR as the denominator, our theory was that as the individual gets drowsy, the MOE will increase.

$$MOE = \frac{MAR}{EAR}$$

4.3.2 FEATURE NORMALIZATION

When we were testing our models with the core features discussed above, we witnessed an alarming pattern. Whenever we randomly split the frames in our training and test, our model would yield results with accuracy as high 70%, however, whenever we split the frames by individuals (i.e. an individual that is in the test set will not be in the training set), our model performance would be poor as alluded to earlier.

This led us to the realization that our model was struggling with new faces and the primary reason for this struggle was the fact that each individual has different core features in their default alert state. That is, person A may naturally have much smaller eyes than person B. If a model is trained on person B, the model, when tested on person A, will always predict the state as drowsy because it will detect a fall in EAR and PUC and a rise in MOE even though person A was alert. Based on this discovery, we hypothesized that normalizing the features for each individual is likely to yield better results and as it turned out, we were correct.

To normalize the features of each individual, we took the first three frames for each individual's alert video and used them as the baseline for normalization. The mean and standard deviation of each feature for these three frames were calculated and used to normalize each feature individually for each participant. Mathematically, this is what the normalization equation looked like:

$$\boldsymbol{Normalised\ Feature}_{n,m} = \frac{\boldsymbol{Feature}_{n,m} - \boldsymbol{\mu}_{n,m}}{\boldsymbol{\sigma}_{n,m}}$$

where:
n is the feature
m is the person
$\mu_{n,m}$ *and* $\sigma_{n,m}$ *are taken from the first 3 frames of the "Alert" state*

Now that we had normalized each of the four core features, our feature set had eight features, each core feature complemented by its normalized version. We tested all eight features in our models and our results improved significantly.

CHAPTER 5

DATA EXPLORATION

5.1 METRICS

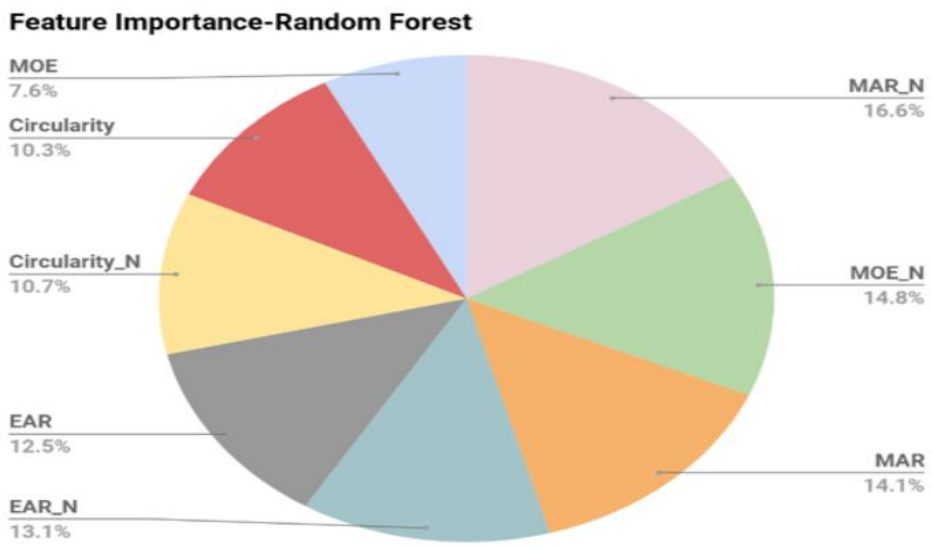
	importance
MAR_N	0.167546
MOE_N	0.148529
MAR	0.140682
EAR_N	0.131376
EAR	0.125618
Circularity_N	0.106515
Circularity	0.103410
MOE	0.076324

5.2 VISUALIZATION

We wanted to get a sense of feature importance so we visualized the results from our Random Forest model.

Mouth Aspect Ratio after normalization turned out to be the most important feature out of our 8 features. This makes sense because when we are drowsy, we tend to yawn more frequently. Normalizing our features exaggerated this effect and made it a better indicator of drowsiness in different participants.

