Deep Learning for Driver Drowsiness Detection

MSc Research Project
Data Analytics

Ravjyot Singh Duggal
Student ID: x21128901

School of Computing
National College of Ireland

Supervisor: Dr. Christian Horn

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Ravjyot Singh Duggal
Student ID:	x21128901
Programme:	Data Analytics
Year:	2022
Module:	MSc Research Project
Supervisor:	Dr. Christian Horn
Submission Due Date:	15-12-22
Project Title:	Deep Learning for Driver Drowsiness Detection
Word Count:	6092
Page Count:	20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Ravjyot Singh Duggal
Date:	15th December 2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Deep Learning for Driver Drowsiness Detection

Ravjyot Singh Duggal
x21128901

Abstract

There is a state between awake and sleep which is drowsiness. Driver fatigue is the most common cause of road accidents in India. One of the recent studies carried out by a journal [1] states that the India accounts for 11% of casualties around the world in road accidents even though the country has only 1% of the global vehicle count. In contrast, the global deaths from road accidents is 11%. A system needs to be in place to overcome this problem. This research has tried to create a solution to alleviate this problem not only in India but in the entire globe. A model was developed with deep learning techniques for detecting driver drowsiness by facial points obtainment. The images were recorded at 15 frames per second (fps), state of the art Haar-cascade technology was used for detecting open and closed eyes. Convolutional neural network has been used here for detecting driver drowsiness particularly eyes blink, open and closure rate, and the face position.

1 Introduction

There are a large number of road accident related fatalities in India and one of the causes is drowsiness of driver. These number are on the rise with the passage of time, the vast majority of the accidents are caused on road are due to driver mistakes like over drowsiness while driving and over speed on the roads. Currently the automobiles manufacturers are facing this issue while manufacturing their vehicles. Several other factors which caused road accidents are consumption of alcohol, the conditions of weather, negligence on road by driver and sometimes invisibility of road in fog conditions. There are many devices in place by traffic police to see whether the driver is drunk or not these devices will check the level of drink a person has consumed, however, the system to catch the driver drowsiness is currently not in the market and there must be some technology to judge whether the person is drowsy or not. The system to analyse whether the person is drowsy or not is challenging for the old card which are already on road from long time, many car manufacturers lie Volvo, BMW and Mercedes are already worked upon the technology for capturing driver drowsiness, but these cars are expensive and not all people can purchase these. Refer Figure 1

There is a need for having a good system which can detect driver drowsiness in old cars as well using some technology like using the facial points of a person, the mouth movement, and the behaviour of a person. Although the driver will be conscious when he/she is driving but a single second can alter the situation permanently. When the driver is drowsy, the car steering behaviours will get changes and there will be frequent

[1] https://www.aa.com.tr/en/asia-pacific/india-tops-world-in-road-deaths-injuries/2425908

Figure 1: A speedometer from a high end car

changing in lanes of road, a long journey which is continuous, and no sleep are the major causes of drowsiness in driver which leans to road accidents.

There can be many solutions for avoiding road accidents like taking regular breaks while on the road, sound system and types of alarm are in place that can give reminder to the driver to take a break while driving the vehicle on the road. In this system, it will be hard to know when the driver is drowsiness or not as its drowsiness can happen at any time. road accidents are generally caused at night-time as the drowsiness is more frequent at that time, some driver's pattern can be used to predict the driver drowsiness like frequent opening of mouth for yawning, frequent closure of eyes, the expression of face, person's voice and lastly the car steering pattern. A system can be in place to alert the driver if the focus of driver is deviated and if the pattern of the steering is getting changed frequently. The severity of accidents on roads are a vital reason for declining in GDP of a country since the average age group which is affected by the road accidents in today's world are in between 18-29. The morbidity and mortality rate are very high in the road accidents cases and the reason is driver drowsiness.

Figure 2: Image of an open eye

A survey is conducted by National Sleep foundation which did the analysis on the cases of road accidents, the primary cause is driver drowsiness, 52% of the youngsters who have drowsiness in which 17% of the people fall asleep while driving. In this regard many car manufacturers have developed a system in their card like DAC which stands for driver alert control, in this application it predicts the driver drowsiness based on a

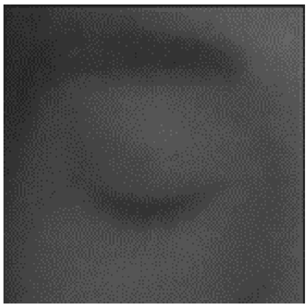


Figure 3: Image of a closed eye

camera which is mounted in the vehicle. Mercedes also has a similar technology in place for the cars which is attention assist system. In this technology, if the driver steering pattern is same then an alert is there to awake the driver. All these systems are designed for high end cars and not for the existing cars on roads and drove by common public. In this work, an effort has been made to build a system for capturing the driver drowsiness by checking the person eye and mouth movement. This research has used deep learning techniques for building the models and constraints on budget. A camera can be mounted in the person's car and model will check on real time if the driver is drowsy or not.

The main goal of this project is to answer the below research question- "How effectively can deep learning be utilized to detect driver drowsiness?"

This can be fragmented in to the following objectives of this research:

1. Train the models from MRL Eye Dataset.
2. Avoid Overfitting.
3. Achieve highest possible Recall and F1 scores.

The dataset is taken from MRL eye dataset which contains 84K+ photos. Various deep learning models have been used like Convolutional neural networks, Recurrent neural networks, Long short term memory.

2 Related Work

Sometimes back the driver drowsiness detection is started. It is a vital concern as there are rising numbers of accidents on road in India. There are many approaches proposed which relates to deep learning which is concerned. This research is categorized into 3 topics primarily which are firstly, Driver drowsiness pattern, secondly, the aspect related to vision and deep learning and lastly, Approach related to physiological - sensors. The aim of this research is focused on Vision and deep learning aspect.

