**Palestine Technical University - Khadoorie**

**Faculty of Engineering and Technology**

**Department of Computer Systems Engineering**



**"Introduction to Graduation Project" Thesis**



**DriveSafe: AI-Based Driver Drowsiness Detection System**

**Prepared by:**

Aseel Jayousi

Hanan Abuzainab

**Supervised by:**

Anas Melhm, Ph.D.

Tulkarm – Palestine

June 2025

**Table of contents**

[**LIST OF FIGURES** 3](#_Toc199766079)

[**LIST OF TABLES** 3](#_Toc199766080)

[ABSTRACT 4](#_Toc199766081)

[ملخص 5](#_Toc199766082)

[CHAPTER 1 – INTRODUCTION 6](#_Toc199766083)

[1.1 Motivation 7](#_Toc199766084)

[1.2 Problems 8](#_Toc199766085)

[1.3 Objectives 8](#_Toc199766086)

[CHAPTER 2 – LITERATURE REVIEW 9](#_Toc199766087)

[2.1 Traditional Drowsiness Detection Methods 10](#_Toc199766088)

[2.1.1 Physiological Signal-Based Approaches (EEG, EOG, ECG) 10](#_Toc199766089)

[2.1.2 Vehicle Behavior-Based Approaches 11](#_Toc199766090)

[2.2 Vision-Based Drowsiness Detection (Without AI) 11](#_Toc199766091)

[2.3 Deep Learning Techniques for Detecting Drowsiness 12](#_Toc199766092)

[2.3.1 Single Model Approaches 12](#_Toc199766093)

[2.3.2 Hybrid Model Approaches 12](#_Toc199766094)

[2.4 Comparison between Single Model and Hybrid Model 13](#_Toc199766095)

[2.5 Conclusion 13](#_Toc199766096)

[CHAPTER 3 - PROPOSED SOLUTION 14](#_Toc199766097)

[3.1 Proposed Model 15](#_Toc199766098)

[3.2 Proposed Scenario 16](#_Toc199766099)

[3.3 Diagrams 17](#_Toc199766100)

[3.3.1 Activity Diagram 17](#_Toc199766101)

[3.3.1 Sequence Diagram 18](#_Toc199766102)

[3.4 Project Timeline: 19](#_Toc199766103)

[Reference list 20](#_Toc199766104)

**LIST OF FIGURES**

[Figure 1:Driving Drowsiness Monitoring using EGG signal 10](#_Toc199766174)

[Figure 2:Eye landmark points used for EAR calculation 11](#_Toc199766175)

[Figure 3:llustrates the drowsiness detection pipeline using a hybrid CNN-LSTM model. 16](file:///C:\Users\ToP%20NeT\Desktop\monday%20(1).docx#_Toc199766176)

[Figure 4:Activity diagram 17](file:///C:\Users\ToP%20NeT\Desktop\monday%20(1).docx#_Toc199766177)

[Figure 5:Sequence diagram 18](#_Toc199766178)

[Figure 6:Project life cycle 19](#_Toc199766179)

**LIST OF TABLES**

[Table 1:The difference between Single Model and Hybrid Model 13](#_Toc199766188)

[Table 2:Project Timeline 19](#_Toc199766189)

# ABSTRACT

Driver drowsiness is a major cause of road accidents, especially during long or late-night trips. This project introduces a real-time drowsiness detection system that monitors the driver’s full facial features using a front-facing camera. It detects signs of fatigue like eye closure and head movement, then alerts the driver before it becomes dangerous. While similar systems are usually only available in modern high-end vehicles car, Through intelligent monitoring, the solution contributes to safer roads and more attentive driving.

# ملخص

تُعدّ حالة النعاس أثناء القيادة من الأسباب الرئيسية لحوادث الطرق، خاصةً خلال الرحلات الطويلة أو أثناء القيادة ليلاً. يقدّم هذا المشروع نظامًا للكشف عن النعاس يعمل في الوقت الحقيقي، حيث يقوم بمراقبة ملامح وجه السائق بالكامل باستخدام كاميرا أمامية. يكتشف النظام علامات التعب مثل إغلاق العينين وحركة الرأس، ثم يُصدر تنبيهًا للسائق قبل أن تتطور الحالة إلى وضع خطير. وعلى الرغم من أن أنظمة مشابهة تتوفر عادةً فقط في السيارات الحديثة والفاخرة، فإن هذا النظام يهدف من خلال المراقبة الذكية إلى المساهمة في تعزيز السلامة على الطرق ودعم التركيز والانتباه أثناء القيادة.

## 

# CHAPTER 1 – INTRODUCTION

This chapter introduces the project and highlights its main idea.