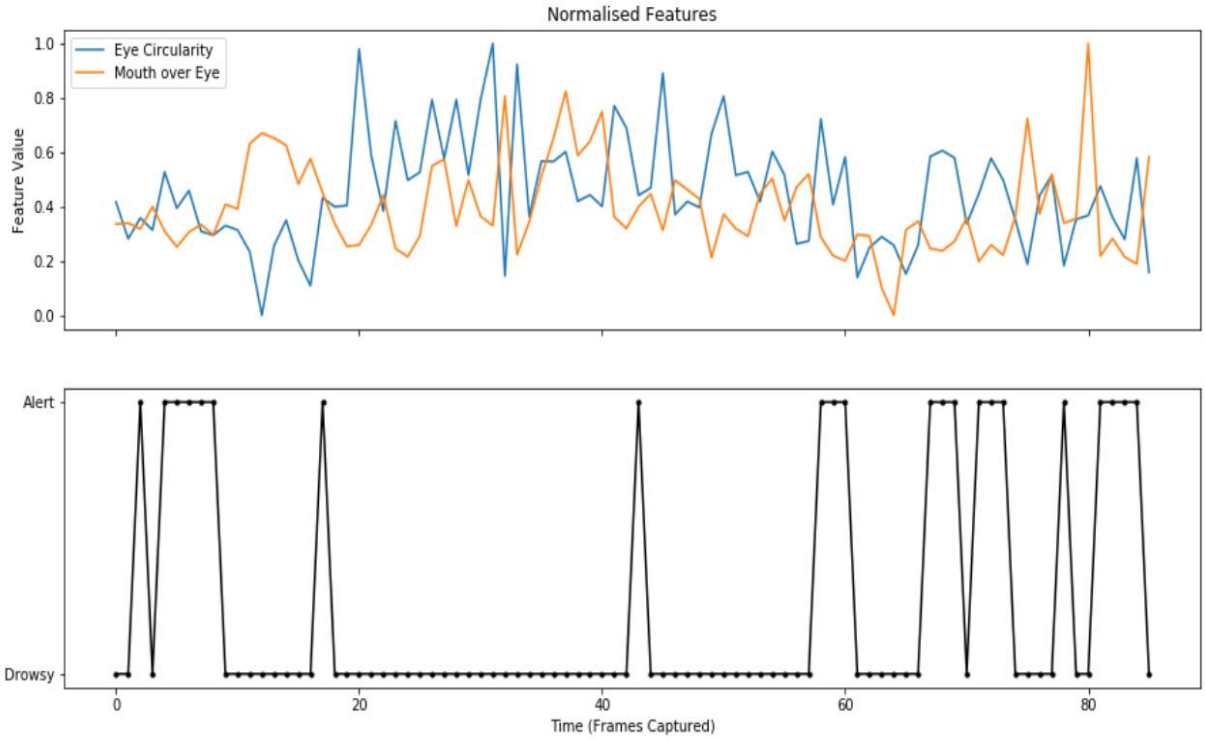
5.2.1



5.2.2 Normalized Feature Value with Time (Frames Captured):

5.2.3 State Prediction with Time (Frames Captured)

For input image captured by webcam:



CHAPTER 6

SYSTEM DESCRIPTION

6.1 TOOLS USED

1.Python 3 Interpreter

2.OpenCV

OpenCV is an open source computer vision library which was designed for computational efficiency and having a high focus on real-time image detection. One of OpenCV's goals is to provide a simple-to-use computer vision infrastructure which helps people to build highly sophisticated vision applications fast. The OpenCV library, containing over 500 functions, spans many areas in vision. Because computer vision and machine learning often goes hand-in-hand, OpenCV also has a complete, general-purpose, Machine Learning Library (MLL). This sub library is focused on statistical pattern recognition and clustering. The MLL is very useful for the vision functions that are the basis of OpenCV's usefulness, but is general enough to be used for any machine learning problem.

3.Dlib Libraries and their dependencies

It is a C++ toolkit for machine learning, it also provides a python API to use it in your python apps. One of its best features is a great documentation for C++ and Python API. It is used in the code to detect faces and get facial landmarks coordinates especially the 12 points which define the two eyes left and right. After getting the 12 points of left and right eye, we compute Eye aspect ratio to estimate the level of the eye opening. Open eyes have high values (>0.2) of EAR while the closed eye it is getting close to zero.

4.Keras – pip install Keras (to build our classification model). It uses TensorFlow as backend.

5.Pygame – pip install pygame (to play alarm sound).