The content sections of your report should of course be structured into subsections. Note that here there are 2 subsections subsection 5.1 and subsection 5.2.

2.1 Haar Casacade based strategies

(Suryawanshi and Agrawal; 2020), in this research, for detecting the face using the camera, local binary pattern are used. Author made efficient use of state of the art Haar cascade technology. adaboost has been used for capturing eye movement, the overall accuracy author has achieved is 90%. There is only one drawback in this approach the deep learning model is not able to classify the images in which there is low light is involved. Driver's face in the low light is not able to capture by the model, the algorithm in this paper needs to be changed.

(Choudhary et al.; 2022) uses CNN along with Haar cascade for detecting the eye movement. OpenCV has been used for detecting driver's state. The classifier is in place for classifying whether the eyes are closed or open. Many characteristics are still needs to be extracted from person's face for making the model predicting accurate.

(Kumar et al.; 2021)in this research, Haar cascade is used along with OpenCV, detection of eyes is captured by Haar cascade whereas OpenCV is used with the help of webcam for eyes and face detection. In detecting accident or crash of the vehicle, sensor which is Piezoelectric has been used in this paper. Model is doing well in many parameters like recognizing the eye time was just two seconds, the detection of driver sleepiness was 4 seconds. The only drawback was the technology was old which was used in this paper.

(Kamarudin et al.; 2019) in this paper, Haar cascade along with eye aspect ratio is used. For deploying the system, author has used Rasberry Pi. the average eye aspect raio is 0.337 when the eyes are open, when the eyes are closed 0.143. Raberry Pi 3 has used in this research, eye respect ratio will decide the driver drowsiness.

(Chowdhury and Jos; 2022) - in this research paper, CNN along with Haar casacde is used for detecting movements of eyes. the counter will get increased if the driver is not drowsy, as soon as the counter increases the threshold value, the model classification will set to drowsy and an alarm will get triggered. Numerous machine and deep learning methods were used in the development process, the accuracy highest achieved is 72%. The drawback in this research is that the model fales to work properly in low light conditions.

(Farahnakian et al.; 2021) In this author used Haar cascade for detecting eyes along with CNN which are on pre-trained model, RLDD dataset was used in this research. In the dataset, 60 people photos and videos were used in which 51 were men and 9 were women. The people are in the age between 21 and 58. Author achieved the accuracy of 99% for the eyes and 96% for the mouth. The overall accuracy achived is 92%. In this research the behaviour of the driver is not captured properly. There should be new framework for detection.

(Pulluri and Ranjana; 2022) - in this paper, author has used Haar casacade technology for face detection. OpenCV was used for recording real time face and eye aspect ratio is used for checking eye present state - close or open. Dilib by HOG has been used for creating face landmarks. 79% is the overall accuracy achieved by the model when the glasses were used by the person, by using Dilib, the overall accuracy is 90% which is achieved by the author. In this research, it is missing the hybrid system, hence it is needed to enhance the model.

(Bano et al.; 2021) - author has used CNN as the primary algorithm for the use, haar cascade has also been used for detecting the face and in particular eyes. The author has used layers, loss function and optimizers for efficient use in the model. By using CNN with haar cascade, author have received accuracy of 98%. Author has used various evaluation matrix like precesion, recall and F1 score. The loss received by the model is 0.0612 and for the validation loss, author has achieved is 0.10.

(Sharma et al.; 2021) - In this research paper, author have used local binary pattern histogram, the author has also used haar cascade for detection of driver drowsiness. For detecting the face, LBPH was used, when author used plain background 92.3% accuracy is achieved, whereas when there is an exhaustive background, the accuracy is dropped to 87%. The overall accuracy achieved is 91%. There should be sensitivity as the evaluation matrix.

(Amangi et al.; 2022) - In this reseach paper, CNN has been used along with OpenCV, Haar cascade and Dlib. For detecting facial landmarks, Haar cascade is used. For extracting the regions of eyes, the author gets the horizontal projection and the position geometrical of eye were used. With complexity function, dynamic threshold has been used. CNN has the accuracy of 97% whereas ANN achieved the accuracy of 68%. Sensitivity should be use as the evaluation matrix.

2.2 Vision Procedures and Deep learning

(Magán et al.; 2022)- in this research paper, use of advanced driving assistance system is in place which focuses on the detecting the drowsiness of driver. CNN and RNN have been used in this paper. On the training dataset, author has achieved accuracy of 65% whereas on the test dataset, author has achieved 60% accuracy. Author has used the fuzzy logic which stand out from rest of the paper as it reduces the risk of alarm, which is false, in this paper the overall specificity achieved is of 93%, model only classified 22 videos correctly out of 61 videos.

(Suresh et al.; 2021) - in this paper, author has used dataset MRL eye which has almost 84000 images of eyes, deep learning algorithm has used for classifying whether the person is drowsy or not. Overall accuracy of 86.05% has been achieved, F1 score is 84% for eyes closed and 87% is the score when the eyes are open. Transfer learning can be used to increase the system performance.

(Nandyal et al.; 2022) - in this paper, author used KLT-ViolaJones which is used for locating the facial features. Region of interest has been used for extracting the feature points. Histogram oriented gradient has been used for detecting drowsiness of driver. L-DenseNet is more affected than CNN. In evaluation matrix, the overall accuracy achieved is 98%, sensitivity achieved is 91%, F1 score was 97%, specificity achieved is 92% and the precesion achieved is 98%. The accuracy of classification was not good. Facial points should be increased.

(Dey et al.; 2019) - in this paper, a pre trained model is used which is HOG. Author used support vector machines for extracting face features like mouth, nose, and eye positions. Eye aspect ratio is also analysed. Author has analysed length of nose and opening of mouth ratio. In hardware, Logitech c930 web camera has been used for collecting the images. Overall sensitivity achieved by the FLDA model is 89% whereas 95% sensitivity achieved is for support vector machines. Author has not used accuracy as the evaluation matrix.