Traffic accidents are a major problem that causes material and moral losses and is one of the main causes of injury, disability, and death around the world [1][2] with the increasing number of vehicles on the roads. There are many causes of accidents, including excessive speed and the use of mobile phones while driving, among others [1] especially in low-income countries, the causes of accidents are increasing [3]. One of the serious and slowly developing causes is drowsiness [4], a real problem in which the driver loses concentration and is less able to notice sudden changes in the road, making incorrect decisions, and having difficulty staying on the right lane. Factors that cause drowsiness include driving long distances or during night driving, or it may occur due to the driver's lack of sleep, which causes poor concentration. There are early signs of drowsiness, such as frequent yawning, frequent blinking, and heavy eyelids, but these signs are often ignored by drivers, leading to accidents that affect not only the driver but also passengers and other road users. Therefore, there is a need for systems that detect driver drowsiness in its early stages to reduce traffic accidents and their significant effects. Many cars still don't have drowsiness detection systems, like older or low-cost cars, leaving a segment of drivers unprotected from drowsiness and its consequences. Therefore, it has become necessary to find a smart, practical, and low-cost solution that suits all categories to monitor drivers and keep them alert. This project aims to develop a system to detect driver drowsiness, monitor their behavior in real time, and detect potential signs of drowsiness. If signs appear, the system alerts the driver. This system can reduce traffic accidents resulting from drowsiness, save lives, and make driving safer.

## 1.1 Motivation

The idea of a driver drowsiness detection system is motivated by the following key points:

1. **Providing safety for all drivers**: Every driver deserves to reach their destination safely and to use drowsiness detection systems that are easy to use, affordable, and effective. Drowsy driving is a major risk, especially in developing countries and low-cost vehicles that lack such systems.
2. **Save lives**: By preventing accidents that could be avoided by drowsiness detection, we aim to reduce the resulting harm and protect the lives of drivers, passengers, and other road users.
3. **Lack of driver awareness**: Drowsiness is a danger while driving. Many drivers ignore drowsiness and continue driving without considering it a risk, leading to accidents.

### 

## 1.2 Problems

The following problems highlight the need for the proposed drowsiness detection system:

1. **Drowsiness increases the risk of accidents**: When drivers are tired, their focus and reaction time go down—especially during long trips or late at night. Some drivers even experience “microsleep,” where they briefly fall asleep without realizing it, which can lead to serious accidents.
2. **Such systems are not widely available**: Most cars don’t have tools that can detect signs of drowsiness. And the systems that do exist are usually expensive and found only in luxury vehicles, not in regular or public transport cars [5].
3. **Drivers often don’t realize how tired they are**: Many people think they’re fine to drive, even when they’re actually very tired. This makes it hard to judge their own alertness, which increases the risk without them noticing.

## 1.3 Objectives

The focus of this project is to design and implement an intelligent system that monitors driver alertness in real time. The main objectives are to:

1. **Enhance road safety** by providing a real-time alert system that helps prevent accidents caused by driver drowsiness.
2. **Design a cost-effective and adaptable solution** that can be installed in various types of vehicles, including personal cars, buses, and trucks, making it accessible to a wide range of users.
3. **Raise awareness about drowsy driving** by using the system to highlight the risks of driver fatigue and promote the use of intelligent safety technologies.
4. **Present the system as a proof of concept** to demonstrate how AI can contribute to road safety and inspire future development beyond the student level.

# CHAPTER 2 – LITERATURE REVIEW

This chapter presents a review of existing studies and related work relevant to the project.

## 2.1 Traditional Drowsiness Detection Methods

### 2.1.1 Physiological Signal-Based Approaches (EEG, EOG, ECG)

Some of the earliest methods for detecting drowsiness focused on physiological signals, in other words, measuring how the body responds internally when someone starts to feel tired. These approaches rely on specialized sensors that monitor brain activity, eye movement, and heart rate to detect signs of fatigue.

* **EEG (Electroencephalogram):** This measures electrical activity in the brain. Studies have shown that as a person gets drowsy, high-frequency brain waves like beta waves start to decrease, while slower waves such as theta and delta increase. These shifts are considered early indicators of sleepiness.
* **EOG (Electrooculogram):** This tracks how the eyes move and blink. When someone is tired, their blinks slow down, and the eyes tend to remain closed for longer periods — both of which are strong signs of drowsiness.
* **ECG (Electrocardiogram):** This monitors heart rate and rhythm. A decrease in heart rate or irregular heartbeat patterns can also point to the onset of fatigue.

One study [6] used EEG signals to estimate PERCLOS — the percentage of eyelid closure over time — and achieved a very low error rate (RMSE = 0.117), which shows high accuracy in predicting drowsiness.

Also in [7], the authors proposed a hybrid drowsiness detection system based on EEG and ECG signals. They tested the approach on 22 subjects in a simulated driving environment and achieved an accuracy of 80% using only two electrodes—one for EEG and one for ECG.

Even though these methods are accurate, they’re not very practical for real-world use. Most of them require physical contact with the skin or scalp through electrodes, which can be uncomfortable and not ideal for daily driving situations. That’s why their use is mostly limited.