6.2 TECHNOLOGY USED

The various technology that can be used are discussed as:

TensorFlow:

IT is an open-source software library for dataflow programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production. TensorFlow computations are expressed as stateful dataflow graphs. The name TensorFlow derives from the operations that such neural networks perform on multidimensional data arrays. These arrays are referred to as "tensors".

Machine learning

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

OpenCV:

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

6.3 BASIC STEPS

Step 1 – Take image as input from a camera.
Step 2 – Detect the face in the image and create a Region of Interest (ROI).
Step 3 – Detect the eyes and mouth from ROI and feed it to the classifier.
Step 4 – Classifier will categorize whether eyes are open or closed.
Step 5 – Check whether the person is drowsy.

CHAPTER 7

ALGORITHMS

7.1 BASIC CLASSIFICATION METHODS

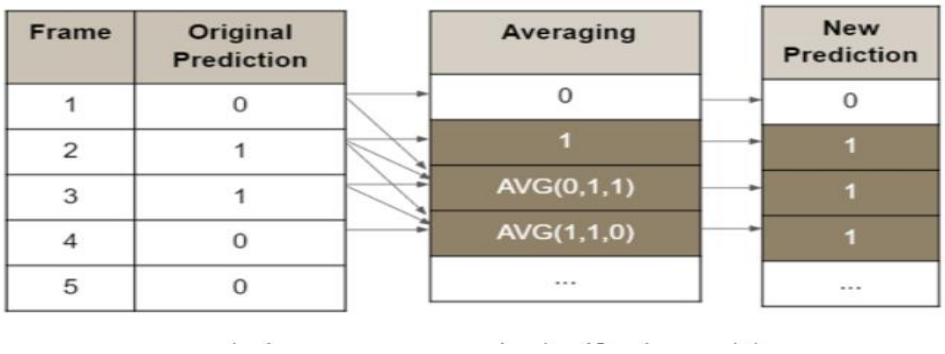
7.1.1 Description

After we extracted and normalized our features, we wanted to try a series of modeling techniques, starting with the most basic classification models like logistic regression and Naive Bayes, moving on to more complex models containing neural networks and other deep learning approaches.

In order to train and test our models, we split our dataset into data from 17 videos and data from 5 videos respectively. As a result, our training dataset contains 8160 rows and our test dataset contains 2400 rows.

To predict the label for each frame in the sequence

The way we introduced sequence to basic classification methods was to average the original prediction results with the prediction results from the previous two frames. Since our dataset was divided into training and test based on the individual participants and the data points are all in the order of time sequence, averaging makes sense in this case and allowed us to deliver more accurate predictions.



7.2 Convolutional Neural Networks (CNN)

7.2.1 Description

It is a special type of deep neural network which performs extremely well for image classification purposes. A CNN basically consists of an input layer, an output layer and a hidden layer which can have multiple numbers of layers.

Convolutional Neural Networks (CNN) are typically used to analyze image data and map images to output variables. However, we decided to build a 1-D CNN and send in numerical features as sequential input data to try and understand the spatial relationship between each feature for the two states. Our CNN model has 5 layers including 1 convolutional layer, 1 flatten later, 2 fully connected dense layers, and 1 dropout layer before the output layer. The flatten layer flattens the output from the convolutional layer and makes it linear before passing it into the first dense layer. The dropout layer randomly drops 20% of the output nodes from the second dense layer in order to prevent our model from overfitting to the training data. The final dense layer has a single output node that outputs 0 for alert and 1 for drowsy.

7.2.2 CNN PARAMETERS

Activation Function	Relu/Sigmoid
Optimizer	Adam
Loss Function	Binary Crossentropy
Number of Epochs	100
Learning Rate	0.00001

