**TEAM MEMBERS**

1. Maradana Vijaya Krishna (A0178453W)
2. Putrevu Manoj Niyogi (A0213557E)
3. Sivasankaran Balakrishnan (A0065970X)

**Master of TECHNOLOGY INTELLIGENT SYSTEMS (PART-TIME) 2020  
  
SEMESTER 2 – Pattern Recognition and Machine Learning Systems (PRMLS)  
FINAL PROJECT REPORT**

**Driver Drowsiness Detection System**

Contents

[Executive Summary 2](#_Toc51742694)

[Objective 3](#_Toc51742695)

[Our Steps 4](#_Toc51742696)

[Technologies Used 4](#_Toc51742697)

[Driver Drowsiness Detection System Workflow 5](#_Toc51742698)

[The Model 6](#_Toc51742699)

[The model of layers 11](#_Toc51742700)

[Our approach 14](#_Toc51742701)

[Results from both models: 16](#_Toc51742702)

[Project deliverables: 17](#_Toc51742703)

[Steps to run the Project: 19](#_Toc51742704)

[Conclusions 20](#_Toc51742705)

[Limitations 20](#_Toc51742706)

[References: 20](#_Toc51742707)

[Appendix A: GRADUATE CERTIFICATE - Intelligent Reasoning Systems (IRS) Project Proposal 21](#_Toc51742708)

[Appendix B : Individual Project Report 25](#_Toc51742709)

# Executive Summary

Sleepiness can impair driving performance as much or more so than alcohol, studies show. (Dawson and Reid, 1997; Powell, 2001).

The American Automobile Association (AAA) estimates that one out of every six (16.5%) deadly traffic accidents, and one out of eight (12.5%) crashes requiring hospitalization of car drivers or passengers is due to drowsy driving. (AAA, 2010)

One analysis estimated the cost of automobile accidents attributed to sleepiness to be between $29.2 to $37.9 billion. (Leger, 1994)

(41%) admitted to having fallen asleep at the wheel at some point.; one in ten drivers (10%) reporting they did so within the past year. (AAA, 2010)

More than one-quarter of drivers (27%) admitting they had driven while they were “so sleepy that [they] had a hard time keeping [their] eyes open” within the past month (AAA, 2010)

Researchers estimate that more than 70 million Americans suffer from a sleep disorder. (Institute of Medicine, 2005) One of the most serious consequences of insufficient sleep is traffic accidents due to drowsy driving.

Experts suspect that even these disturbingly high figures underestimate the number of accidents or near-miss accidents due to drowsy driving because of drivers being unaware or not admitting they were drowsy at the time of the accident, or police not acquiring that information.

In spite of education to create awareness among driving community, policy initiatives by the Government to reduce the accidents numbers show no promising decline. In recent years, many of the luxury car companies have already moved in this direction but majority of the solutions are relatively employing complex sensors and devices which are costly and so is the need for simple, effective relatively low cost solutions that can be used by everyone.

Drowsy driving is a prevalent and serious public health issue that requires a simple low cost, fool proof continuous monitoring system that can be used by everyone.

# Objective

There are several types of solutions

* Steering pattern monitoring
* Vehicle position in lane monitoring
* Physiological measurement
* Driver face/eye monitoring

Steering pattern monitoring – It primarily uses steering input from electric power steering system. Learns driving behavior through steering input and compares it with the later rides. The system is based on the fact that the number of micro-corrections on the steering are lower than the one found in normal driving conditions[1, 2]. Any difference beyond threshold activates audio visual cues to draw attention. Monitoring this way only works when the driver actually steers the vehicle actively and does not activate automatic lane keeping system. The system can function reliably only at particular environments and is too dependent on the geometric characteristics of the road and, to a lesser extent, on the kinetic characteristics of the vehicle.

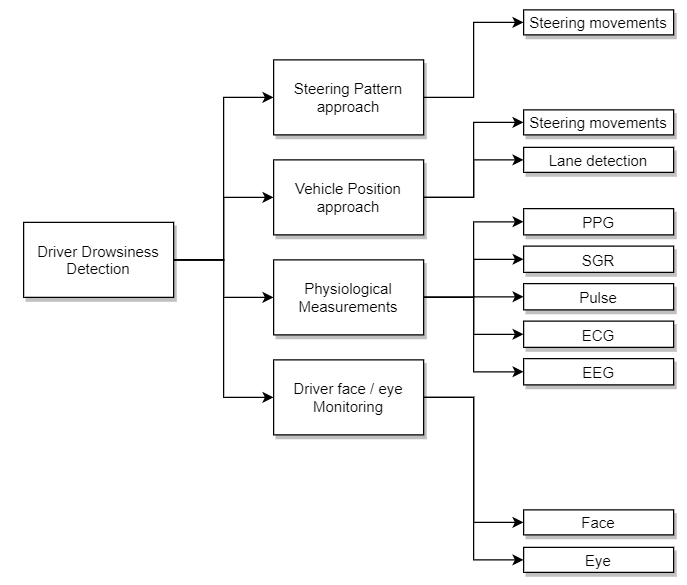
Vehicle position in lane monitoring – This method is based on the externally mounted camera(s) and associated software, which monitors the vehicle position relative to the lane. It depends on external factors such as road marking, weather, and light conditions.

