

Drowsiness Detection System

CS-485 Computer Vision and Image Processing



Session: 2021-2025

Project Supervisor
Samyan Qayyum Wahla

Submitted By
Shahzaib Rafi 2021-CS-02

Contents

1	Introduction	2
1.1	Background	2
1.2	Project Objectives	2
2	Literature Review	3
2.1	Real-Time Driver-Drowsiness Detection System Using Facial Features	3
2.1.1	Explanation:	3
2.1.2	Algorithms:	3
2.1.3	Limitations	5
2.1.4	Accuracy:	5
2.1.5	Research Papers	5
2.2	A Real-time Driving Drowsiness Detection Algorithm With Individual Differences Consideration	6
2.2.1	Explanation	6
2.2.2	Limitations	6
2.2.3	Accuracy	6
2.2.4	Research Papers	6
2.3	Deep CNN: A Machine Learning Approach for Driver Drowsiness Detection Based on Eye State	7
2.3.1	Explanation	7
2.3.2	Limitations	7
2.3.3	Accuracy	7
2.3.4	Research Papers	7
2.4	CNN Based Driver Drowsiness Detection System Using Emotion Analysis	8
2.4.1	Explanation	8
2.4.2	Limitations	9
2.4.3	Accuracy	9
2.4.4	Research Papers	9
3	Methodology	10
3.1	Explanation:	10
3.2	Working:	10
4	DataSet	11
5	Acceptance Criteria:	11
6	Evaluation	12
6.1	Evaluation Criteria:	12
6.2	Results	12
6.2.1	Without Data Augmentation	12
6.2.2	Data Augmentation	13
7	Test Cases	14
7.1	Test Case 1	14
7.1.1	Image	14

7.1.2	Classification	14
7.2	Test Case 2	14
7.2.1	Image	15
7.2.2	Classification	15
8	Reasons for Lower Accuracy	16
8.1	Small Dataset	16
8.2	GPU Availability in Google Colab	16
9	Limitations	16
10	Future work	16
11	Conclusion	16

List of Tables

1	Comparison on testing and training on base data	12
2	Table shows comparison of accuracy, precision, recall and F-1 score on base data. Data inside parenthesis shows accuracy of the model when tested on adversarial samples generated using the PGD whilst data outside parenthesis shows accuracy of the model on newly generated PGD adversarial samples after undergoing adversarial training.	13
3	Comparison on testing and training on base data after data augmentation	13
4	Table shows comparison of accuracy, precision, recall and F-1 score on base data. Data inside parenthesis shows accuracy of the model when tested on adversarial samples generated using the PGD whilst data outside parenthesis shows accuracy of the model on newly generated PGD adversarial samples after undergoing adversarial training after data augmentation.	14

1 Introduction

Project focuses on improving safety for drivers, with a primary focus on truck drivers who play a vital role in transportation. I aim to develop a simple and effective AI system capable of identifying when drivers become too tired to drive safely.

1.1 Background

The modern world heavily relies on road transportation, connecting cities, communities, and industries. This dependence underscores the importance of ensuring that our roadways are as safe as possible. However, drowsy driving poses a significant threat to this safety.

Driver fatigue is a persistent issue in the transportation sector. Long hours behind the wheel, often accompanied by insufficient rest, can lead to a state of drowsiness that impairs a driver's alertness, decision-making, and reaction times. This, in turn, increases the risk of accidents. Moreover, the consequences of drowsy driving are far-reaching, resulting in injuries, property damage, and, tragically, loss of life.

Therefore, addressing drowsy driving is a matter of universal importance, with potential benefits for all.

1.2 Project Objectives

This project aims to:

- Create an AI system capable of recognizing signs of drowsiness in drivers.
- Train this system using videos of drivers to learn and understand these signs effectively.
- Contribute to road safety by reducing accidents caused by tired drivers.

2 Literature Review

2.1 Real-Time Driver-Drowsiness Detection System Using Facial Features

2.1.1 Explanation:

"DriCare is almost a real-time system," says the research team. They faced three challenges. The first challenge is the movement of the driver's face, making it hard to keep track since the KCF algorithm has poor face-tracking accuracy in a complex environment. The second challenge is determining several key facial points, and the third is defining the driver's level of drowsiness.

