Facial Feature Detection and Driver Drowsiness Monitoring

ECE 5821 Final Project Report

From
Ankita Negi (37961411)
Link to github repo:
https://github.com/ankitan07/ECE_5821_Driver_drowsiness_detection/tree/master

1. Introduction:
Road accidents, a dark consequence of industrialization's progress, stand as a poignant reminder of our advancement's bittersweet reality. The astounding strides we've made in manufacturing, technology, and transportation have been marred by a disturbing shadow: the toll of road accidents on human lives. The World Health Organization (WHO) and the Centers for Disease Control and Prevention (CDC) have ranked road accidents as the eighth leading cause of global mortality, only surpassed by various diseases. Every year, these accidents claim millions of lives, while leaving countless others to grapple with debilitating injuries and lasting disabilities.

In a world where motor vehicles number a staggering 1.4 billion (as of 2020, according to WHO), even the slightest increase in road accidents exacts an enormous human cost. In the United States alone, a projected 243.4 million licensed drivers (2023) navigate the roadways alongside over 280 million registered vehicles (2021). Such figures underscore the undeniable integration of vehicles into the fabric of human existence, compelling us to prioritize measures that mitigate the tragic toll of road accidents.

2.Project Description:
This project aims to develop a facial feature detection and drowsiness detection for driver monitoring. The goal is to accurately detect facial landmarks and understand their attention and distraction while driving.
The project outcome are in two-fold. Firstly, the implementation of accurate facial feature detection that will contribute to a more robust understanding of driver behavior. Secondly, by monitoring attention and distraction patterns, the system will be equipped to issue real-time alerts to drowsy drivers

3. Related Works and state-of-the-art technologies:
There has been much research done on detecting drivers' fatigue. Much research on fatigue has been done that focuses on some of the known factors like EEG and ECG signals, steering motions and divers face monitoring. In this project we would be focusing on the driver's facial feature as the majority of the symptoms of fatigue.
Following are the major activities that has been explored in the till to date research to detect the fatigue or attention :

1. yawning
2. Electrocardiography (ECG),Electroencephalography (EEG), and Electrooculogram (EOG) to access the driver's conditions
3. head orientation
4. Steering movement and braking movement
5. Eye: Continuous eye closure, percentage of eye closure(PERCLOS) in a certain period of time, distance between the Eyelids,speed of eye blink, direction of gaze, degrees of eye openness; variability in size
6. 3-D geometric reasoning to detect EOR

In this project we are interested to only explore the technologies that helps in facial feature detection to detect drivers drowsiness

3.1 Algorithms and Techniques:
Facial feature tracking systems have been in development in many OEMs for use in hands free driving. Examples include Ford's Bluecruise and GM's Supercruise which allow you to not have to put your hand on the wheel as long as this system detects that you are paying attention to the road. According to professionals at Tesla, a facial feature tracking and feature detection system is also being rolled out to users in the Beta for Full Self Driving in anticipation of hand free availability in the future.

Most of the projects till data have explored the idea of facial landmark detection and the experimented with the following ratio to be considered as drowsy:

1	Optimal level of eye openness or Eye aspect ratio(EAR)	$EAR(i) = \frac{\|p_{38} - p_{42}\| + \|p_{39} - p_{41}\|}{2\|p_{37} - p_{40}\|}$
2	Mouth aspect ratio(MAR)	$MAR(i) = \frac{\|p_{63} - p_{67}\|}{\|p_{61} - p_{65}\|}$
3	Percentage of Eye Closure over Time(PERCLOS)	$PERCLOS = \frac{count\ of\ frames\ when\ the\ eyes\ are\ closed}{total\ count\ of\ frames\ up\ until\ that\ moment} \times 100\%$
4	Level of Eyebrow(LEB)	$LEB(i) = \frac{\|p_{21} - p_{40}\| + \|p_{22} - p_{40}\|}{2}$

3.2 Eye detection related work:
Following are some details of research work in different areas:

1	Varying Threshold	Used the common method of identifying the facial features and explored the different ratios like EAR, MAR, LEB , PERCLOS etc. Finally explored the optimal classifier like SVm, Kn, decision tree etc for the classification of the extracted features using the above mentioned ratios.
2	Dynamically Varying Threshold	Used the common method of identifying the facial features and classification based on EAR, MAR ratio. Instead of using the static ratio they have included the dynamic concept of varying the EAR and MAR based on the time the driver has driven, decreasing the value with time. For this they have included the LBPH method to reset the EAR and MAR value to initial value when it detects a new driver.
3	LSTM encoder model	As drowsiness is a time dependent variable this research explored the LSTM model. For this they trained the LSTM encoder and decoder model on the non drowsy data and used the reconstruction loss to determine if the subject is drowsy. As the encoder for feature extraction and decoder has been trained on non Drowsy data the reconstruction loss for the drowsy clip is higher. Thus they explored the threshold value to classify drowsy and non drowsy
4	Multi-timescale drowsiness characterization	They explored the "multi-timescale context" to enhance predictions by concatenating information from different timescales. For this they extract eye images, used spatial CNN to estimate eyelids distance and temporal CNN module to process sequences of eyelids distances over 1-minute intervals, producing estimates of drowsiness probabilities at four different timescales: short-term (1s), intermediate-term (10s), medium-term (30s), and long-term (1min).