Physiological measurement – Employs body sensors to measure parameters like brain activity, heart rate, skin conductance, muscle activity.

All the above types of models are relatively expensive.

Driver face/eye monitoring – This technique is simple and low-cost model relatively compared to other models and can be used in majority of the vehicles. The only limitation of this model is that the driver should not use dark colored spectacles that obscures the monitoring of eyes.

This technique monitors the state of eyes and if both the eyes are in a closed state for a defined threshold time, alarm is raised to draw attention of the driver.



*Figure 1 – Types of Solutions*

# Our Steps

The approach follows the CRISP-DM process. It consists of five stages, as listed below:

1. Understand the Business Requirements
2. Understanding & Analyzing the Data
3. Build the model
4. Verify / Optimize the model
5. Analyze the results and reveal the insights

The approach involves training the model with several real-world images of eyes in both ‘open’ and ’closed’ states. We have picked up the images from [MRL Eye Dataset](http://mrl.cs.vsb.cz/eyedataset).

The adopted approach is algorithmically simple, intuitive, and easy to implement. We relied on the proven off-the-shelf solution wherever appropriate, for example based on the popularity and success we chose Viola-Jones face and eye detectors.

Easily portable to different platforms.

Computationally in expensive as the training is done offline and the trained model is picked up for use.

# Technologies Used

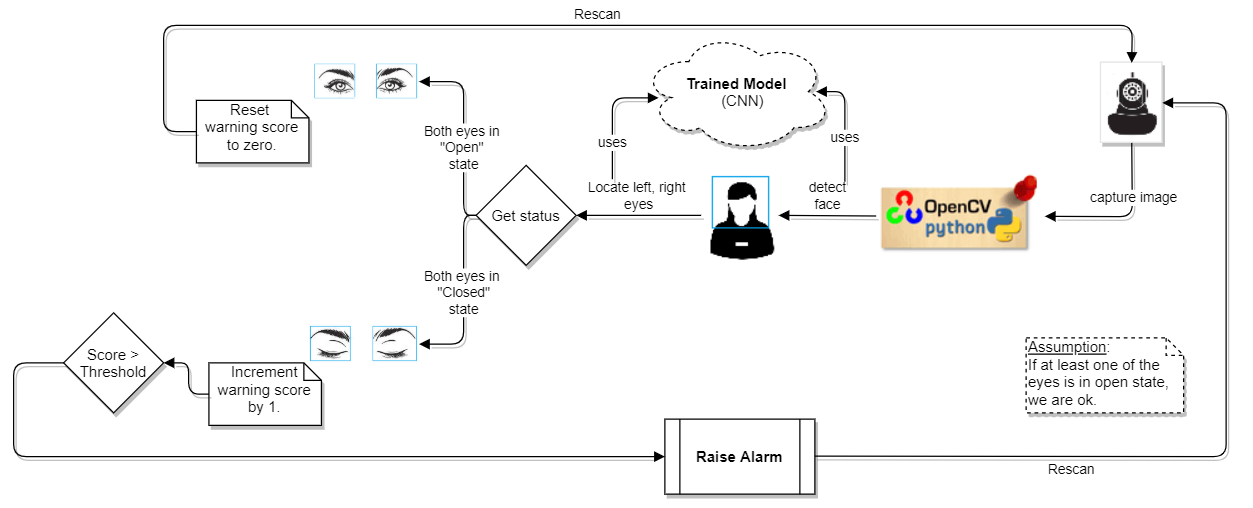
The model developed for the project work uses Python and following open source libraries:

* Numpy – Statistical and Scientific Computing
* Tensorflow – Dataflow and differentiable programming software library
* Keras – Neural network library that runs Tensorflow
* Matplotlib – Graphing library

**Environment used**

* Jupyter Notebook is an interactive computing environment for coding in python.

# Driver Drowsiness Detection System Workflow



*Figure 2 Workflow Diagram*

The workflow steps are detailed below:

* The vehicle is fit with a camera device in an ideal location to be able to capture the driver.
* A python program, **drowsiness detection.py** captures the video images in color and converts to monochrome gray color images.
* From the gray image, the driver’s face, left eye, and right eye are detected with the help of **haar cascade specification** files for face and eyes.
* The captured sub images of eyes are resized to 24 x 24 pixels and the color intensity is normalized.
* Using the pre-trained model, the images are predicted for **open** or **closed** status.
* If both eyes are in a **closed** state, for a prescribed threshold score, the system treats it as a threat and then raises an alarm to alert the driver.
* If the driver is in a state of restlessness and keeps blinking at varied levels of fractions of time, the state of the eye is sensed and the score value is incremented / decremented based on the result of status to be closed / open.