7.2.3 CNN MODEL DESIGN:

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_8 (Conv1D)            (None, 6, 64)             256
_________________________________________________________________
flatten_8 (Flatten)          (None, 384)               0
_________________________________________________________________
dense_22 (Dense)             (None, 32)                12320
_________________________________________________________________
dense_23 (Dense)             (None, 16)                528
_________________________________________________________________
dropout_4 (Dropout)          (None, 16)                0
_________________________________________________________________
dense_24 (Dense)             (None, 1)                 17
=================================================================
Total params: 13,121
Trainable params: 13,121
Non-trainable params: 0
```

7.3 Long Short-Term Memory Networks

7.3.1 Description

Another method to deal with sequential data is using an LSTM model. LSTM networks are a special kind of Recurrent Neural Networks (RNN), capable of learning long-term dependencies in the data. Recurrent Neural Networks are feedback neural networks that have internal memory that allows information to persist.

Internal memory space of RNN while processing new data:

When making a decision, RNNs consider not only the current input but also the output that it has learned from the previous inputs. This is also the main difference between RNNs and other neural networks. In other neural networks, the inputs are independent of each other. In RNNs, the inputs are related to each other.

Why LSTM?

We chose to use an LSTM network because it allows us to study long sequences without having to worry about the gradient vanishing problems faced by traditional RNNs. Within the LSTM network, there are three gates for each time step: Forget Gate, Input gate, and Output Gate.

Forget Gate: as its name suggests, the gate tries to “forget” part of the memory from the previous output.

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Forget Gate Formula

Input Gate: the gate decides what should be kept from the input in order to modify the memory.

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Input Gate Formula

Output Gate: the gate decides what the output is by combining the input and memory.

$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

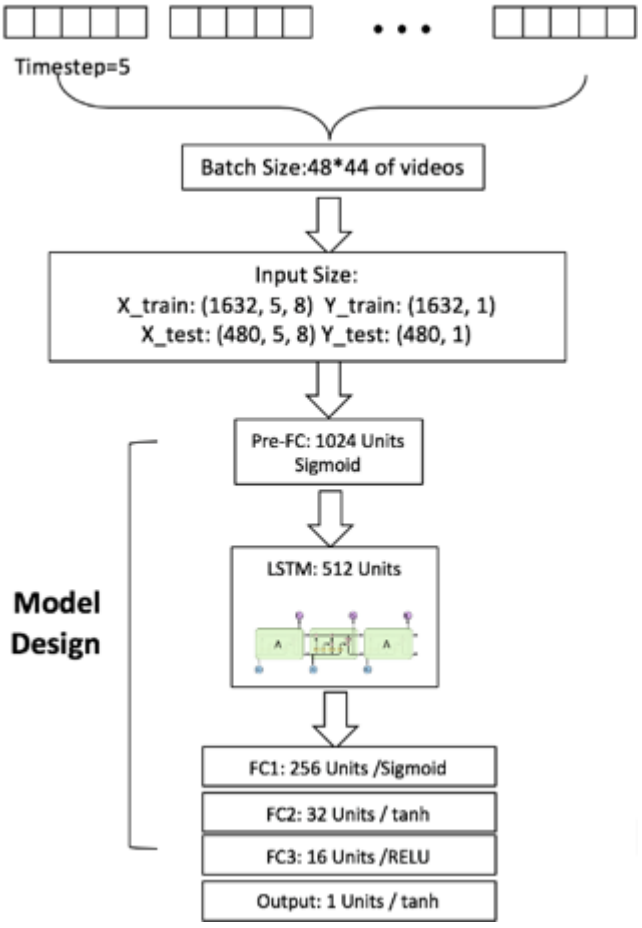
Output Gate Formula

First, we converted our videos into batches of data. Then, each batch was sent through a fully connected layer with 1024 hidden units using the sigmoid activation function. The next layer is our LSTM layer with 512 hidden units followed by 3 more FC layers until the final output layer.

7.3.2 LSTM PARAMETERS

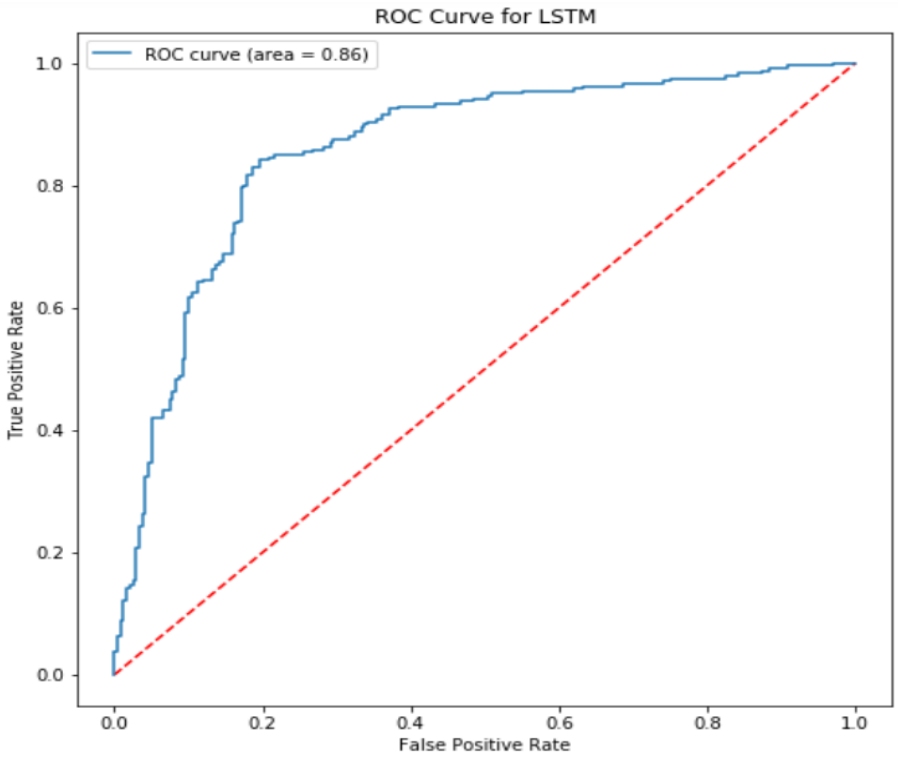
Number of Epochs	50
Learning Rate	0.00005
Timestep	5

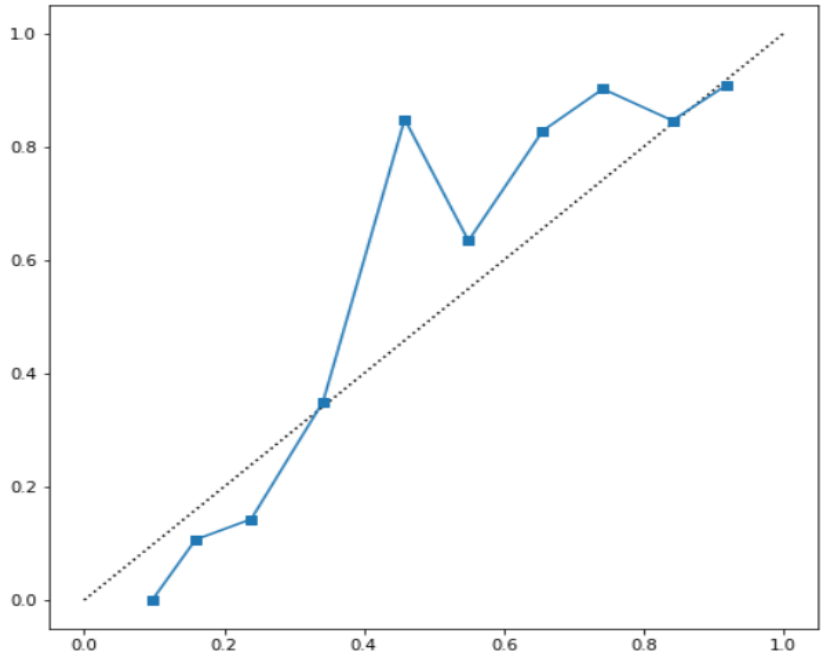
7.3.3 LSTM MODEL DESIGN



7.3.4 GRAPHICAL REPRESENTATION OF THE RESULTS:

ROC CURVE & CALIBRATION CURVE (Between mean predicted values and fraction of positives)





7.3.5 CONCLUSION

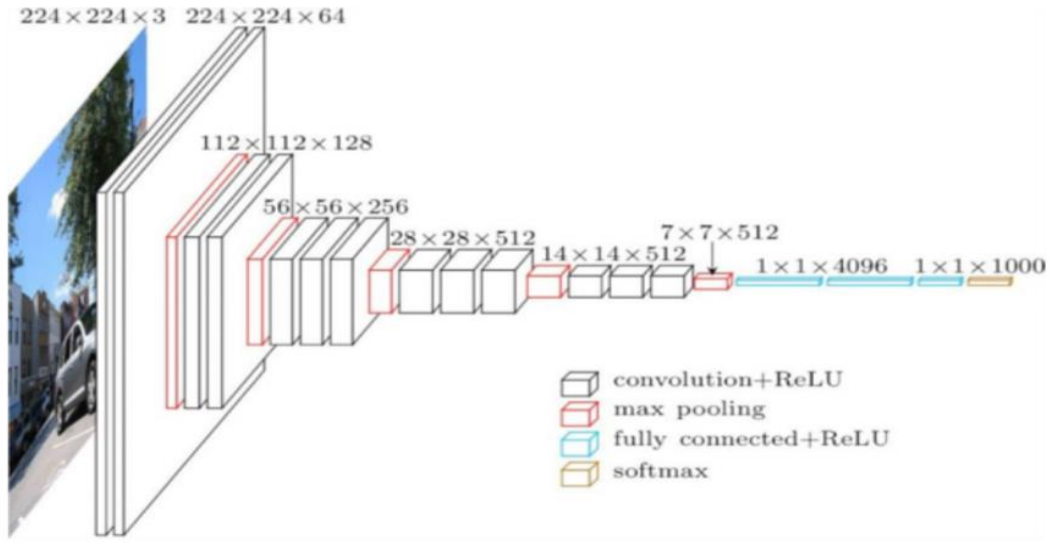
After hyperparameter tuning, our optimized LSTM model achieved an overall accuracy of 77.08% with a much lower false-negative rate of 0.3 compared to the false-negative rate of our kNN model (0.42).