We have tried using the basic ideas of these models using the data we had. As modern research explores pretrained models, neural networks, we planned to explore the similar idea.

4 Best hybrid model analysis
Exploring the current models for feature detection we mapped down their pros and cons:

CNN for model feature detection		
SSD	• Detects and classifies objects in a single pass.	• Performance dip when object size is small.
YOLO	• Can detect multiple object in just one pass • Highest accuracy and frames/sec compared to any other models.	• Computational intensive hardware. • Works with Darknet-19 as its base network.
Masked RCNN	• Masked RCNN is more accurate compared to SSD as it uses 2 passes for object detection.	• Requires at least 2 passes to detect an object and hence slower and requires more effort.

We would be using the Yolo model based on the pros and cons of the detection and feature extraction model for this project.

5. Dataset :

We have used combination of 2 dataset:

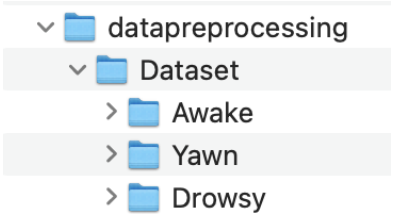
1. Publicly available driver drowsiness video dataset collected by Jaypee Institute of Information Technology, Nodia, India. 3 subject (2male , 1female)
2. Self generated dataset, 1 subject(1 female)

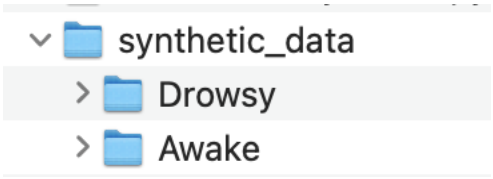
We have tried to cover different surroundings in the dataset like: with and without wearing glasses/sunglasses under a variety of simulated driving scenarios, including normal driving, yawning, slow blink rate, falling asleep, under day and night illumination conditions.
The dataset consists of 1 video and images extracted from the video. The extracted images have been shortlisted, classified and stored in 3 different folders ie - Awake(397 images), Drowsy (617 images), Yawn (169 images).
The reason for weird data ratio is because of the data availability and also because models easily identify yawn and awake and was confusing initially to detect drowsy and hence we went ahead with sign more drowsy dataset for training

5.1 Dataset creation:

We used the OpenCV library for webcam access and image manipulation to create a set of images for both "Awake" and "Drowsy" states using a webcam.
We created 25 images for each state (a total of 50 images), with a 2-second delay between captures. We tried capturing the image in a different environment with different lighting and with /without specs.
The images are saved in separate folders named "Awake" and "Drowsy" within a directory called "synthetic_data."
The following are some example of images created:

The following images clearly show the segregation of data into subfolders. We then merge the data from synthetic_data/Drowsy to dataprocessing/Dataset and synthetic_data/Awake to dataprocessing/Awake