Figure 1:Driving Drowsiness Monitoring using EGG signal

### 2.1.2 Vehicle Behavior-Based Approaches

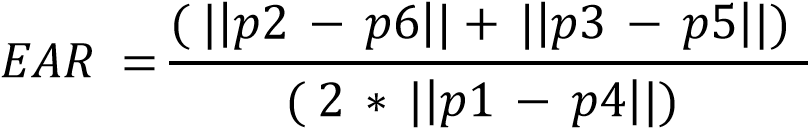
Another way to detect driver drowsiness is by observing how the driver controls the vehicle - like steering wheel movements, lane drifting, or braking patterns. These systems don’t require physical contact with the driver and often use sensors already built into modern vehicles.

In [8], the authors proposed a non-intrusive driver drowsiness detection model using steering wheel data. They applied an adaptive neuro-fuzzy feature selection method with a support vector machine classifier, achieving 98.12% accuracy in detecting whether the driver was drowsy or alert.

However, these methods usually detect drowsiness only after the driver’s behavior has already changed. They can also produce false alarms due to external factors like road curves, wind, or bumps. In addition, not all vehicles have the required sensors, and installing them can be expensive or impractical.

## 2.2 Vision-Based Drowsiness Detection (Without AI)

Before the use of artificial intelligence, vision-based drowsiness detection systems relied on handcrafted geometric features extracted from facial landmarks. A key metric used in these systems is the **Eye Aspect Ratio (EAR)**, which measures eye openness and is defined as:



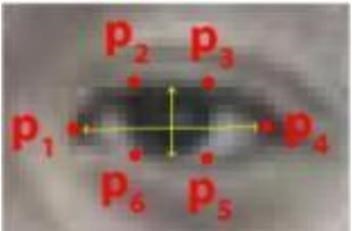


Figure 2:Eye landmark points used for EAR calculation

In [9], Soukupova and Cech proposed a real-time eye blink detection method based solely on EAR values and facial landmarks. Their system, which used Dlib for facial point detection, did not involve any machine learning techniques. Instead, it applied threshold-based classification to detect blinks in real time.

The method worked well in normal conditions but was less accurate with low light or head movement.

## 2.3 Deep Learning Techniques for Detecting Drowsiness

With the development of deep learning, there has been a shift in understanding and processing data. It solves problems in a way similar to how humans think and matches the neural networks in the human brain, including convolutional neural networks (CNNs) and recurrent neural networks (RNNs).

**CNN**: A neural network designed to extract features from datasets, useful for visual data such as images or videos. It consists of multiple convolutional layers for spatial

**RNN :** is designed for sequential data like time series and language. It uses loops to retain past information but struggles with long sequences due to the vanishing gradient problem. **LSTM (Long Short-Term Memory)** is a type of RNN developed to solve this issue and can handle long-term dependencies more effectively.

### 2.3.1 Single Model Approaches

This approach is based on a single model and is simpler than hybrid models in terms of structure and number of layers, but it is not suitable in cases that require analysis of temporal and spatial patterns together.

In [10] a system for driver drowsiness detection was proposed using computer vision and deep learning techniques. Two models were studied and compared: a custom-designed CNN model and a pre-trained VGG16 model. A dataset containing diverse images was used, and the accuracy of the CNN model was 97%, while the custom VGG16 model achieved 74%.

Also, in this study [11], a drowsiness detection system was developed using convolutional neural networks and computer vision. This system uses real-time video processing using OpenCV to extract and analyze facial features and eye ratios. The results achieved an accuracy of 97.2%. However, this system detects drowsiness automatically, meaning it processes images individually without taking into account the temporal sequence of facial movements.

### 2.3.2 Hybrid Model Approaches

This approach relies on more than one model and is used when combining temporal and spatial analysis is needed. It requires powerful hardware to operate.

In [12] a system is proposed to detect driver drowsiness from video while driving. A dataset containing video clips was used, divided into frames. The system combines two deep learning models, CNN and LSTM, and was then tested and compared with a set of other models. After comparison, the best accuracy results were achieved using the CNN + LSTM model, The precision attained reached 98.3% for training and 97.31% for testing.

## 2.4 Comparison between Single Model and Hybrid Model

Table 1:The difference between Single Model and Hybrid Model

|  |  |  |
| --- | --- | --- |
| **Aspect** | **Single Model (CNN)** | **Hybrid Model (CNN +LSTM)** |
| Framework operation | Uses CNN layers to extract features from Static image | Uses CNN for feature extraction and LSTM for temporal and sequence analysis. |
| Temporal Analysis | Cannot handle temporal analysis. | Strong temporal modeling — can detect trends over time |
| Hardware  Requirements | Suitable for low-power devices (Raspberry Pi) | Requires more computational resources (PC with GPU) |
| Key Strengths | Excellent for Classifying single images. | Excellent for sequential and time related data. |
| Processing Speed | Faster- only CNN layers | slower due to extra LSTM  processing layer |
| Accuracy | High accuracy per frame, but may miss behavioral trends | high for temporal analysis. |
| Complexity | Simple | More complex (combined CNN + RNN/LSTM) |
| Performance Metrics | - Custom CNN model: 97% accuracy [10] | CNN + LSTM model: 97.31% test accuracy, 98.3% training precision [12] |

## 2.5 Conclusion

#### After evaluating both the CNN and the CNN + LSTM models, we found that the hybrid model (CNN + LSTM) offers higher accuracy in detecting driver drowsiness. This is because it doesn’t just analyze individual frames, but also considers how the driver’s facial features change over a sequence of images. This temporal analysis helps the system recognize early signs of fatigue more effectively. For this reason, we decided to adopt the CNN + LSTM model in our system to ensure more accurate and reliable drowsiness detection.