Their first contribution is a new algorithm **MC-KCF (Multiple Convolutional Neural Networks(CNN)-KCF)**. This algorithm is used to track the driver's face and recognize the facial key regions based on key-point detection. This result is sent to the cloud server to estimate the driver's state. For detection accuracy in case of overcast sunlight, rain, etc., they used an illumination enhancement method.

2.1.2 Algorithms:

```
1  # Calibration of MC-KCF Algorithm
2  def calibrate(frame, t, cnt, frames):
3      res_img = None
4      while t <= 10 and cnt <= frames:
5          if t == 0:
6              # Use MTCNN algorithm to detect a human face
7              pass
8          elif not detect_with_MC_KCF(frame):
9              # Use MTCNN algorithm to detect a human face
10             pass
11         else:
12             # Use MC-KCF algorithm to detect a human face
13             pass
14
15         t += 1
16         cnt += 1
17         # Output the result res_img
18         update_scope()
19         frame = read_next_frame()
20
21     if t == 10:
22         # Use MTCNN algorithm to align the MC-KCF algorithm
23         # (use MTCNN algorithm to detect human face) in the
24         # current frame
25         pass
26
27         # Output the result res_img
28         update_scope()
29         t = 0
```

```

30     cnt += 1
31     frame = read_next_frame()

```

Here is how the DriCare Algorithm correctly detects a face. First, they get the **frame**.

- They initialize two variables (t and cnt) to zero.
- Then, a while loop is run until either t exceeds 10 or cnt exceeds the total frames.
- **Inside the loop:** They check if it is the first frame. They use **MTCNN (Multi-Task Cascaded Convolutional Neural Networks)** to detect a face; otherwise, they try with MC-KCF if, for some reason, it fails, they will use MTCNN to detect a face.
- Then they increment t and cnt and output the result image.
- Load the next frame.
- (*While the loop stops or breaks...*) They check if t == 10s. If not, then they go straight to loading the next frame, but if t == 10s, they will read the face from MC-KCF only.
- Then inc t and cnt, output *res_iimage*, and load the next image.

In short, they continuously track the face of a person in the image while **aligning the drift window** back to the person's face in case he moves his face.

```

1  #Input:
2  #frames of the video
3
4  #Output:
5  #Evaluation of the degree of driver #fatigue
6
7  #Load the frames of video
8  #Assess the states of the eye and mouth
9  #Calculate r the ratio of the frame of #eye closure in 1 minute
10 #and t a duration time of eye closure.
11 #Calculate b the frequency of blinking #and y the number of
12 #yawning in 1 minute.
13
14 if r>30% then
15   Wr=1
16 end if
17
18 if t>2s and is not yawning then
19   Wt=1
20 end if
21
22 if b>25 or b<5 then
23   Wb=1
24 end if
25
26 if y>=2 then
27   Wy=1
28 end if
29

```

```

30 Calculate T the total value of these weight. (T=Wr+Wt+Wb+Wy )
31
32 if T>=2 then
33 The driver is drowsy
34 else
35 The driver is awake
36 end if

```

Next, they check if the driver is drowsy. For this, the above algorithm's output will be the input. It simply checks how much the driver's eyes are open, how frequently they are blinking, and whether they are yawning. Based on these factors, it determines if the driver is drowsy or not.

2.1.3 Limitations

- Proposed System does not consider individual differences.
- A very slight drop in accuracy if background is dark or driver is wearing glasses.

2.1.4 Accuracy:

The researchers claimed to have reached **92%** accuracy.

2.1.5 Research Papers

W. Deng and R. Wu, "Real-Time Driver-Drowsiness Detection System Using Facial Features," in IEEE Access, vol. 7, pp. 118727-118738, 2019, doi: 10.1109/ACCESS.2019.2936663.

2.2 A Real-time Driving Drowsiness Detection Algorithm With Individual Differences Consideration

2.2.1 Explanation

The paper begins by stating that many drowsiness detection approaches are universal, meaning they don't consider individual differences in drivers' faces. As a result, they have many exceptions in which their systems don't work.