(Pachouly et al.; 2020) - in this research paper, CNN have been used for checking the movement of eyes. Dlib is used for extracting features of the face. OpenCV library has been used in this paper for the gui purpose. Author has also used Haar adaboost for detecting the face. 240x240 Images used for training the model. Author have received overall accuracy of 94% by using the test dataset. In this paper where there is dark facial points was not taken into consideration by the model, author should have used CNN for improving the model.

(Chowdhury et al.; 2021) - in this research paper, CNN have been used for detecting face of the person. Haar cascade has been used for extracting the eye features. The best part in this paper is that the model is accurately classifying the person is drowsy or not even if driver is wearing any glasses, the model is accurately classifying the if the person is drowsy or not. Overall accuracy achieved is 97% and this is for the dataset validation and if the driver is wearing any glasses the overall accuracy achieved is 92%. Author has used MRL dataset, in real time, the accuracy achieved is 94.5%. If there were any low light conditions, then the model is not able to classify driver drowsiness. CNN particularly 3D should be used for improving the model.

(Cheerla et al.; 2022) - in this research paper, HOG has been used for detecting the objects. The dataset used by the author consists of 68 landmarks. The percentage of drowsiness is calculated, if the percentage which is calculated is less than 66 that means that driver is not drowsy otherwise the person is drowsy. Author should have used sensitivity in this paper to check driver drowsiness.

(Sinha et al.; 2021) - in this research paper, author have used many different algorithms for detecting the drowsiness of driver some of them are Yolo v3, DLib and Viola Jones. RNNs and CNN were used for detecting drowsiness of driver. The image pixels were 511x423. In this paper, MC-KCF is the best running model and it outperformed MTCNN with accuracy of 95%. The camera which is used here is of bad quality, good quality cameras should be use for capturing the face of the person.

(Phan et al.; 2021) - in this research paper, the author used two solutions which are different in detecting the drowsiness of driver. In approach first, facial landmarks were used in detecting the eye blink. This solution is solely based on threshold of driver. In the second approach, some deep learning architectures were used like MobileNetV2 and ResNet-50VT, later the video of driver is analysed, in the video, each frame is detected. Author achieves overall accuracy of 98%. The system which is built does not capture the video in live, for capturing information in real time, big data analysis should be used.

(Nandhini et al.; 2022) - in this research paper, use of MobileNetV2 has been used, RNN and CNN has also been used to check if the driver is drowsy. Rasberry Pi has been used for tracking the person in realtime for driver drowsiness. ReLu activation function has been used for model training. Overall accuracy of 92% has been achieved by the author and for validation the accuracy achieved of 88%. For deployment, author should decrease the size of the model.

2.3 Results and Conclusions

To conclude, in all the research papers one thing is common that authors have used CNN with Haar cascade for detecting driver drowsiness.

3 Research Methodology

In this research, various methodologies are used on the MRL eye dataset for solving this business problem. After reading vast literature review, it is worth noted that many authors have used deep learning techniques for solving this problem. CRISPDM was primarily used as the methodology in various research paper as explained in the literature review.

In this project, various python libraries like OpenCV, Keras and Tensorflow were used to build a system for detecting the open and close eye of the driver and then alert the driver if drowsiness detected. If driver closed the eyes, then the system will inform the driver immediately. OpenCV library has been used for monitoring and collecting driver's

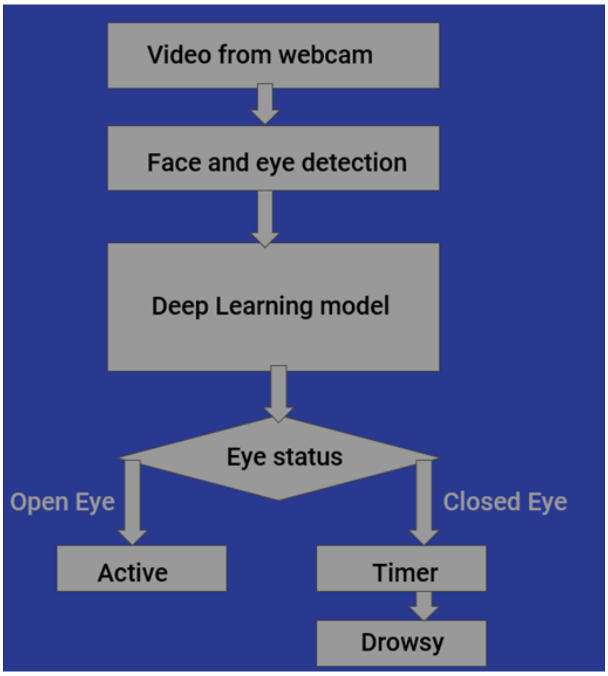


Figure 4: Process Flow Diagram

live image via webcam and it feeds directly to the deep learning model and then model classification will take place to see if the driver eyes are closed or opened. CRISP-DM is used for implementing a project related to mining of data which contains 6 primary phases as discussed in Fig 4. Some steps include - gathering of data, exploratory data analysis, techniques related to modelling and the last step includes the deployment of the final model for the usage.

Configuration of the system - Operating System - Windows 11 Home Single Language, Model Name - Dell Alienware M15R6, Processor - Intel core i7 RAM - 16GM, SSD - 500GB, GPU - 8GB Nvidia 3060

3.1 Understanding of Data

This is the second part of CRISP-DM which involves data collection and to check if the data collected is aligned with the actual requirement of the business. MRL dataset has been used here, dataset is divided into two classes which are - 1. Dataset - Closed Eye - This dataset contains 41946 images 2. Dataset - Open Eye - This dataset contains 42952 images

In the above-mentioned dataset - The dataset contains many different eyes with different angles of the eyes like the photo is taken from right, left, above and below angle for best training of the model. People with wearing specs are also considered for training the model. Both of the dataset includes the eyes of both female and male volunteer which have glasses on and off. All the images are having the size non-uniform and .png is the format of the image. All the images are 75x75 but the images pixels are varying in the dataset. In data preparation part the pixels size is resized. The size of the complete dataset is 453MB.