# CHAPTER 3 - PROPOSED SOLUTION

This chapter presents the proposed system, and how it works

In this chapter, we present a proposed driver drowsiness detection system that initiates detection immediately using video inputs and determines the driver's state using a proposed hybrid model. It uses indicators to determine the system's state. If drowsiness is detected, the system activates an alert to warn the driver.

## Proposed Model

The proposed model consists of three main stages: Input, Processing, and Decision Making. These stages work together to enable real-time drowsiness detection, as illustrated in Figure 3.

**1. Input Stage**

The system uses real-time video captured from a front-facing **camera** mounted on the dashboard of the vehicle.

During the development phase, we use the **NTHU Drowsy Driver Detection (NTHU-DDD) dataset**[13], a dataset designed specifically for drowsiness detection. It contains annotated video sequences of drivers in various states such as awake, blinking, yawning, drowsy, and asleep, recorded under different lighting conditions (daylight, night, infrared). The dataset is split into training, validation, and testing subsets. Each video is segmented into individual frames before further processing.

**2. Processing Stage**

* **Face Detection:** Each video frame is passed through a face detection algorithm to locate the driver’s face. Only the facial region is extracted and forwarded for further processing.
* **Preprocessing:** The extracted face is converted to grayscale to reduce computational complexity, resized to a fixed input size (128×128 pixels), and normalized to ensure consistent input across all frames.
* **Feature Extraction(CNN):** The preprocessed face image is fed into a Convolutional Neural Network, which extracts key spatial features from each frame. These features represent facial characteristics such as eye openness, mouth shape, and head orientation.
* **Frame Sequencing:** A sequence of consecutive frames (20 frames) is grouped together to preserve the temporal flow of facial expressions. This sequence forms the input for the LSTM.
* **Temporal Analysis(LSTM):** The CNN features are sent to an LSTM network, which looks at how the face changes over time across a group of frames. It learns to recognize signs of drowsiness, like eyes staying closed for several frames, frequent blinking, or yawning that lasts over time.

**3. Decision Making stage**

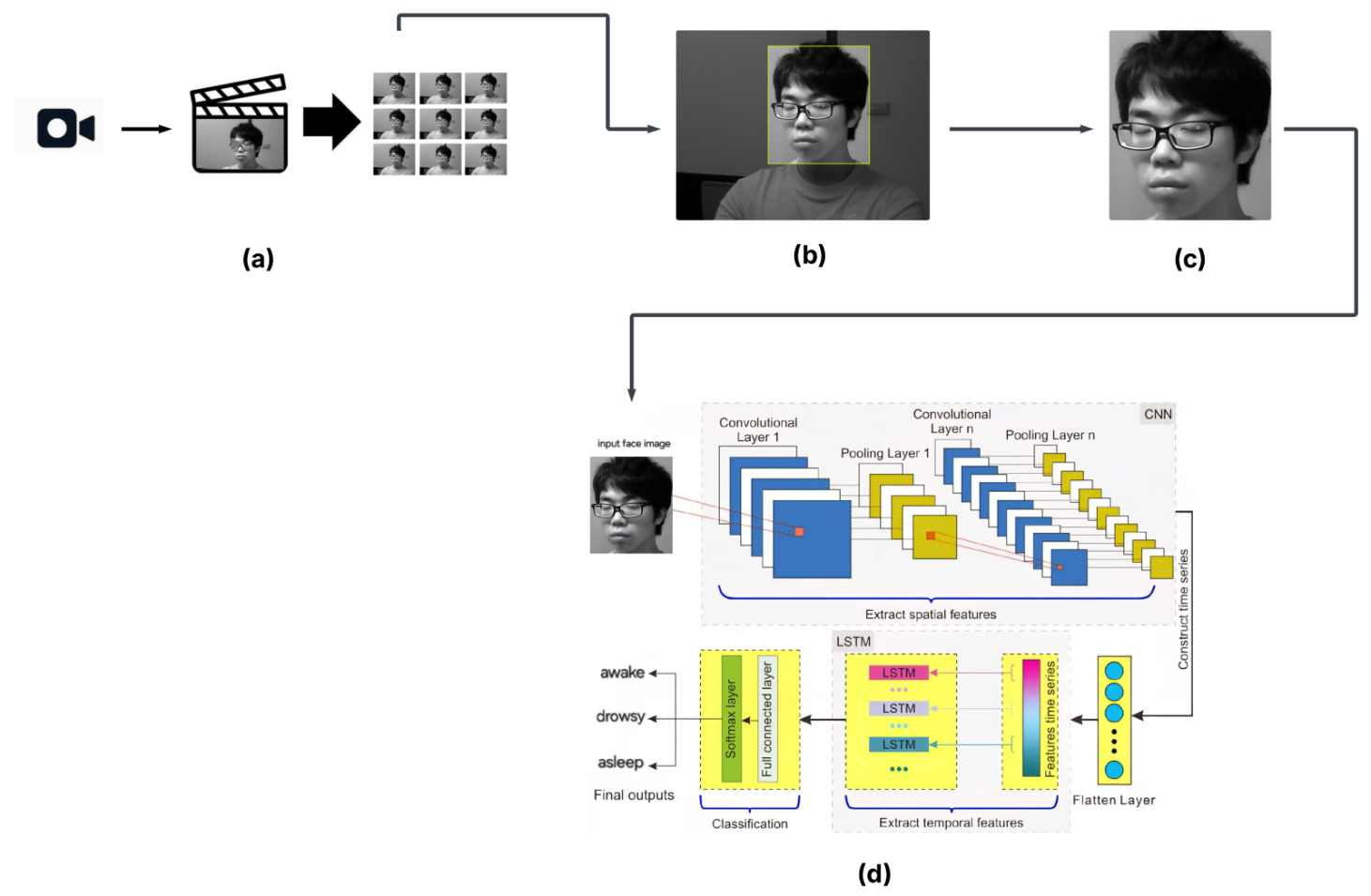
* **State Classification:**  
  After analyzing the input sequence, the model classifies the driver’s condition into one of three states:
* **Awake**
* **Drowsy**
* **Asleep**
* **System Response:**   
  Based on the detected state, the system responds in real time using a buzzer to alert the driver:
* **Alert:** No action is taken, and the system continues monitoring.
* **Drowsy:** A medium-speed buzzer is triggered to gently alert the driver.
* **Asleep:** A high-speed, urgent buzzer is activated to strongly warn the driver.
* **continuous monitoring:**   
  The system continuously updates its prediction as new frames are captured, ensuring that any change in the driver’s condition is quickly detected and handled.