# The Model

With the outstanding performance of deep learning in the field of computer vision, the convolution neural network CNN (convolutional neural networks) shows superior performance on the target detection task. We have used object detection using **Haar** feature based cascade classifiers. It is an effective object detection method proposed by **Paul Viola** and **Michael Jones** in their paper **Rapid Object Detection using a Boosted Cascade of Simple Features[3]** in 2001. It is a Machine / Deep Learning based approach where a cascade function is trained from a lot of positive and negative images. The trained model is then used to detect unseen face / object images.

**How is the face, eyes are located?**

Given a digital image, how does the computer find the location and size of a face or extending the problem further, how can the computer identify objects, in general? To be able to solve this problem, let us think how do we humans identify objects? For example, given an image how do we identify if it has an apple in it? Probably, we look for the red colored, round object. If the objective is to identify a banana, we may look for a yellow colored, curved object. If we are asked to find a face, we may search for the eyes, nose, mouth, ears, etc. which are the features of a face and discard the objects which does not find such features of the face. This is the similar technique we make the computer learn. It is the same idea behind the famous real time object detection technique proposed by Viola and Jones as an algorithm.

**Viola-Jones Algorithm**

The Viola–Jones object detection framework is the first object detection framework to provide competitive object detection performance in real-time proposed in 2001 by Paul Viola and Michael Jones. Although it can be trained to detect a variety of object classes, it was motivated primarily by the problem of face detection. We have used this to detect the status of eyes. Viola and Jones have used 4960 facial images and 9544 non-facial images for the model to be able to get trained and capture patterns to detect faces.

The three key components of Viola-Jones detection algorithm are

* Integral image representation of original image to reduce the computations drastically
* Employing AdaBoost to select critical visual features
* Technique to discard image portions of no importance

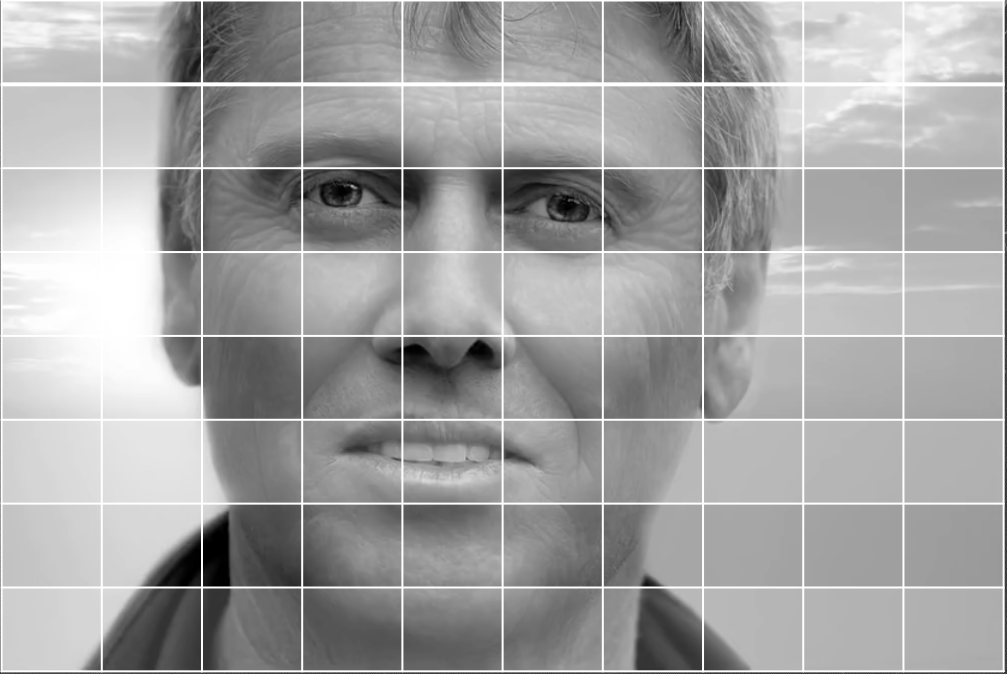
To be able to appreciate the algorithm, we will go through the workflow steps:

Let us assume that we are processing the following image shown below to capture Eyes, nose and mouth.



*Figure 3 Sample Face Image*

The image is divided into multiple rectangular scan regions at different scales.