Figure 5: CRISP-DM

3.2 Data Preparation

Preparation of the data is the most crucial part of CRISPDM methodology. In this research project, only eye dataset has been picked for training the model where 0 represent closing the eye whereas 1 represents opening of the eye.

Data is prepared as follows - Firstly, the data is splitted into two parts - open and close. Shutil library is used for splitting the data and then copy the data into their folders. Then, with the help of Shutil library further split has been performed based on folders. These folders were train and validation folder. Training set have 95% of the total data and 5% for validation. Testing have been performed live; the data has been captured through openCV on local machine.

3.3 Modelling

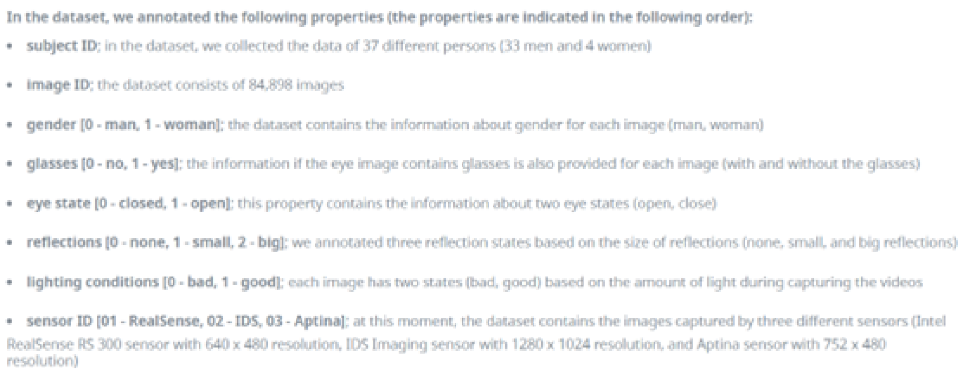
Building model is the primary step in methodology - CRISP-DM, in this step the actual model is built and tested on live data. For building the model, convolutional neural network has been used.

3.3.1 Convolutional Neural Network

CNN is a very efficient deep learning technique which accepts an image as input and then it converts into feature map which is low dimensional which is followed with pooling and lowering further the pixels of image and then flattening for reducing the map of feature in vector which is single and is also the neural network input. In the layer, which is hidden, each neuron have weights attached for differentiating the objects in the images which are learning for classifying each of them from each other.

The CNN's are able to capture very vital features with dataset training which given upper hand if compared to other methods which have the efforts manual. CNN's have a capability of extracting many facial features from an image which is the input without requiring pre-processing of the image with high accuracy.

Figure 6: Image Samples from MLR Eye Dataset



In the dataset, we annotated the following properties (the properties are indicated in the following order):

- **subject ID**; in the dataset, we collected the data of 37 different persons (33 men and 4 women)
- **image ID**; the dataset consists of 84,898 images
- **gender [0 - man, 1 - woman]**; the dataset contains the information about gender for each image (man, woman)
- **glasses [0 - no, 1 - yes]**; the information if the eye image contains glasses is also provided for each image (with and without the glasses)
- **eye state [0 - closed, 1 - open]**; this property contains the information about two eye states (open, close)
- **reflections [0 - none, 1 - small, 2 - big]**; we annotated three reflection states based on the size of reflections (none, small, and big reflections)
- **lighting conditions [0 - bad, 1 - good]**; each image has two states (bad, good) based on the amount of light during capturing the videos
- **sensor ID [01 - RealSense, 02 - IDS, 03 - Aptina]**; at this moment, the dataset contains the images captured by three different sensors (Intel RealSense RS 300 sensor with 640 x 480 resolution, IDS Imaging sensor with 1280 x 1024 resolution, and Aptina sensor with 752 x 480 resolution)

Figure 7: Data Dictionary Captured from Official Website of MRL Dataset

3.4 Evaluation

The evaluation phase consists of evaluating the performance of the model, the major evaluation metrics used in this project is accuracy since in this project we are capturing live data and feeding into the model so accuracy plays a vital role in evaluating the model performance. Sensitivity has also been used as the vital evaluation matrix as if the driver is feeling drowsy and the model doesn't alert the driver then it causes a road accident. 16 deep learning models were made and out of it 5 best performing deep learning models were discussed in this paper the best models in terms of overfitting (the difference between the categorical accuracy of train and test dataset). Their findings are further evaluated with the help of these evaluation matrix. The reviewing of the process involves various steps which are taken for having a detailed overview of the tasks which were executed in the project implementation for checking if there any part which is missing which can be considered quality assurance part.

4 Design Specification

For solving the question, there is a need of good architecture which is well designed. For the design specification. refer the Figure 8 where the complete design is implemented. All the photos in the dataset are of .png format.