To address this issue, they introduced a new parameter for assessing whether a driver's eyes are open or closed. This parameter is called **EAR (Eyes Aspect Ratio)**. They state that EAR is found to be more stable compared to traditional methods, thanks to the Cascaded Pose Regression algorithm. Furthermore, there is a strong correlation between EAR and the size of a driver's eyes.

$$\text{EAR} = \frac{\|P_2 - P_6\| + \|P_3 - P_5\|}{2\|P_1 - P_4\|}$$

where $P_i, i = 1, 2, \dots, 6$ is the coordinate of eyes landmarks.

The paper introduces a new algorithm called **DCCNN (Deep Cascaded Convolutional Neural Network)** for detecting a driver's face in live video. DCCNN eliminates the need for manual feature extraction, a common step in traditional face detection algorithms. Their system has two modules:

1. **Offline Training:** This is the initialization process. The driver is asked to open their eyes for some time in front of a simulator and, similarly, to keep their eyes closed in front of the simulator for some time. For each instance, an EAR is calculated and labeled as EAR_1 and EAR_2 , respectively. They are both taken as positive and negative samples on which a **unique SVM (Support Vector Machine) classifier** is trained for each driver.
2. **Online Monitoring:** In this step, the driver is driving while their live video is provided to **DCCNN**, which then uses the previously trained SVM to determine if the eyes are closed or not. It then notifies the driver if the eyes are closed for too long or if there's an improper head position (which can be implied if DCCNN can't detect the face).

2.2.2 Limitations

- Proposed System first requires learning of Driver's facial feature.
- Algorithm of Proposed System is slower in comparison to other by a percentage of around 20%+

2.2.3 Accuracy

The researchers claim **98.8%** accuracy.

2.2.4 Research Papers

Feng You, X. Li, Y. Gong, H. Wang and H. Li, "A Real-time Driving Drowsiness Detection Algorithm With Individual Differences Consideration," in IEEE Access, vol. 7, pp. 179396-179408, 2019, doi: 10.1109/ACCESS.2019.2958667.

2.3 Deep CNN: A Machine Learning Approach for Driver Drowsiness Detection Based on Eye State

2.3.1 Explanation

In this work, a new framework is proposed using deep learning to detect driver drowsiness based on Eye state while driving the vehicle. To detect the face and extract the eye region from the face images, Viola-Jones face detection algorithm is used in this work. Other methods try to determine drowsiness based on steering and brake pattern but that depend on road and traffic conditions.

Once the face is detected, Viola-jones eye detection algorithm is used to extract the eye region from the face images and given as input to CNN. In this work, Viola-Jones object detection algorithm with **Haar cascade classifier** was used and implemented using OPEN CV with python. The CNN produces feature vectors when they receive image of eyes.

In a simple CNN, there are four layers which create feature maps, shorten them, remove negative values and produce class scores. But the one deep CNN proposed in the paper there are five layers:

- In each first four layer labeled Conv2d1 to 4, these operations are performed : **batch normalization, ReLU, MaxPooling, and dropout**. Each layer is output is input of next layer. Their details are :
 1. It takes 128x128 input images, applies 84 filters of size 3x3.
 2. It applies 128 filters of size 5x5.
 3. It applies 256 filters of size 5x5.
 4. It applies 512 filters of size 5x5.
- Fifth layer requires 8,388,864 parameters

In the end, Soft max layer in CNN classify the images in to sleepy or non-sleepy images.

2.3.2 Limitations

- The Old Algorithm (Viola-jones face detection) is prone to generate false detection of faces.
- Model is resource-extensive and computationally demanding.

2.3.3 Accuracy

The researchers claim **96.42%** accuracy.

2.3.4 Research Papers

Venkata Rami Reddy Chirra, Srinivasulu Reddy Uyyala, Venkata Krishna Kishore Kolli "Deep CNN: A Machine Learning Approach for Driver Drowsiness Detection Based on Eye State" in Revue d'Intelligence Artificielle Vol. 33, No. 6, December, 2019, pp. 461-466

2.4 CNN Based Driver Drowsiness Detection System Using Emotion Analysis

2.4.1 Explanation

This paper attempts to solve the drowsiness detection efficiently by detection of the driver's fatigue and identification of emotion of driver.