Figure 3:llustrates the drowsiness detection pipeline using a hybrid CNN-LSTM model.

(a)A video stream is captured in real-time using a front-facing camera. (b) From each frame, the driver’s face is detected and localized using a face detection algorithm. (c) The detected face region is then cropped to isolate the relevant features. (d) The cropped face images are passed to a CNN to extract spatial features related to eye and facial conditions. These features are then fed into an LSTM network to analyze temporal changes over time. The final output is a classification of the driver’s state into awake, drowsy, or asleep.

## 3.2 Proposed Scenario

In a driver drowsiness detection system, the model is trained on a dataset of facial images. A night-vision camera is used to capture real-time video of the driver's face to extract, analyze, and classify frames. An LED indicator is used to alert the driver whether the system is operating or not, and an alert device (buzzer) warns the driver.

The system relies on two LED indicators, red and green, to indicate the system status to the driver and provide visual notification of the system status, helping the driver identify the system's operation or make adjustments if an error occurs.

* **Red LED**: The red LED turns on when there is a specific error or problem.
* case 1: The driver's face cannot be seen due to the camera not being directed toward them, their head tilted, or their wearing sunglasses that obstruct their vision.
* case 2: A hardware error, such as a camera connection failure.

* **Green LED**: The green LED turns on when the system is operating properly. This results in the following three cases:
* case 1: The driver's face is visible while awake.

The system continues without activating an alert or issuing any signal.

* Case 2: The driver's face is visible while drowsy.

The system activates a medium-speed buzzer to alert the driver.

* Case 3: The driver's face is visible while asleep.

The system activates a high-speed buzzer to wake the driver.

## 3.3 Diagrams

This section includes two diagrams that illustrate the internal behavior of the drowsiness detection system.

### 3.3.1 Activity Diagram

The following diagram illustrates the core workflow of the proposed drowsiness detection system.