7.4 VGG16 CNN MODEL

7.4.1 Description

VGG16 is a convolutional neural network model which was proposed by K. Simonyan and A. Zisserman from the University of Oxford in their paper “Very Deep Convolutional Networks for Large-Scale Image Recognition”. The model managed to achieve 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes. ILSVRC uses a smaller set of ImageNet with roughly 1000 images in each of 1000 categories. There are approximately 1.2 million training images, 50,000 validation images, and 150,000 testing images.

7.4.2 VGG16 NETWORK ARCHITECTURE



Transfer Learning:

Transfer learning focuses on using the knowledge gained while solving one problem and applying it to solve a different but related problem. It is a useful set of techniques especially for cases when we have limited time to train the model or limited data to fully train a neural network. Since the data we were working with had very few unique samples, we decided to use VGG16 model with the ImageNet dataset.

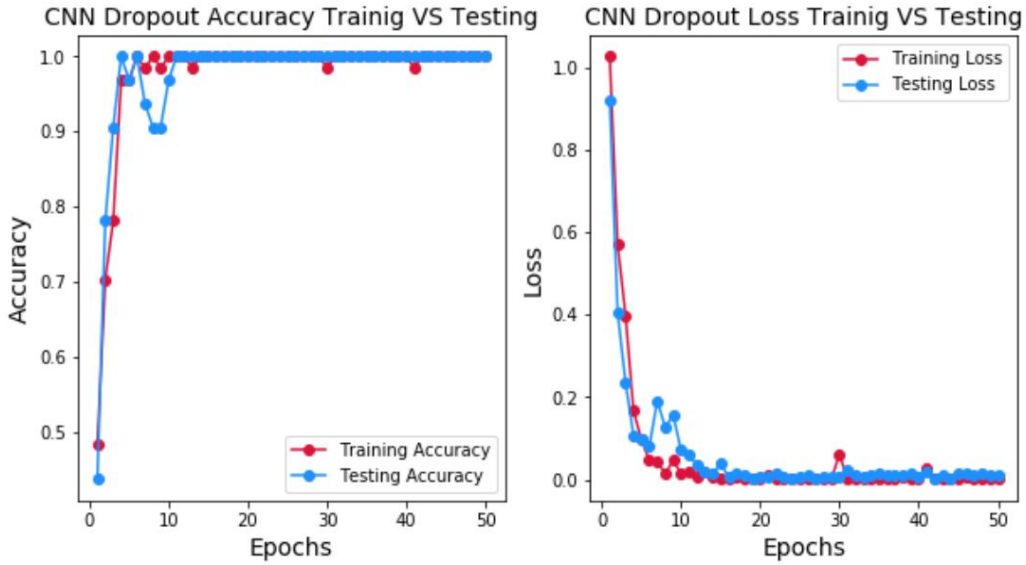
ImageNet consists of images with different resolutions. Therefore, the resolution of images needs to be changed to a fixed value of 256×256. The image is rescaled and cropped out and the central 256×256 patch forms the resulting image.

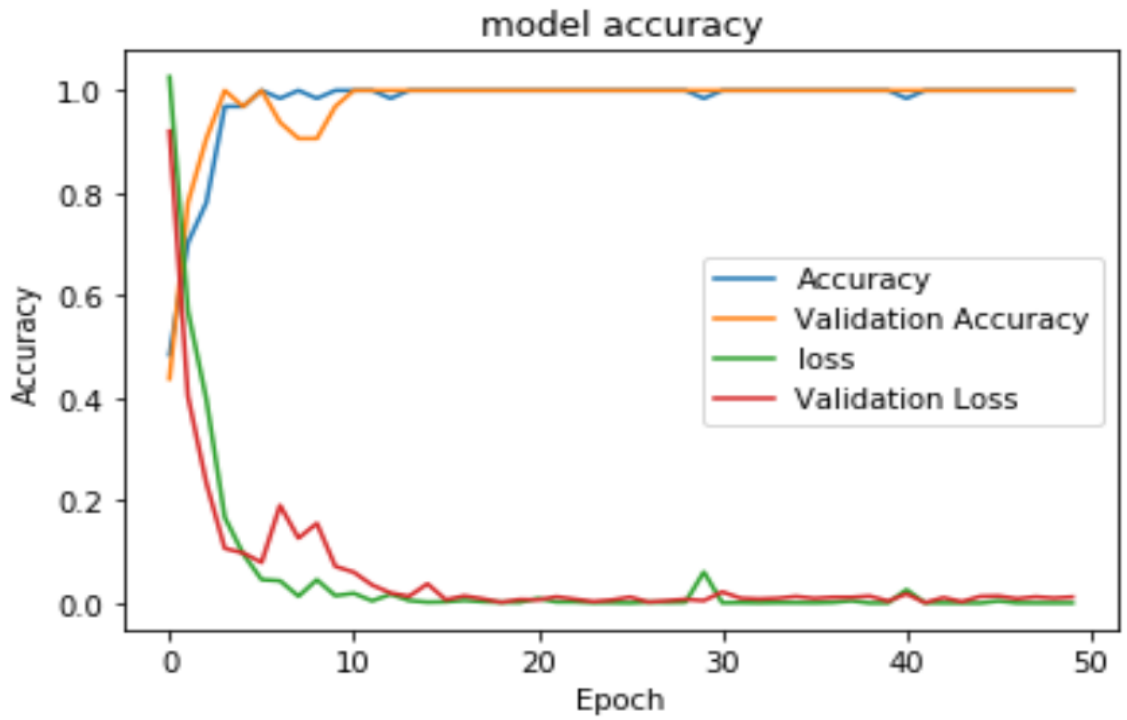
The input to cov1 layer is a 224 x 224 RGB image. The image is passed through a stack of convolutional layers, where the filters are used with a very small receptive field: 3×3. In one of the configurations, the model also utilizes 1×1 convolution filter, which can be seen as a linear transformation of the input channels followed by non-linear