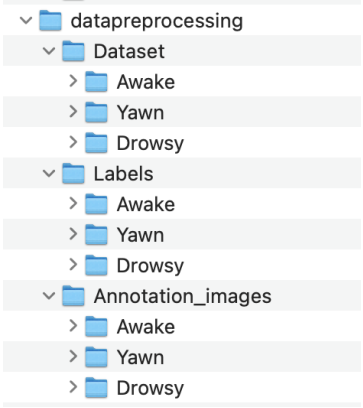


5.2 Dataset preprocessing and annotations:

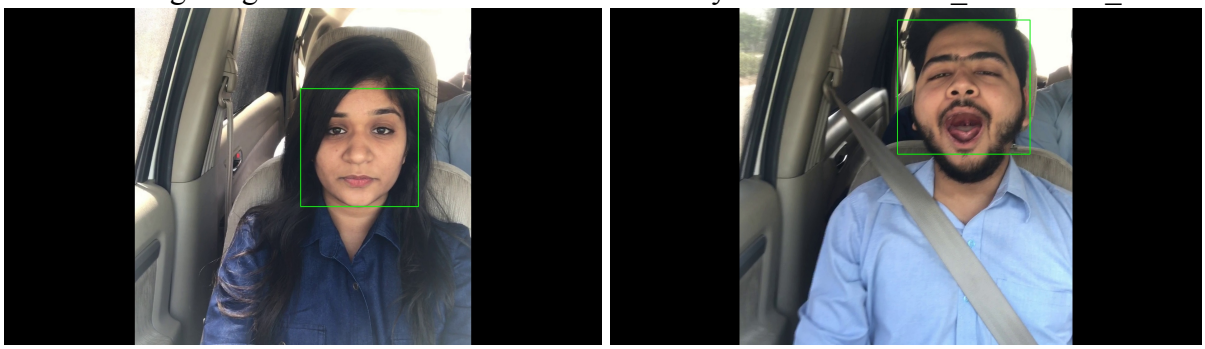
We then performed preprocessing and annotation of images in a labeled dataset. We used a pre-trained face detection model "haarcascade_frontalface_default" from OpenCV. This pretrained model detects frontal faces in images by applying the .detectMultiScale() method. This method analyzes the image and returns the coordinates of the detected faces in the form of rectangles (x, y, width, height). Further steps are as follow

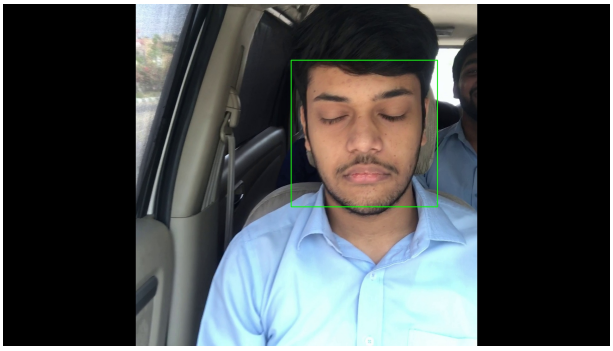
1. We filter the images that have only one annotation box and delete the rest of the images from the datapreprocessing folder.
2. Create the label and annotationa_images folder
3. These single annotated images coordinates from "haarcascade_frontalface_default" are used to get the annotation of the face in yolo format and saved in the label according to the label folder.
4. Annotated images are also saved in the annotion_images folder. This is not used further but was just saved to manually check if the model was performance in the annotation image.

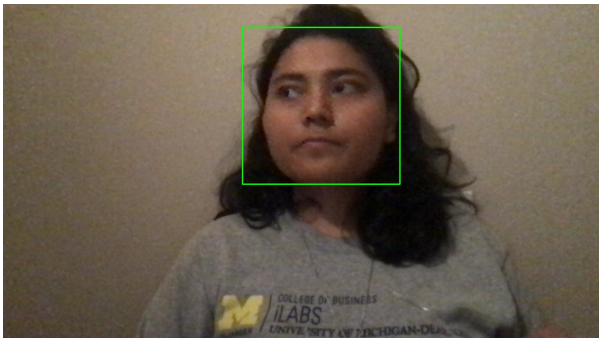
The following image shows the final segregation of data into subfolders



The following image shows the final annotation done by the "haarcascade_frontalface_default"







5.3 Data splitting

We would be using 70% of the dataset for training purposes, 30% for validation and all the data for the testing purpose.

	Training	Validate	Testing
Awake	208	89	297
Drowsy	432	185	617
Yawn	118	51	169

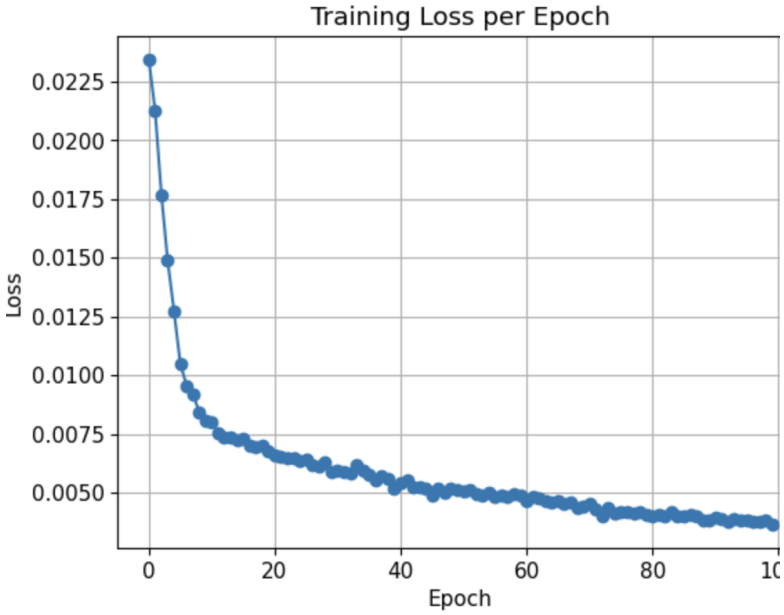
6. **Process and Results**

6.1 Training:

We would be using the train.py script with the YOLOv5 model for object detection using the Ultralytics library.