1. **Detection of the driver's fatigue:** It has three phases:

- **Data Gathering:** Here they collect data about driver behaviour and his driving to teach their model.
- **Pre-Processing Phase:** In this phase they label each frame as whether normal, aggressive etc. They collect image into time domain of 10 second and check for signs of distraction and behaviour.

```
1      Input: Xi= Sum (Xni);i=1,2,3...n Xi=(X1 i+X2 i+X3 i+...+Xn i)
2      Output: Window size:150
3      Sliding window:sw=1
4      Recurrence threshold RT
5      Recurrence Plot function:using equation 1
6      for samples Xi self dataset
7      do
8          for representations r belongs to rep(Xi)
9          do
10             Image150X100=R(a,b)x
11          end
12          Image=concatenate(ImageX)
13      end
```

- **Deep Learning Phase:** In this step, output of previous phase is fed to CNN which extracts important features from image. This proposed model uses two levels of convolution filters.

```
1      Input:
2          Training Image (Xtrain,Ytrain)
3          Testing Image(Xtest,Ytest)
4      ConvolutionProcess:
5          Initialize the Parameters
6          Batch size:50
7          Epochs:25
8          Droprate:0.25
9          Poolsize:(2,2)
10         for convolutionlayer(n,2n)=(2,4)
11             for filtersize(2,3,4,5)
12                 do
13                     filter size(10,20)
14                     model.add(conv2D(2*filtercount,
15                                     (filtersize,filtersize)))
16                     model.add(maximumpooling2D,poolsize)
17                 end
18             model.add(Dense(quantity_levels,activate=softmax))
```

```

19         losses.catoegorial-crossentropy,optimizing=optimizer()
20         model.evaluate()
21     end
22 end

```

2. **Emotion Detection System:** Here the preprocessed images are compared with trained images to determine the driver's emotions, which can be classified into multiple levels like normal, anger, disgust, fear, happiness, and sadness. Emotion is identified through **Driver Emotion Detection Classifier (DEDC)**. This CNN has three layers: the first two layers classify the 50x50 pixel image, and the final layer magnifies the image to accurately identify the driver's emotion. Based on driver's emotion, a song is played.

2.4.2 Limitations

- Proposed System first requires learning of Driver's facial features and Behaviour.
- Detecting correct emotion from face isn't always possible.

2.4.3 Accuracy

Researchers claimed **93%** accuracy.

2.4.4 Research Papers

H.Varun Chand and J.Karthikeyan "*CNN Based Driver Drowsiness Detection System Using Emotion Analysis*" IASC, 2022, vol.31, no.2 16 June 2021

3 Methodology

3.1 Explanation:

This project work on a pre-deployment training approach. Initially, the model is trained and tested using a non-augmented dataset. Subsequently, the augmented data undergoes modifications through adversarial attacks such as **PGD, FGSM, and DeepFool**, and the model is retrained and retested. Following this, new augmented data is generated, and the model undergoes another round of testing and training, encompassing both the original and adversarial contexts.

The reason for adversarial attacks is to study the **robustness of the networks under adversarial conditions**. The robustness acts as a proxy to real life conditions which might exhibit poor lighting, out-of-focus camera zoom, overexposed/underexposed shutter, height or width shift in the frame etc.

3.2 Working:

Dataset Setup:

- I used a dataset consisting of about 4000 images.
- This dataset was divided into training and cross-testing sets with a 80 20 split.

Training Process:

- I trained the models on the training set, which includes images of open and closed eyes (2 classes).
- Models were built using neural network architectures, such as Convolutional Neural Networks (CNNs).
- The training process involved adjusting the model's parameters to minimize a binary cross-entropy loss function.

Cross-Testing:

- Cross-testing was performed on a subset of images from the original dataset.
- This process helps assess how well the model generalizes to new, but related, data.

Evaluation on New Data:

- For final evaluation, new images (353) that the model had never seen before.
- This provides a more rigorous assessment of the model's performance on unseen data.

Adversarial Attacks:

- The models were subjected to adversarial attacks, such as PGD, FGSM, and DeepFool.
- Adversarial attacks involve introducing perturbations to input data to deceive the model.