*Figure 4 Sample Face Image (partitioned)*

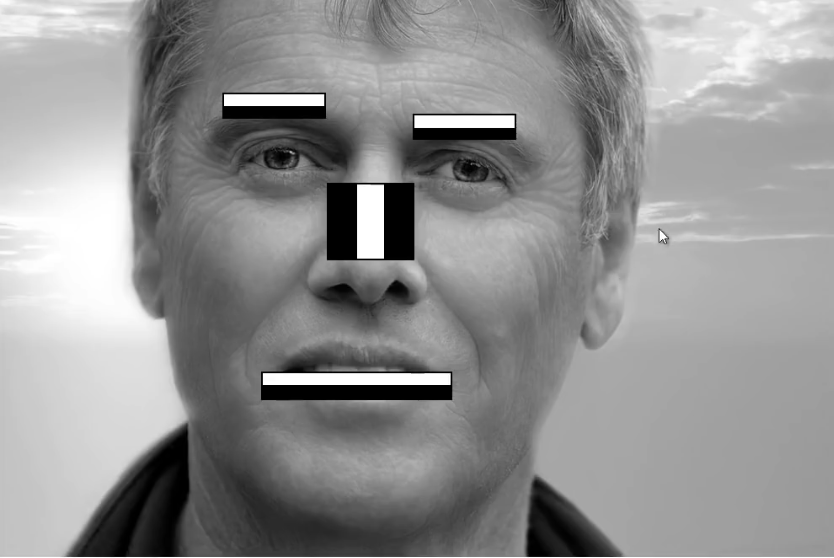
Viola-Jones proposed various filters to capture edges, lines, etc. from an image. They act like convolutional kernels. We can use them to detect facial features like eyes, nose, and mouth, etc. They look as shown below:

|  |  |
| --- | --- |
| **Edge detecting filters** | **Line detecting features** |
|  |  |

The first **edge detecting** filter can be used to detect the eyebrows as that region has darker eyebrows and lighter forehead region and resembles the filter.

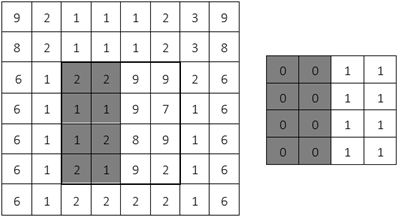
The first **line detecting** filter can be used to capture nose portion of the image as that region in the image has shining lighter region in the center of the nose and dark shaded region sideways and resembles the filter.

The first **edge detecting** filter can be used to capture the lower lip of the mouth portion of the image as it has darker lower and lighter upper portion of the lower lip and it resembles the filter.



*Figure 5 Face Image with matching Filters*

Scanning all the portions of the image at different scales and trying to see for a match among the filters would require huge number of computations. So, Viola-Jones have come up with an alternative representation of the image, termed **Integral Image**. This representation reduces the computations drastically. Let us see how this is done with an example. Let us assume that a portion of the above image can be represented in numeric form of intensities as shown below:



*Figure 6 Image with intensities and a filter*

In order to check if the pattern of 16 pixels selected above matches with the right-side filter, we need to arrive at a numerical representation of both.

Let us compute numerical representation of filter first:

i.e.

Let us compute numerical representation of selected region from the image:

i.e.

So, we say that the selected image pattern matches with the filter by 62.5%.

This way, we need to evaluate for varying scales of images and filters. This involves huge number of computations. So, Viola-Jones have suggested a different representation of the image intensities to reduce the calculations to a constant number irrespective of image and filter sizes.

It works like this. First step is to transfer the above image pattern into its **integral form** as shown below.

**Original image**:

Sum of bright area pixel values = 62

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 9 | 2 | 1 | 1 | 1 | 2 | 3 | 9 |
| 8 | 2 | 1 | 1 | 1 | 2 | 3 | 8 |
| 6 | 1 | 2 | 2 | 9 | 9 | 2 | 6 |
| 6 | 1 | 1 | 1 | 9 | 7 | 1 | 6 |
| 6 | 1 | 1 | 2 | 8 | 9 | 1 | 6 |
| 6 | 1 | 2 | 1 | 9 | 2 | 1 | 6 |
| 6 | 1 | 2 | 2 | 2 | 2 | 1 | 6 |
| 6 | 1 | 2 | 2 | 2 | 1 | 1 | 6 |

Integral form of representation of the above image:

Sum of bright area pixel values

(Using new technique)