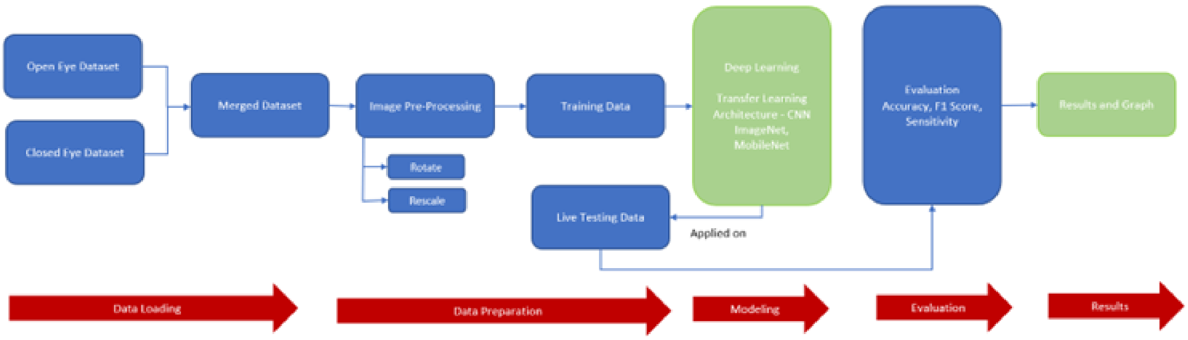


Figure 8: Driver Drowsiness System - Design Architecture

In the implementation, firstly both the dataset which are open eye and closed eye are merged and once it is merged then the image pre-processing is done in which image is rotated by 20% or 0.2 which is anti-clockwise. The argument given to the ImageData-Generator function is rotation_range = 0.2. Rescaling of images plays a vital role in the image pre-processing; rescaling of images took place as the images are in non-uniformity. There is a performance issue as it will give the results less accurate hence there is a need for rescaling the images for having the same base size. In this research 3990 images were having the size distorted and is scaled to 1. /255 ratio. This ratio can be changed as per the further model training and observations. Next, the data is split into training and validation data for training the model. The processed data is split based on train and validation dataset for model building process. In the train data, the train_datagen is used here, the target size for creating the train dataset is 80, 80, the batch size is 8, the mode of class will be categorical, and training will be the subset. In the modelling phase, deep learning architecture has been used, Convolutional neural networks have been used, ImageNet and MobileNet has been used. Transfer learning played a vital role in training the models. ImageNet is used as a weight in training the model. The activation function used are Relu and SoftMax. Once the models are trained, the next step is evaluating the model performance, Accuracy, sensitivity, and F1 Score is the evaluation matrix used in this research project. In the last stage that is results, we check the results and the graphs of the model to see if the model is accurately predicting everything it is trained for. The model is trained on many epochs and it took nearly an hour for tuning the model on the basis of different epochs, different number of epochs has been used in order to train the model better.CNN along with Haar cascades were used to capture the face of the person. Relu and Softmax were used as the activation function in training of the model. Accuracy and sensitivity were the primary metrics used in evaluating the model.

5 Implementation

The data is captured form MRL dataset. All the images are in the format of .png, the images are later resized with the help of image data generators, images are rescaled, images are then split into train and validation dataset. The split between train and validation dataset is 80-20. The images which are pre-processed are then added in data frames for increasing the performance of the model by learning the images which are having size diversity, brightness high, blurring, many different angles and lastly blurring. With the help of this, the problem of overfitting in the dataset of images is avoidable.

5.1 Pre-Processing of Images

The dataset of closed eye contains 39,946 images which has size varying and open eyes contains the data of 40,952 images. The images are then transformed into a numpy array which is further utilized by the architecture. Before doing this, the images are augmented as the phase of pre-processing using Image data generator. There are steps involved in the pre-processing which are as follows:

5.1.1 Rotation

In this activity, rotation of images took place by rotating the images by 20% or 0.2 which is anti-clockwise. The parameters are further fed in the ImageDataGenerator function for rotation of the images. The parameter used here is rotation_range = 0.2. The rotation of the images is done in both train dataset as well as validation dataset. The function name for rotating the images is train_datgen.

5.1.2 Rescaling

In this activity, rescaling of images took place as the images are in non-uniformity. There is a performance issue as it will give the results less accurate hence there is a need for rescaling the images for having the same base size. In this research 3990 images were having the size distorted and is scaled to 1./255 ratio. This ratio can be changed as per the further model training and observations. The rescaling is done with the help of ImageDataGenerator function and the parameter is rescale which was given to the function and the result of ImageDataGenerator is further stored in the variable train_datagen. The recalling of the images is done on both train as well as validation dataset.

5.1.3 Train Validation Split

In this activity, the processed data is split on the basis of train and validation dataset for model building process. in the train data, the train_datagen is used here, the target size for creating the train dataset is 80, 80, the batch size is 8, the mode of class will be categorical and training will be the subset.

```
In [7]: batchsize=8

In [8]: train_datagen= ImageDataGenerator(rescale=1./255, rotation_range=0.2,shear_range=0.2,
            zoom_range=0.2,width_shift_range=0.2,
            height_shift_range=0.2, validation_split=0.2)

        train_data= train_datagen.flow_from_directory(r"C:\Users\ravjy\Documents\College Subjects\Semester 3\Research Project\driver dro
                                    target_size=(80,80),batch_size=batchsize,class_mode='categorical',subset='training' )

        validation_data= train_datagen.flow_from_directory(r"C:\Users\ravjy\Documents\College Subjects\Semester 3\Research Project\driver
                                    target_size=(80,80),batch_size=batchsize,class_mode='categorical', subset='validation')

        Found 64719 images belonging to 2 classes.
        Found 16179 images belonging to 2 classes.
```

Figure 9: Train Validation Split

5.2 Deep learning architecture

In this research, there were several research papers which were studied for choosing the best model for the dataset. The dataset is divided into 80-20 ratio 80 is for training and 20 is for validation dataset for solving the objectives of research. Convolutional neural network along with Haar cascade has been used for training the model effectively. 16 deep learning models were made using computational power of the system. Out of it, 5 best performing deep learning models were discussed in this paper the best models in terms of overfitting (the difference between the categorical accuracy of train and test dataset) The best one later got selected and stored in a .h5 file which can later use for live test data and for deployment in various applications like Raspberry Pi, Arduino and Flask rest API.

5.2.1 Convolutional Neural Network

After the image pre-processing is completed, the images are fed to the CNN architecture.

5.3 Model Building Process

The dataset is a huge dataset, a sophisticated techniques is used which is transfer learning where in this research a pre trained model is used but later that model is customized according to our needs. The models used in this research were trained on relatively large datasets and the transfer weights which were gained and learned through many hours of training with the help of GPUs which are very highly configured and are top end.