Adversarial Training:

- Models were trained on the original dataset plus adversarially generated samples.
- This process helps the model become more robust to adversarial attacks.

Data Augmentation:

- Data augmentation techniques, like rotation and flipping, were applied during training.
- Augmentation is used to artificially increase the diversity of the training dataset, potentially improving model generalization.

4 DataSet

Dataset used in the training is split into two, where one is for training and the other for testing. Each split is further divided into two classifications: **Open and Closed**.

5 Acceptance Criteria:

The project will be considered successful when:

1. The AI system can consistently and accurately detect signs of driver drowsiness.
2. The system operates in real-time with minimal latency, ensuring practical usability.
3. The AI system has been trained on a sufficiently diverse dataset to handle various driver behaviors and conditions.

6 Evaluation

6.1 Evaluation Criteria:

To evaluate the success of the project, the following criteria will be used:

1. **Accuracy:** The AI system should achieve a high accuracy rate in drowsiness detection.
2. **Latency:** The system's processing time should ensure real-time functionality.

6.2 Results

6.2.1 Without Data Augmentation

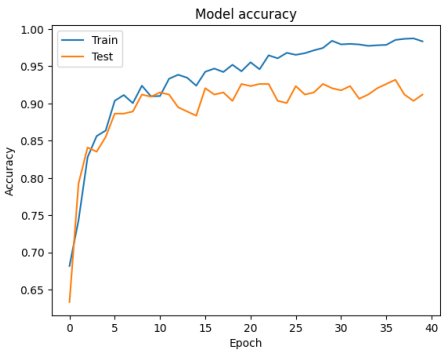
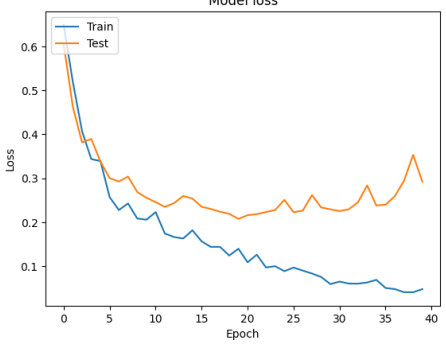
	Graph
Eye model accuracy for training and testing dataset versus epoch count	 <p>The graph shows the model's accuracy over 40 epochs. The training accuracy (blue line) starts at approximately 0.65 and rises to about 0.98. The testing accuracy (orange line) starts at approximately 0.65 and rises to about 0.92.</p>
Eye model binary cross entropy loss for training and testing datasets versus epoch count	 <p>The graph shows the model's loss over 40 epochs. The training loss (blue line) starts at approximately 0.6 and decreases to about 0.05. The testing loss (orange line) starts at approximately 0.6 and decreases to about 0.25.</p>

Table 1: Comparison on testing and training on base data

After executing the code on the provided dataset, I computed various factor values.

Config	Accuracy	Precision	Recall	F-1 Score
Base	93	92	93	94
PGD	65.98 (65.57)	78	66	54
FGSM	64.75 (64.75)	42	65	51

DeepFool	64.75 (64.75)	42	65	51
----------	---------------	----	----	----

Table 2: Table shows comparison of accuracy, precision, recall and F-1 score on base data. Data inside parenthesis shows accuracy of the model when tested on adversarial samples generated using the PGD whilst data outside parenthesis shows accuracy of the model on newly generated PGD adversarial samples after undergoing adversarial training.

6.2.2 Data Augmentation

	Graph
Eye model accuracy for training and testing dataset versus epoch count	<p>Model accuracy</p>
Eye model binary cross entropy loss for training and testing datasets versus epoch count when training data is augmented	<p>Model loss</p>

Table 3: Comparison on testing and training on base data after data augmentation

After executing the code on the provided dataset, I computed various factor values.

Config	Accuracy	Precision	Recall	F-1 Score
Base	84	88	84	84
PGD	72.13 (72.95)	73	72	68
FGSM	64.75 (64.75)	42	65	51
DeepFool	64.75 (64.75)	42	65	51

Table 4: Table shows comparison of accuracy, precision, recall and F-1 score on base data. Data inside parenthesis shows accuracy of the model when tested on adversarial samples generated using the PGD whilst data outside parenthesis shows accuracy of the model on newly generated PGD adversarial samples after undergoing adversarial training after data augmentation.