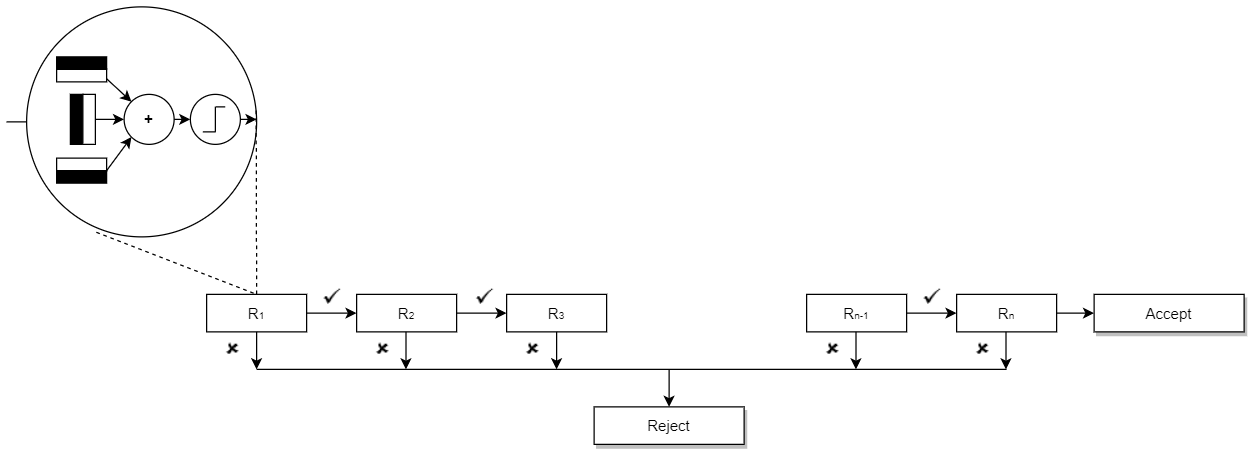
= (133 + 25) – (65 + 31)

= 62

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 9 | 11 | 12 | 13 | 14 | 16 | 19 | 28 |
| 8 | 21 | 23 | 25 | 27 | 31 | 37 | 54 |
| 23 | 28 | 32 | 36 | 47 | 60 | 68 | 91 |
| 29 | 35 | 40 | 45 | 65 | 85 | 94 | 123 |
| 35 | 42 | 48 | 55 | 83 | 112 | 122 | 157 |
| 41 | 49 | 57 | 65 | 102 | 133 | 144 | 185 |
| 47 | 56 | 66 | 76 | 115 | 148 | 160 | 207 |
| 53 | 63 | 75 | 87 | 128 | 162 | 175 | 228 |

This way, irrespective of image size, there will always be only two addition and one subtraction operations to be performed to compute the sum of a rectangle region.

To detect objects of interest, the original image is portioned into rectangular regions and scanned to match with the cascade of filters. If a rectangular region succeeds through all cascades, it classified as a successful match otherwise not. Following figure depicts this idea pictorially:



*Figure 7 Scanning flow over regions of image*

# The model of layers



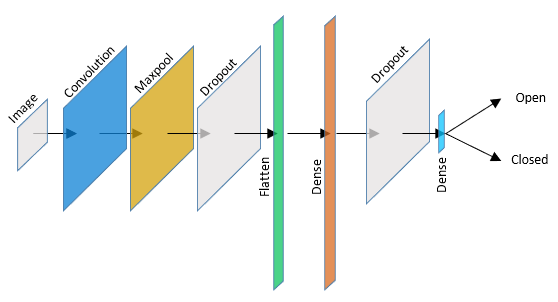


Figure 8 Model I (One Convolution, MaxPool layers)

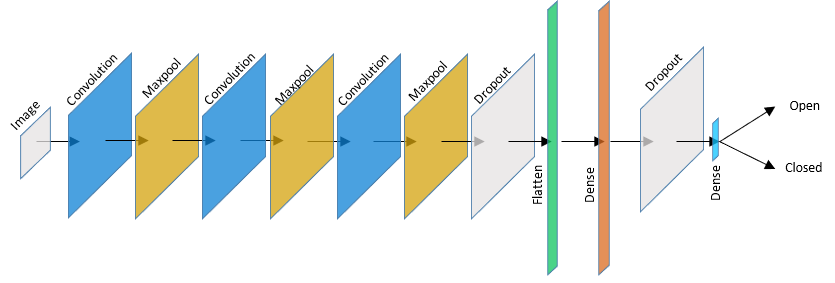


Figure 9 Model II ( 3 pairs of Convolution, MaxPool layers)

**Why did we use convolution layer?**

The convolution layer (CONV) uses filters that perform convolution operations as it is scanning the input with respect to its dimensions. Its hyperparameters include the filter size and stride. The resulting output is called feature map or activation map. It is used to blur, sharpen, emboss, detect edges, etc.

For example, given following image, applying various filters will result in the transformations.4



|  |  |  |
| --- | --- | --- |
| **Transformation** | **Filter** | **Resulting image** |
| Edge Detection | |  |  |  | | --- | --- | --- | | 1 | 0 | -1 | | 0 | 0 | 0 | | -1 | 0 | 1 | |  |
| Edge Detection | |  |  |  | | --- | --- | --- | | 0 | -1 | 0 | | -1 | 4 | -1 | | 0 | -1 | 0 | |  |
| Edge Detection | |  |  |  | | --- | --- | --- | | -1 | -1 | -1 | | -1 | 8 | -1 | | -1 | -1 | -1 | |  |

To give visual cue on the convolution process here is an illustration with and without padding:

|  |  |
| --- | --- |
| **Convolution on a generic image** | **Convolution on a padded generic image** |
|  |  |

By employing convolution layers, we extract high level features of the image.