5.3.1 Model Requirements

In this research, transfer learning is used. InceptionV3 is used as the primary function and it is a V3 architecture, the arguments given in the function were - include_top = False which means whether to include the fully connected layers at the top as the last layer of the network by default is set to true but for our case, we don't want the fully connected layers at the top. The second argument is weights which is imagenet, this is by default and the last argument is input_tensor which is used for sharing the inputs between many networks which are different to each other. The input shape is given, the height is 80, the width is 80 and the channels is 3 (RGB).

```
bmodel = InceptionV3(include_top=False, weights='imagenet', input_tensor=Input(shape=(80,80,3)))
hmodel = bmodel.output
hmodel = Flatten()(hmodel)
hmodel = Dense(64, activation='relu')(hmodel)
hmodel = Dropout(0.5)(hmodel)
hmodel = Dense(2,activation= 'softmax')(hmodel)

model = Model(inputs=bmodel.input, outputs= hmodel)
for layer in bmodel.layers:
    layer.trainable = False
```

Figure 10: Model Requirements

5.3.2 Building the models

The input shape of the Keras Inception v3 must be of the shape (299, 299, 3) by default. But the size of most of the raw images is around (80,80,3). Thus, customization is needed so that the input shape can be changed from (299, 299, 3) to (80, 80, 3).

That is why the parameter include_top was set to be False in order to remove the fully connected top layer. Then the output of 'bmodel' is stored in the new variable 'hmodel'. The output of the keras tensor 'hmodel' at this point of time is (1, 1, 2048). To convert this multi dimensional matrix/ array in to one dimensional array, the function flatten needs to be used. After its application, the shape of hmodel becomes (2048). This transformation in to one dimensional array is required as we need to feed this output to the dense layer, which accepts only this shape of inputs. Final dense layer contains 2 neurons as we need to classify the images in to two categories: "Open eye" and "Closed eye".

Then Keras function 'Model' is used along with the input and output parameters to instantiate the deep learning model. Since the pre-trained weights of imagenet have been used, another Keras function layer. Trainable is set to False so that the weights do not get overwritten.

The above stated architecture is same for all the models. A total of 16 models were used but only 5 best performing ones (in terms of lower differences between the categorical accuracies between the train and test datasets) are being explained below. The screenshots of these models and further elaboration is given in section 6 in a step by step manner.

5.3.3 Model 1

There was a dense layer (with 64 neurons) between the flatten and the final dense layer. A dropout is also used between the two dense layers for regularization. The value of parameter 'patience' in the EarlyStopping function of Keras was set to 7.

5.3.4 Model 2

This model was an exact replica of model 1. The only change was in the value of parameter 'patience'. It was set to 14 in this case. The value of parameter 'patience' in the EarlyStopping function of Keras was set to 7.

5.3.5 Model 3

Dropout of 0.25 was introduced after the flatten layer. The number of neurons in the dense layer were increased to 128. The value of parameter 'patience' in the EarlyStopping function of Keras was set to 7.

5.3.6 Model 4

BatchNormalization was introduced after the flatten and dense layers.

5.3.7 Model 5

4 dense layers with varying number of neurons (in decreasing count) were added. High dropouts (0.5) were introduced after all the dense layers except the last one.

6 Evaluation

The raw MRL Eye dataset contains a total of 37 folders. There are a total of 84,898 images in these folders. After the pre-processing the train dataset contains 40,952 images in the 'Open Eyes' folder whereas 39,946 images in the 'Close Eyes' folder. The test dataset contains 2,000 images each in the 'Open Eyes' and 'Close Eyes' folders. It is essential to gauge the performance of the models that were built on the validation as well as test datasets.

Categorical accuracy, Recall, Precision and F1 Score were used as evaluation metrics of choice. However, the values of Recall and Precision were the exact same for the corresponding categorical accuracy values for all the models.

The base model's architecture has already been described in section 5.3.2.

6.1 Experiment 1

There was a dense layer (with 64 neurons) between the flatten and the final dense layer. A dropout is also used between the two dense layers for regularization. In the "Early Stopping" criteria of Keras, the parameter 'patience' was set as 7. However, the difference between the categorical accuracies on train and test datasets was 12.46%, thereby indicating the presence of overfitting.

6.2 Experiment 2

This model's architecture was the same as model 1. Only change was the value of the parameter 'patience', which was set to be 14. However, the difference between the categorical accuracies on train and test datasets was higher (13.06%) as compared to experiment 1.

```
bmodel_1 = InceptionV3(include_top=False, weights='imagenet', input_tensor=Input(shape=(80,80,3)))
hmodel_1 = bmodel_1.output
hmodel_1 = Flatten()(hmodel_1)
hmodel_1 = Dense(64, activation='relu')(hmodel_1)
hmodel_1 = Dropout(0.5)(hmodel_1)
hmodel_1 = Dense(2,activation= 'softmax')(hmodel_1)

model_1 = Model(inputs=bmodel_1.input, outputs= hmodel_1)
for layer in bmodel_1.layers:
    layer.trainable = False
```

Figure 11: Model Architecture for Experiment 1

```
checkpoint_1 = ModelCheckpoint(r"model_1.h5",
                               monitor='val_loss',save_best_only=True,verbose=3)

earlystop = EarlyStopping(monitor = 'val_loss', patience=7, verbose= 3, restore_best_weights=True)

learning_rate = ReduceLROnPlateau(monitor= 'val_loss', patience=3, verbose= 3, )

callbacks=[checkpoint_1,earlystop,learning_rate]
```