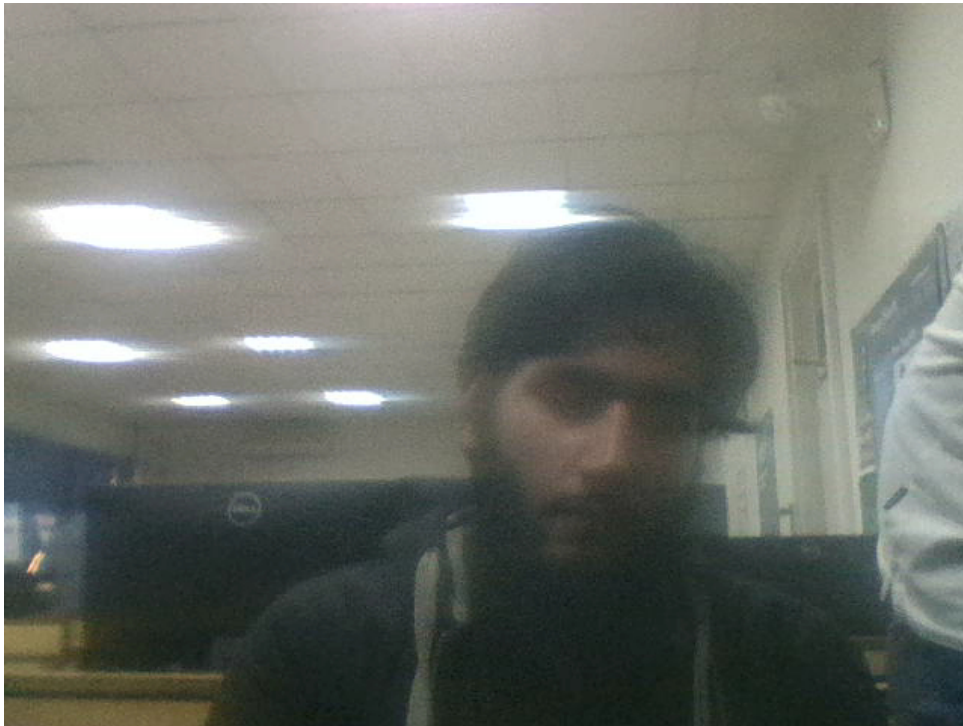
7 Test Cases

Here i provided two test cases, one where person is actually sleeping and the other where he is acting. Since the system was trained against adversarial attacks too to combat issues raised due to ethnicity, it is able to tell apart.

7.1 Test Case 1

Here person is actually sleeping.

7.1.1 Image



7.1.2 Classification

```
[INFO] loading eye model  
[INFO] Probability of eye being open: [[0.07918657]]  
[INFO] Predicted label: closed
```

7.2 Test Case 2

Here person is acting rather than actually sleeping.

7.2.1 Image



7.2.2 Classification

```
[INFO] loading eye model  
[INFO] Probability of eye being open: [[0.9325189]]  
[INFO] Predicted label: open
```

8 Reasons for Lower Accuracy

8.1 Small Dataset

First reason is the small size of dataset. To achieve higher accuracy, more diverse and larger dataset will be required.

8.2 GPU Availability in Google Colab

While running the code in Google Colab, I encountered the absence of the `nvidia-smi` command. The differences in GPU access between my local setup and Colab could potentially contribute to variations in the calculated factors.

9 Limitations

- Proposed System does not consider individual differences.
- Only grayscale adversarial images are generated.

10 Future work

Individual differences matter alot especially if different ethnicity so the model should be modified to also be trained on the individual too.

11 Conclusion

This project aims to make the roads safer and protect drivers' well-being. Our main goal is to create a smart AI system that can spot tired drivers to prevent accidents and injuries. We want to tackle the widespread problem of drowsy driving and make roads safer for everyone. We hope for a future where all vehicles have AI safety features, making our cities healthier and more secure.