We trained a custom YOLOv5 model, extracted the trained model weights and used this new weights to test the performance of the new model on our test dataset.

We experimented with no epochs, pre-trained weights, various dataset combinations and train and validate ratios. We initially tested with 500 epochs but found that losses reached a minimum near around 100 epochs as shown in figure below. And hence planned to continue with the epoch.



6.2 Testing and selecting YOLOv5 Model:
After further testing we finally tested with different model like :
Model 1: trained data ratio 85%, validate data ratio 15%
Model 2: trained data ratio 70%, validate data ratio 30%

*For model 2 we further manually refined the data and created and included a bit diverse data compared to data used in above model

The output of these 2 models are as given in the table. Below.
We have compared the model with the best weight and last weight of each trained model.

			Output								
		Total Image	Drowsy label	Awake Label	Yawn Label	True positive	False Negative	Recall	Precision	overall recall	overall precision
best_weight model1	Drowsy	618	579	185	0	579	185	93.69%	99.66%	**96.61%**	**88.99%**
best_weight model1	Yawn	169	0	4	169	169	4	100.00%	100.00%		
best_weight model1	Awake	421	2	419	0	419	2	99.52%	68.91%		
last_weight model1	Drowsy	618	575	204	0	575	204	93.04%	99.65%	**96.27%**	**88.37%**
last_weight model1	Yawn	169	0	1	169	169	1	100.00%	100.00%		
last_weight model1	Awake	421	2	419	0	419	2	99.52%	67.15%		
best_weight model2	Drowsy	617	617	0	1	617	1	100.00%	93.20%	**98.90%**	**96.37%**
best_weight model2	Yawn	169	0	0	169	169	0	100.00%	99.41%		
best_weight model2	Awake	397	45	384	0	384	45	96.73%	100.00%		
last_weight model2	Drowsy	617	617	1	2	617	3	100.00%	96.26%	**99.41%**	**97.79%**
last_weight model2	Yawn	169	0	0	169	169	0	100.00%	98.83%		
last_weight model2	Awake	397	24	390	0	390	24	98.24%	99.74%		

We found that the model 2 with the last weight performs better than any other model and hence we would be using this model for further training.

6.2 Implementing audio alarm
We used the above trained YOLOv5 model for object detection to detect closed eyes and yawns in real-time webcam video. We have included an alarm using the "espeak" command.
It uses global variables to control the alarm status and prevent overlapping alarms.
The alarm can be triggered in two ways: continuous closed eyes detection (alarm_status) or multiple yawns in a short time (alarm_status2).
Following is an attached video clip that shows the model in real time.

Reference:

1. Alberto Fernández; Rubén Usamentiaga ; Juan Luis Carús 1ORCID and Rubén Casado. Driver Distraction Using Visual-Based Sensors and Algorithms. Sensors. 2016. https://doi.org/10.3390/s16111805
2. Rizwan Ali Naqvi; Muhammad Arsalan; Ganbayar Batchuluun; Hyo Sik Yoon; Kang Ryoung Park. Deep Learning-Based Gaze Detection System for Automobile Drivers Using a NIR Camera Sensor. Sensors. 2018. https://www.mdpi.com/1424-8220/18/2/456#B22-sensors-18-0045
3. Isha Gupta; Novesh Garg; Apoorva Aggarwal; Nitin Nepalia. Real-Time Driver's Drowsiness Monitoring Based on Dynamically Varying Threshold. 2018 Eleventh International Conference on Contemporary Computing (IC3). 2018. https://www.researchgate.net/publication/328911666_Real-Time_Driver%27s_Drowsiness_Monitoring_Based_on_Dynamically_Varying_Threshold
4. Quentin Massoz; Jacques G. Verly; Marc Van Droogenbroeck. Multi-Timescale Drowsiness Characterization Based on a Video of a Driver’s Face. Sensors. 2018. https://www.mdpi.com/1424-8220/18/9/2801
5. Gülin Tüfekci; Alper Kayabaşı; Erdem Akagündüz; İlkay Ulusoy. Detecting Driver Drowsiness as an Anomaly Using LSTM Autoencoders. First Online. 2023. https://link.springer.com/chapter/10.1007/978-3-031-25075-0_37