Figure 12: Keras Parameters for Model for Experiment 1

```
bmodel_2 = InceptionV3(include_top=False, weights='imagenet', input_tensor=Input(shape=(80,80,3)))
hmodel_2 = bmodel_2.output
hmodel_2 = Flatten()(hmodel_2)
hmodel_2 = Dense(64, activation='relu')(hmodel_2)
hmodel_2 = Dropout(0.5)(hmodel_2)
hmodel_2 = Dense(2,activation= 'softmax')(hmodel_2)

model_2 = Model(inputs=bmodel_2.input, outputs= hmodel_2)
for layer in bmodel_2.layers:
    layer.trainable = False
```

Figure 13: Model Architecture for Experiment 2

```
checkpoint_2 = ModelCheckpoint(r"C:\Users\Amit\Documents\Academics\NCI\Semester_3\Driver Drowsiness\models\model_2.h5",
                               monitor='val_loss',save_best_only=True,verbose=3)

earlystop = EarlyStopping(monitor = 'val_loss', patience=14, verbose= 3, restore_best_weights=True)

learning_rate = ReduceLROnPlateau(monitor= 'val_loss', patience=3, verbose= 3, )

callbacks_2=[checkpoint_2,earlystop,learning_rate]
```

Figure 14: Keras Parameters for Model for Experiment 2

6.3 Experiment 3

In order to mitigate the problem of overfitting, regularization technique of deep learning, viz, dropouts were used after the flatten and penultimate dense layers. The value of the parameter 'patience' in the 'EarlyStopping' criteria in Keras was also set to 5. This resulted in the difference of 10.5% between the categorical accuracies on train and test datasets.

```
bmodel_7 = InceptionV3(include_top=False, weights='imagenet', input_tensor=Input(shape=(80,80,3)))
hmodel_7 = bmodel_7.output
hmodel_7 = Flatten()(hmodel_7)
hmodel_7 = Dropout(0.25)(hmodel_7)
hmodel_7 = Dense(128, activation='relu')(hmodel_7)
hmodel_7 = Dropout(0.5)(hmodel_7)
hmodel_7 = Dense(2,activation= 'softmax')(hmodel_7)

model_7 = Model(inputs=bmodel_7.input, outputs= hmodel_7)
for layer in bmodel_7.layers:
    layer.trainable = False
```

Figure 15: Model Architecture for Experiment 3

6.4 Experiment 4

A model with similar architecture to that of experiment 3 was use. The only difference was the introduction of the 'Batch Normalization' just before the dropouts. However, this degraded the overall performance.

```
from tensorflow.keras.layers import BatchNormalization

bmodel_8 = InceptionV3(include_top=False, weights='imagenet', input_tensor=Input(shape=(80,80,3)))
hmodel_8 = bmodel_8.output
hmodel_8 = Flatten()(hmodel_8)
hmodel_8 = BatchNormalization()(hmodel_8)
hmodel_8 = Dropout(0.25)(hmodel_8)
hmodel_8 = Dense(128, activation='relu')(hmodel_8)
hmodel_8 = BatchNormalization()(hmodel_8)
hmodel_8 = Dropout(0.5)(hmodel_8)
hmodel_8 = Dense(2,activation= 'softmax')(hmodel_8)

model_8 = Model(inputs=bmodel_8.input, outputs= hmodel_8)
for layer in bmodel_8.layers:
    layer.trainable = False
```

Figure 16: Model Architechture for Experiment 4

6.5 Experiment 5

After a lot of experimentation, it was apparent that adding multiple dense layers with lower number of neurons and high dropouts after each dense layer were achieving lower difference between the train and test categorical accuracies. This was the motivation for this experiment.

```
bmodel_16 = InceptionV3(include_top=False, weights='imagenet', input_tensor=Input(shape=(80,80,3)))
hmodel_16 = bmodel_16.output
hmodel_16 = Flatten()(hmodel_16)
hmodel_16 = Dense(128, activation='relu')(hmodel_16)
hmodel_16 = Dropout(0.5)(hmodel_16)
hmodel_16 = Dense(64, activation='relu')(hmodel_16)
hmodel_16 = Dropout(0.5)(hmodel_16)
hmodel_16 = Dense(32, activation='relu')(hmodel_16)
hmodel_16 = Dropout(0.5)(hmodel_16)
hmodel_16 = Dense(8, activation='relu')(hmodel_16)
hmodel_16 = Dropout(0.5)(hmodel_16)
hmodel_16 = Dense(2,activation= 'softmax')(hmodel_16)

model_16 = Model(inputs=bmodel_16.input, outputs= hmodel_16)
for layer in bmodel_16.layers:
    layer.trainable = False
```

Figure 17: Model Architechture for Experiment 5

6.6 Discussion

Following is the brief summary of the results obtained in the main experiments:

Experiment Number	Categorical Accuracy on Train Dataset	Categorical Accuracy on Test Dataset	Difference	Patience Value in Early Stopping	Batch Normalization
1	94.78%	82.32%	12.46%	7	No
2	94.98%	81.92%	13.06%	14	No
3	93.88%	83.38%	10.50%	5	No
4	90.68%	76.97%	13.71%	5	Yes
5	92.50%	84.10%	8.40%	5	No

Figure 18: Evaluation Matrix of 5 Best Models

High categorical accuracies on the test datasets were obtained in experiments 1, 2 and 3. However, despite the best efforts the problem of overfitting still persists. The difference between the categorical accuracies on the train and test datasets must be below 5%.

As stated in section 6, the values of recall and precision were the exact replicas of the respective values of categorical accuracies in all the models, this leads to a suspicion that there might be an issue which has not yet been identified. Same values of recall and precision means that the value of F1 score will be the same too, that is: recall = precision = F1 Score. This appears to be bizarre and needs to be investigated.

7 Conclusion and Future Work

Stating the research question to give an immediate context:

"How effectively can deep learning be utilized to detect driver drowsiness?"