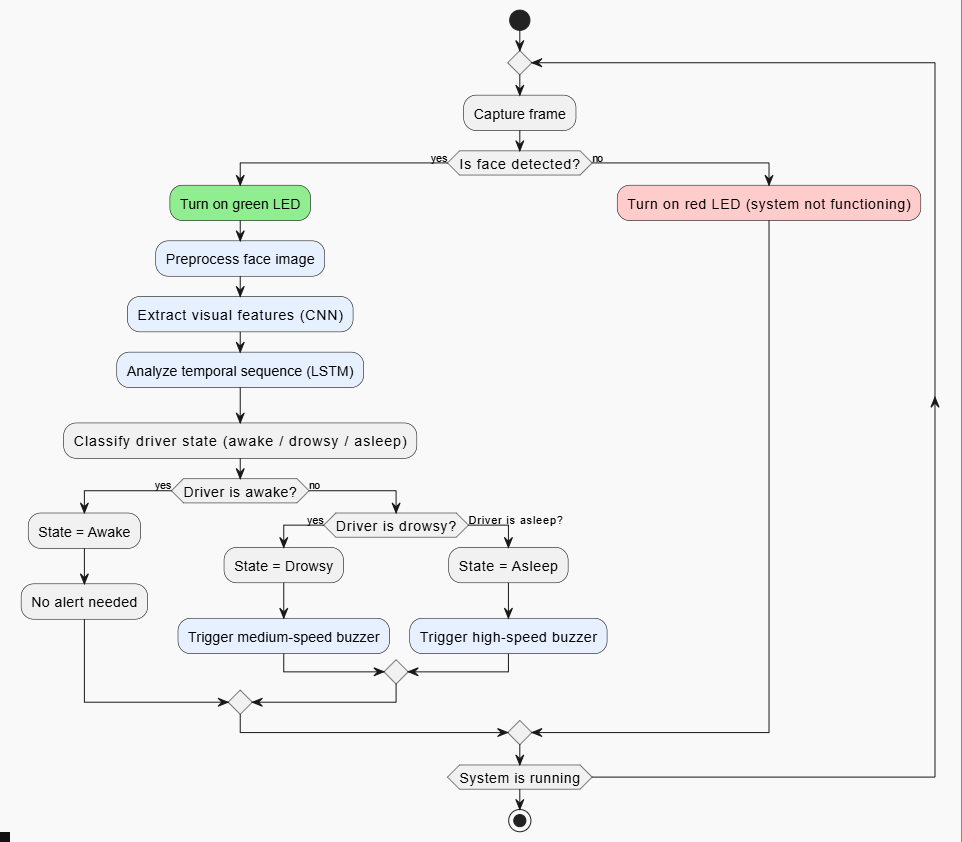


Figure 4:Activity diagram

### 3.3.1 Sequence Diagram

The following diagram illustrates a sequence diagram of the interactions between the proposed drowsiness detection system and the remaining components.