**Why did we use Max pooling ?**

Max pooling layer helps reduce computational time and complexity by reducing the number of parameters.

**Why did we use Dropout ?**

Dropout is a technique used in neural networks to prevent overfitting the training data by dropping out neurons. It forces the model to avoid relying too much on particular sets of features.

**Why did we use ReLU activation function?**

The Rectified Linear Unit (ReLU) is used primarily because it is simple, intuitive and reduces complex calculations which can become a burden as we are operating on high volume of data intensity values of image data (when compared to other functions like Sigmoid or Tanh).

|  |
| --- |
| **ReLU** |
|  |
| ReLU |

**Why did we use Softmax activation function as the last layer ?**

The Softmax activation function transforms the output from previous layer into probabilities. It is the ideal function for classification.

**How did we decide on number of layers, number of neurons in each layer?**

There are plenty of discussions and tips over the net with differing recommendations. One suggestion is to follow the approaches listed below:

* Experimentation – Says, one cannot analytically calculate and has to perform a controlled experiment and arrive at conclusions
* Intuition – Says, use experience with the domain and modelling to arrive at a conclusion
* Go deep – Says, deep neural networks perform better
* Borrow ideas – Says, get ideas based on research done on similar work and conclude
* Search – Says, study the metrics resulting from the following strategies:
* Random – random configuration of layers and neurons
* Grid – perform systematic search of these parameters
* Heuristic – try a directed search across configurations
* Exhaustive – perform an exhaustive study of all combinations

# Our approach

Neural networks with two hidden layers are sufficient to represent almost any non-linear function. There is no theoretical proof that states requirement of more than two hidden layers.

Following table summarizes the capabilities of neural networks with various hidden layers:

|  |  |
| --- | --- |
| **Hidden**  **Layers #** | **Result** |
| 0 | Network will be capable of representing linear separable functions or decisions. |
| 1 | Can approximate any function that contains a continuous mapping from a finite space to another |
| 2 | Network can represent an arbitrary decision boundary to an arbitrary accuracy with rational activation functions and can approximate any smooth mapping to any accuracy. |

Using too few neurons may result in underfitting as the model may not capture adequate patterns from few neurons relative to the data and its distribution.

Using too many neurons in the hidden layers may result in multiple problems such as overfitting and training time.

**The rules of thumb recommended:**

We have designed two models one with minimum required layers and the other with 3 sets of convolutional and maxpool layers as shown below:

|  |  |
| --- | --- |
| **Model with one pair of Convolution and Maxpool** | **Model with three pair of Convolution and Maxpool** |
|  |  |

# Results from both models:

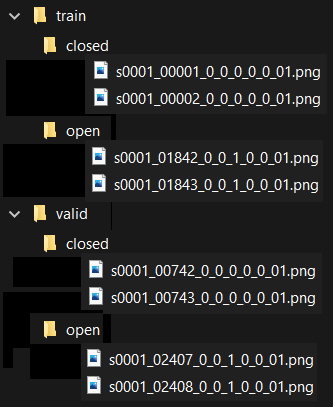
|  |  |
| --- | --- |
| **Model I** | **Model II** |
|  |  |
|  |  |

Clearly from above results it is evident that the model attains a stable of accuracy after 15 epochs and the model is not required to have multiple pairs of convolution and maxpool layers.

# Project deliverables:

1. The packet **DDDS.zip** will have the following files.
2. Driver Drowsiness Detection\_Version 1.0.docx (current document)
3. Driver Drowsiness Detection System.pptx (mid-term presentation document)
4. Data folder (folder holding the training and validation images in folders downloaded from [MRL Eye dataset](http://mrl.cs.vsb.cz/eyedataset))
5. Models folder (folder holding hierarchical model files viz., cnnCat2\_model1.h5, cnnCat2\_model2.h5)
6. Drowsiness\_detection.py (main program which uses the model generated by Model.ipynb)
7. Model.ipynb
8. haar cascade files (folder holding haar like filters in xml format)
9. Alarm.wav (used by main program to generate raise an alarm)
10. Img\_directory\_structure.png (image file used in the documentation of model)

Data folder structure:



**Models folder:**

This folder will have 2 hierarchical model files (.h5 format) and 2 model image files (.png), viz.,

* cnnCat2\_model1.h5, cnnCat2\_model1.png
* cnnCat2\_model2.h5, cnnCat2\_model2.png

**cnn** stands for Convolution Neural Networks

**Cat** stands for Classification of Categories

**Haar cascade files:**

* haarcascade\_frontalface\_alt.xml
* haarcascade\_lefteye\_2splits.xml
* haarcascade\_righteye\_2splits.xml