Following were the research objectives:

1. Train the models from MRL Eye Dataset.

2. Avoid Overfitting.
3. Achieve highest possible Recall and F1 scores.

In this research, a binomial classification was done with the help of customized InceptionV3 model (having pretrained weights from Imagenet) to check if the subject's eyes are open or closed. Publicly available "MRL Eye Dataset" was used to train the model. An input pipeline had to be created as the raw datset is in the zipped form containing a total of 37 folders having varying number of images inside them. Moreover, the defualt InceptionV3 model in Keras takes the input of the shape (299, 299, 3) whereas majority of the preprocessed images were of the shape (80, 80, 3). The individual pixel values in all images were also rescaled between 0 and 1.

The output of the input pipeline was fed in to the customized InceptionV3 model. The categorical accuracy of the first model on the train data was 94.78% but on the test data was 82.32%. The difference was 12.46%, thereby indicating the presence of overfitting.

Similarly, for the second model, the categorical accuracy on the train data was 94.98% but on the test data was 81.92%.

In future, model will be trained to recognize yawning. An optimization framework of hyperparameters in Keras, namely 'Keras Tuner' will be used to get a better model. Moreover, dlib library will be used instead of Haar cascade mechanism as the former has been shown to have 99.38% accuracy on the benchmark dataset: "Labeled Faces in the Wild".

References

Amangi, A., Mahajan, A., Huddar, M. and Kurumkar, P. (2022). Driver drowsiness detection system, *Journal homepage: www. ijrpr. com ISSN* **2582**: 7421.

Bano, A., Saxena, A. and Das, G. K. (2021). An efficient detection approach of driver-drowsiness using multiple convolutional haar cascade kernelized cnn (mchc-kcnn) algorithm, ICTACT JOURNAL ON DATA SCIENCE AND MACHINE LEARNING, MARCH 2021, VOLUME: 02, ISSUE: 02.

Cheerla, S., Reddy, D. P. and Raghavesh, K. S. (2022). Driver drowsiness detection using machine learning algorithms, *2022 2nd International Conference on Artificial Intelligence and Signal Processing (AISP)*, IEEE, pp. 1–6.

Choudhary, Y., Aggarwal, A. and Agarwal, A. (2022). Detecting drivers' drowsiness using haar cascade classifier, *2022 9th International Conference on Computing for Sustainable Global Development (INDIACom)*, IEEE, pp. 318–322.

Chowdhury, A. I., Niloy, A. R., Sharmin, N. et al. (2021). A deep learning based approach for real-time driver drowsiness detection, *2021 5th International Conference on Electrical Engineering and Information & Communication Technology (ICEEICT)*, IEEE, pp. 1–5.

Chowdhury, S. and Jos, J. (2022). Driver drowsiness detection using haar cascades, IRJMETS.

Dey, S., Chowdhury, S. A., Sultana, S., Hossain, M. A., Dey, M. and Das, S. K. (2019). Real time driver fatigue detection based on facial behaviour along with machine learning approaches, *2019 IEEE International Conference on Signal Processing, Information, Communication & Systems (SPICSCON)*, IEEE, pp. 135–140.

Farahnakian, F., Leoste, J. and Farahnakian, F. (2021). Driver drowsiness detection using deep convolutional neural network, *2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, IEEE, pp. 1–6.

Kamarudin, N., Jumadi, N. A., Mun, N. L., Ng, C., Ching, A. H. K., Mahmud, W., Morsin, M. and Mahmud, F. (2019). Implementation of haar cascade classifier and eye aspect ratio for driver drowsiness detection using raspberry pi, *Univ. J. Electr. Electron. Eng* **6**(5B): 67–75.

Kumar, V. P., Aravind, P., Pooja, S. N. D., Prathyush, S., AngelDeborah, S. and Chandran, K. S. (2021). Driver assistance system using raspberry pi and haar cascade classifiers, *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)*, IEEE, pp. 1729–1735.

Magán, E., Sesmero, M. P., Alonso-Weber, J. M. and Sanchis, A. (2022). Driver drowsiness detection by applying deep learning techniques to sequences of images, *Applied Sciences* **12**(3): 1145.

Nandhini, P., Kuppuswami, S., Malliga, S., Srinath, P. and Veeramanikandan, P. (2022). Driver drowsiness detection using deep learning, *2022 6th International Conference on Computing Methodologies and Communication (ICCMC)*, IEEE, pp. 1031–1036.

Nandyal, S. et al. (2022). Driver drowsiness estimation based on hybrid feature extraction and light weighted dense convolutional network, *2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE)*, IEEE, pp. 1–6.

Pachouly, S., Bhondve, N., Dalvi, A., Dhande, V. and Bhamare, N. (2020). Driver drowsiness detection using machine learning, IJRASET.

Phan, A.-C., Nguyen, N.-H.-Q., Trieu, T.-N. and Phan, T.-C. (2021). An efficient approach for detecting driver drowsiness based on deep learning, *Applied Sciences* **11**(18): 8441.

Pulluri, R. and Ranjana, P. (2022). An efficient vision based method for detecting drowsiness in real-time, *2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS)*, IEEE, pp. 74–78.

Sharma, A., Shah, K. and Verma, S. (2021). Face recognition using haar cascade and local binary pattern histogram in opencv, *2021 Sixth International Conference on Image Information Processing (ICIIP)*, Vol. 6, IEEE, pp. 298–303.

Sinha, A., Aneesh, R. and Gopal, S. K. (2021). Drowsiness detection system using deep learning, *2021 Seventh International conference on Bio Signals, Images, and Instrumentation (ICBSII)*, IEEE, pp. 1–6.

Suresh, Y., Khandelwal, R., Nikitha, M., Fayaz, M. and Soudhri, V. (2021). Driver drowsiness detection using deep learning, *2021 2nd International Conference on Smart Electronics and Communication (ICOSEC)*, IEEE, pp. 1526–1531.

Suryawanshi, Y. and Agrawal, S. (2020). Driver drowsiness detection system based on lbp and haar algorithm, *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, IEEE, pp. 778–783.