transformations. The convolution stride is fixed to 1 pixel; the spatial padding of convolutional layer input is such that the spatial resolution is preserved after convolution, i.e. the padding is 1-pixel for 3×3 convolutional layers. Spatial pooling is carried out by five max-pooling layers, which follow some of the convolutional layers. Not all the conv. layers are followed by max-pooling. Max-pooling is performed over a 2×2pixel window, with a stride of 2.

Three Fully-Connected (FC) layers follow a stack of convolutional layers: the first two have 4096 channels each, the third performs 1000-way ILSVRC classification and therefore contains 1000 channels. The final layer is a soft-max layer. The configuration of the fully connected layers is the same in all networks.

All hidden layers are equipped with the rectification (ReLU) non-linearity.
We split the training videos into 34,000 images which were screenshots taken every 10 frames. We fed these images to the VGG16 model. We believed that the number of images was sufficient to train the pre-trained model. We calculated the accuracy scores after training the model for 50 epochs.

7.4.3 GRAPHICAL REPRESENTATION OF THE RESULTS





7.4.4 CONCLUSION

It was clear that the model was overfitting. A possible explanation for this is that images that we passed through the model were of 22 respondents sitting virtually motionless in front of a camera with undisturbed backgrounds. So, despite taking a large number of frames (34,000) into our model, the model was essentially trying to learn from 22 sets of virtually identical images. Hence the model didn't really have enough training data in a true sense.