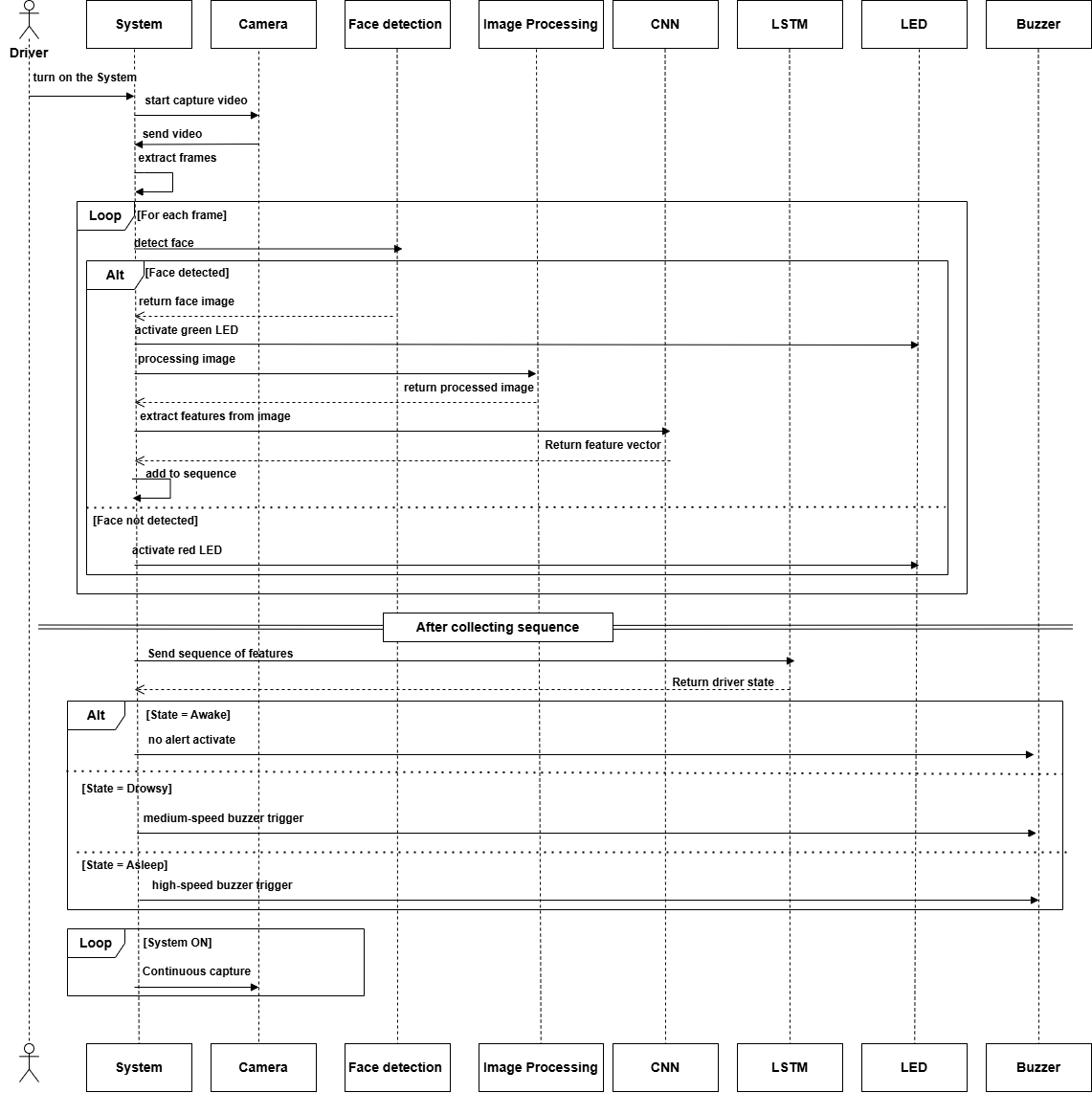
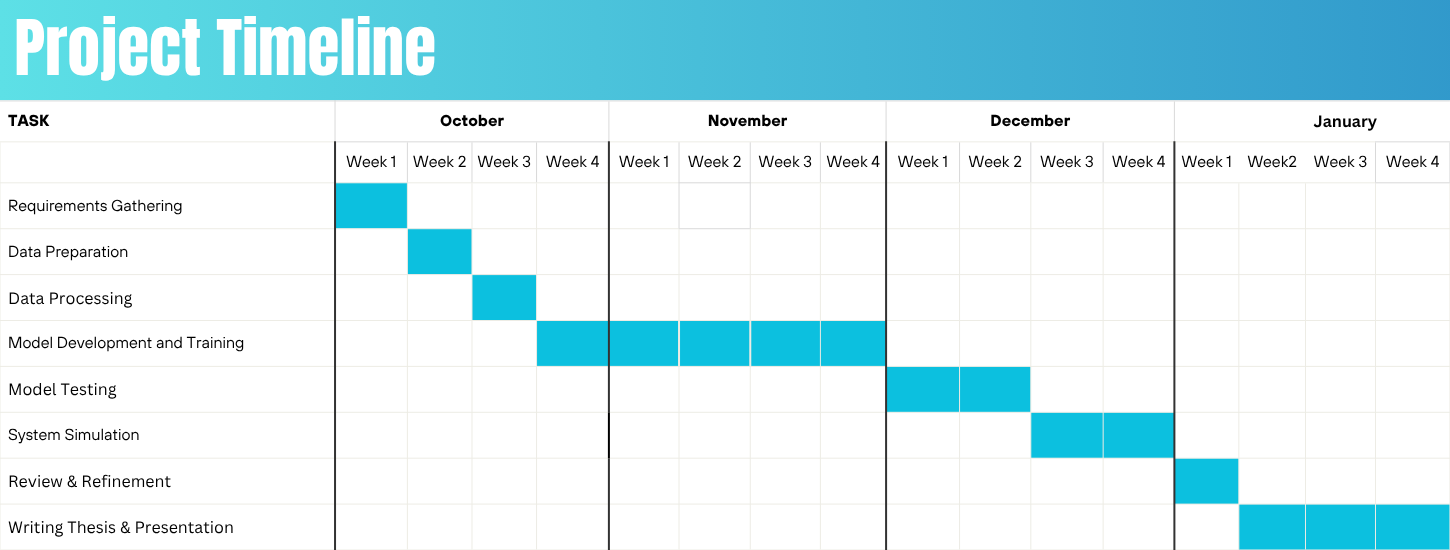
**

Figure 5:Sequence diagram

## 3.4 Project Timeline:

The following chart shows a suggested calendar to show when the project starts and ends. The tasks are divided into weeks for each month to help organize the work.

Table 2:Project Timeline



The circular life cycle diagram in Figure 6 shows how we followed the Agile methodology. This approach is based on working in short phases, with continuous updates and improvements. In each phase, we focused on specific tasks like data preparation, model training. This helped us stay flexible and make changes whenever we faced any issues or needed to improve something.