# Steps to run the Project:

1. Programs are developed in
   1. Python version 3.7.7,
   2. Tensorflow 2,
   3. Keras 2.4.3
   4. Matplotlib 3.1.3
   5. Computer Vision (Open CV) 4.4.0
   6. Pygame 1.9.6
   7. Shell utility
   8. Image files from [MRL Eye Dataset](http://mrl.cs.vsb.cz/eyedataset)
2. Make sure you’ve python running environment along with the libraries is up to date
3. Extract all files from the DDDS.zip to a folder, say **project\_ddds**.
4. Invoke Jupyter Notebook and navigate to the folder **project\_ddds**.
5. Open the python program **model.ipynb**.
6. Run the program.
7. You should see latest files generated in the model folder (two .h5 model files and two .png files).
8. You’ve an option to run one of the two models generated by the **model.ipynb**. To run the first model, edit the program file to make sure it refers to the .h5 file generated from the first model viz., **cnnCat2\_model1.h5**. If you wish to run the second model refer to **cnnCat2\_model2.h5**. It is the following line in the program **drowsiness\_detection.py**.

model = load\_model('models/cnnCat2\_model2.h5')

1. If you are running in a computer with camera device, it starts capturing your image with open/closed eye scores continuously.
2. If the closed eye score is more than 15 it raises an alarm.
3. To stop the program from running, make sure the window showing the video is active and has focus then type the letter ‘q’ to quit.

# Conclusions

Our objective is to implement a small easy to use, fail proof, low budget alert system to detect fatigue of driver during driving. We have studied several systems and implemented and successfully tested an alert system which uses a camera device fixed in front of the driver to detect fatigue by monitoring the state of eyes and warn through a beep.

# Limitations

* Lighting conditions
* Camera quality
* Positioning of camera
* Driver’s cooperation (not wearing dark colored spectacles)

# References:

###### S. H. Fairclough and R. Graham, “Impairment of driving performance caused by sleep deprivation or alcohol: A comparative study,” Human Factors: The Journal of the Human Factors and Ergonomics Society, vol. 41, no. 1, pp. 118–128, 1999.

###### R. Feng, G. Zhang, and B. Cheng, “An on-board system for detecting driver drowsiness based on multi-sensor data fusion using dempstershafer theory,” in Networking, Sensing and Control, 2009. ICNSC ’09. International Conference on Networking, 2009, pp. 897–902.

###### [Paul Viola and Michael J. Jones. Robust real-time face detection. International Journal of Computer Vision, 57(2):137–154, 2004](https://docs.opencv.org/3.4/d0/de3/citelist.html#CITEREF_Viola04)

###### <https://www.wikiwand.com/en/Kernel_(image_processing)>

1. <https://commons.wikimedia.org/wiki/Category:Convolutional_neural_networks>
2. <https://machinelearningmastery.com/how-to-configure-the-number-of-layers-and-nodes-in-a-neural-network/>
3. <https://stats.stackexchange.com/questions/181/how-to-choose-the-number-of-hidden-layers-and-nodes-in-a-feedforward-neural-netw>
4. <https://drowsydriving.org/about/facts-and-stats/>
5. <https://www.nsc.org/road-safety/safety-topics/fatigued-driving>

# Appendix A: GRADUATE CERTIFICATE - Intelligent Reasoning Systems (IRS) Project Proposal

|  |
| --- |
| **Date of proposal:** 10 April 2020 |
| **Project Title:** |
| **Sponsor/Client:** *(Name, Address, Telephone No. and Contact Name)* |
| **Background/Aims/Objectives:** |
| **Requirements Overview:** |
| **Resource Requirements (please list Hardware, Software, and any other resources)**  Hardware proposed for consideration:  Software proposed for consideration:   * Pandas * Numpy * Matplotlib * Seaborn * Sklearn |
| **Number of Learner Interns required: (Please specify their tasks if possible)**  A team of three project members required to architect and implement this system. |
| **Methods and Standards:**   |  |  |  | | --- | --- | --- | | **Procedures** | **Objective** | **Key Activities** | |  | | **Requirement Gathering and Analysis** | The team should meet with ISS to scope the details of project and ensure the achievement of business objectives. | 1.        Gather & Analyze Requirements |  | | 2.        Define internal and External Design |  | | 3.        Prioritize & Consolidate Requirements |  | | 4.        Establish Functional Baseline |  | | **Technical Construction** | To develop the source code in accordance to the design. | 1.        Setup Development Environment |  | | To perform unit testing to ensure the quality before the components are integrated as a whole project | 2.        Understand the System Context, Design |  | | 3.        Perform Coding |  | | 4.        Conduct Unit Testing |  | | **Integration Testing and Acceptance testing** | To ensure interface compatibility and confirm that the integrated system hardware and system software meets requirements and is ready for acceptance testing. | 1.        Prepare System Test Specifications |  | | 2.        Prepare for Test Execution |  | | 3.        Conduct System Integration Testing |  | | 4.        Evaluate Testing |  | | 5.        Establish Product Baseline |  | |  |  | | **Acceptance Testing** | To obtain ISS user acceptance that the system meets the requirements. | 1.        Plan for Acceptance Testing |  | | 2.        Conduct Training for Acceptance Testing |  | | 3.        Prepare for Acceptance Test Execution |  | | 4.        ISS Evaluate Testing |  | | 5.        Obtain Customer Acceptance Sign-off |  | |  |  | | **Delivery** | Deploy the system into production (ISS standalone server) environment. | 1.        Software must be packed by following ISS’s standard |  | | 2.        Deployment guideline must be provided in ISS production (ISS standalone server) format |  | | 3.        Production (ISS standalone server) support and troubleshooting process must be defined. |  | |  |  | |

