Drowsiness Detection System

CS-371 Artificial Intelligence

Session: 2021-2025

Project Supervisor

Samyan Qayyum Wahla

Group Members

Shahzaib Rafi 2021-CS-02

Department of Computer Science
University of Engineering and Technology, Lahore
Pakistan

‘

Contents

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 world today depends a lot on roads to connect cities, towns, and businesses. It's really important to make sure these roads are safe. But, there's a big problem: tired driving.

People driving when they're really tired is a big issue, especially for those who drive for a living. Spending too many hours driving without enough rest can make drivers sleepy. When they're sleepy, they can't pay as much attention, make good choices, or react quickly. This makes accidents more likely. And when accidents happen because of tired driving, they can cause injuries, damage to stuff, and even death.

So, dealing with tired driving is something that matters to everyone because it can make things safer for all of us.

1.2 Project Objectives

This project aims to:

- Create an AI system capable of recognizing signs of drowsiness in drivers.
- 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:

```
# Calibration of MC-KCF Algorithm
def calibrate(frame, t, cnt, frams):
    res_img = None
    while t <= 10 and cnt <= frams:
        if t == 0:
            # Use MTCNN algorithm to detect a human face
            pass
        elif not detect_with_MC_KCF(frame):
            # Use MTCNN algorithm to detect a human face
            pass
        else:
            # Use MC-KCF algorithm to detect a human face
            pass

        t += 1
        cnt += 1
        # Output the result res_img
        update_scope()
        frame = read_next_frame()

    if t == 10:
        # Use MTCNN algorithm to align the MC-KCF algorithm
        # (use MTCNN algorithm to detect human face) in the
        # current frame
        pass

        # Output the result res_img
        update_scope()
        t = 0
```

```
        cnt += 1
        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_image, 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.

```
#Input:
#frames of the video

#Output:
#Evaluation of the degree of driver #fatigue

#Load the frames of video
#Assess the states of the eye and mouth
#Calculate r the ratio of the frame of #eye closure in 1 minute
#and t a duration time of eye closure.
#Calculate b the frequency of blinking #and y the number of
#yawning in 1 minute.

if r>30% then
Wr=1
end if

if t>2s and is not yawning then
Wt=1
end if

if b>25 or b<5 then
Wb=1
end if

if y>=2 then
Wy=1
end if

```

```
Calculate T the total value of these weight. (T=Wr+Wt+Wb+Wy )

if T>=2 then
The driver is drowsy
else
The driver is awake
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,. . . ,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.

3 Methodology

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, FGSW, 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, containing 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 lightning, out-of-focus camera zoom, overexposed/underexposed shutter, height or width shift in the frame etc.

4 DataSet

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

Dataset is available here.

5 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 Conclusion

This project aims to make the roads safer and protect drivers' well-being. The main goal is to create a smart AI system that can spot tired drivers to prevent accidents and injuries.