|  |
| --- |
| Team Name: **GROUP 1** |
| Project Title (repeated): **Driver Drowsiness Detection System** |
| System Name (if decided): |
|  |
| Team Member 1 Name: Anirban Kar Chaudhuri |
| Team Member 1 Matriculation Number: A0108517H |
| Team Member 1 Contact (Mobile/Email):  Mobile: 86118180  Email: anirban.karchaudhuri@gmail.com |
|  |
| Team Member 2 Name: MARADANA VIJAYAKRISHNA |
| Team Member 2 Matriculation Number: A0178453W |
| Team Member 2 Contact (Mobile/Email):  Mobile: 93896379  Email: mvskrishna@yahoo.com |
|  |
| Team Member 3 Name: Putrevu Manoj Niyogi |
| Team Member 3 Matriculation Number: A0213557E |
| Team Member 3 Contact (Mobile/Email):  Mobile: 94575890  Email: manojniyogi@yahoo.com |
|  |
| Team Member 4 Name: Sivasankaran Balakrishnan |
| Team Member 4 Matriculation Number: A0065970X |
| Team Member 4 Contact (Mobile/Email):  Mobile: 97379441  Email: bsivaa@gmail.com |

**Team Formation & Registration**

|  |  |  |
| --- | --- | --- |
| **For ISS Use Only** | | |
| **Program Name:** | **Project No:** | **Learner Batch:** |
| **Accepted/Rejected/KIV:** | | |
| **Learners Assigned:** | | |
| **Advisor Assigned:**  Contact: Mr. GU ZHAN / Lecturer & Consultant  Telephone No.: 65-6516 8021  Email: [zhan.gu@nus.edu.sg](mailto:zhan.gu@nus.edu.sg) | | |

# Appendix B : Individual Project Report

|  |
| --- |
| **Project Title:** Driver Drowsiness Detection System |
| **Team Member 4 Name:** |
| **Team Member 4 Matriculation Number:** |
|  |
|  |
| **Learn new technologies:**   1. I learned and became familiar in web development using html and css 2. Learnt deployment of micro web services using Python Flask library 3. Explored and discovered ways for writing efficient codes in Python as well as testing runtime |
| **My Involvement in the Project** |

|  |
| --- |
| **Project Title:** Driver Drowsiness Detection System |
| **Team Member 3 Name:** Putrevu Manoj Niyogi |
| **Team Member 3 Matriculation Number:** A0213557E |
|  |
|  |
| **Learn new technologies:**   1. I learned and became familiar in development using Python and flask frame work. 2. Acquired the working knowledge of Python libraries applicable for Machine Learning projects. 3. Learned and explored the data analysis / data discovery tools (Orange, Panda – Python Library). 4. Upon acquiring the above knowledge, looking forward to work in machine learning / data analysis related projects. |
| **My Involvement in the Project** |

|  |
| --- |
| **Project Title: Driver Drowsiness Detection System** |
| **Team Member Group 2 Name:** MARADANA VIJAYA KRISHNA |
| **Team Member Group 2 Matriculation Number:** A0178453W |
|  |
|  |
| **Technologies learned & applied:**   1. I had picked up and familiar with web development using Python and its flask framework. 2. Acquired the working knowledge of Python libraries applicable for Machine Learning projects. 3. Learned and explored the data analysis / data discovery tools (Orange, Panda – Python Library). 4. Also learned and impressed with working and implementation of ASR, speech recognition and natural language processing technologies. 5. Upon acquiring the above knowledge, looking forward to work in machine learning / data analysis related projects. |
| **My Involvement in the Project** |

|  |
| --- |
| **Project Title:** Driver Drowsiness Detection System |
| **Team Member 1 Name:** Sivasankaran Balakrishnan |
| **Team Member 1 Matriculation Number:** A0065970X |
|  |
|  |
| **Learn new technologies:**   1. I had picked up and familiar with web development using Python and its flask framework. 2. Acquired the working knowledge of Python libraries applicable for Machine Learning projects. 3. Learned and explored the data analysis / data discovery tools (Orange, Panda – Python Library). 4. Also learned and impressed with working and implementation of ASR, speech recognition and natural language processing technologies. 5. Upon acquiring the above knowledge, looking forward to work in machine learning / data analysis related projects. |
| **My Involvement in the Project** |