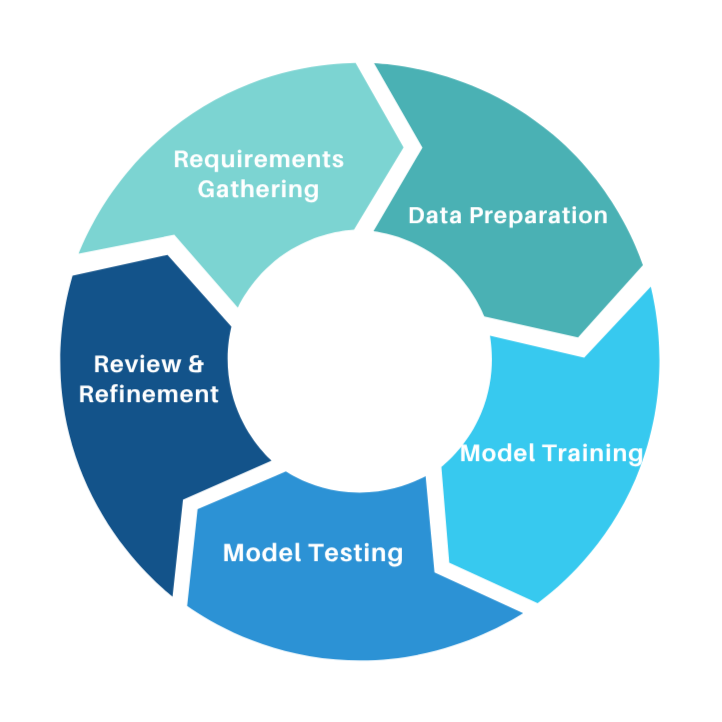


Figure 6:Project life cycle

# 

# CHAPTER 4 - IMPLEMENTATION

# This chapter presents the implementation of the system and how it has been developed and deployed.

In this chapter, the procedural steps for developing the driver drowsiness detection system are outlined. Initially, the dataset undergoes several preprocessing stages to ensure its quality and accuracy for training. Subsequently, the YOLOv11 model is trained on these prepared frames using the computational resources of Google Colab Pro. After training, the system is designed to provide effective alerts to the driver. The trained model is then deployed within a Graphical User Interface (GUI) for real-time operation and visualization. This chapter delves into the intricacies of each step, detailing the methodology behind data preparation, model training, and interface deployment. Additionally, the chapter provides the source code. [here](https://github.com/hananabuzainab/DriveSafe-AI-Based-Driver-Drowsiness-Detection-System-.git).

### 4.1 Data Description

In this project, multiple datasets were used to train and evaluate the proposed driver drowsiness detection system. The **NTHU Drowsy Driver Detection Dataset (NTHU-DDD)** served as the primary source, complemented by additional images from **YawDD**, **Closed Eyes in the Wild (CEW)**, the **Driver Behavior Image Dataset**, and relevant collections from **Kaggle**. Together, these datasets provide diverse facial expressions, lighting conditions, and participant characteristics, thereby enhancing the model’s generalization capability.

The system also includes a third class, **Absent**, representing images of empty seats. These images were collected separately to ensure diversity and help the model avoid false positives.

#### 4.1.1 NTHU Drowsy Driver Detection Dataset (NTHU-DDD)

The **NTHU Drowsy Driver Detection Dataset (NTHU-DDD)**, collected by the Computer Vision Laboratory at National Tsing Hua University, was used as the **primary dataset** in this project. This dataset is **not publicly available**, and access was granted after signing the Dataset License Agreement and obtaining approval from the head of the department[].

The dataset includes **36 subjects** of various ethnicities and genders, recorded under five driving scenarios: **BareFace (NoGlasses), Glasses, Sunglasses, Night-BareFace, and Night-Glasses**. Videos capture a range of driver behaviors, including yawning, slow blinking, nodding, laughing, talking, and looking aside, with two main statuses: **drowsy and non-drowsy**.

All videos were recorded using **active infrared (IR) illumination** in AVI format at 640x480 pixels. Night scenarios were recorded at **15 fps**, and other scenarios at **30 fps**. The dataset was split into **training and testing sets**, with 18 subjects (multiple videos per scenario) for training and the remaining 18 subjects (90 videos in total) for testing, covering a mixture of drowsy and non-drowsy behaviors.

This dataset was chosen as the primary source due to its **comprehensive coverage of drowsiness-related behaviors, lighting conditions, and facial characteristics**. Sample images are shown in Figure X.

#### 4.1.2 Yawning Detection Dataset (YawDD)

The YawDD dataset [https://universe.roboflow.com/utarlddv1/yawdd-nx0vr] includes two sub-datasets: one with normal facial expressions and another with drivers yawning. Participants are diverse in age, gender, and glasses usage. This dataset supplemented NTHU-DDD by providing additional yawning examples.Sample images are shown in Figure X

.

**4.1.3 Closed Eyes in the Wild (CEW)**

A portion of the **CEW dataset** [https://parnec.nuaa.edu.cn/\_upload/tpl/02/db/731/template731/pages/xtan/ClosedEyeDatabases.html], which contains images of eyes labeled as “closed” or “open,” was used, where only the **closed-eye images** were selected to support the detection of eye closure in our model.



#### 4.1.4 Driver Behavior Image Dataset

The Driver Behavior Image Dataset [https://data.mendeley.com/datasets/6y3g6vs2k4/2] includes images capturing various driver behaviors, such as looking aside, talking, and other A portion of this dataset was used to enrich training and testing with diverse driving behaviors.

**4.1.5**  **Kaggle Datasets**

In addition to the previously mentioned datasets, supplementary data were collected from Kaggle to enhance variability in participants, conditions, and modalities. The following four Kaggle datasets were utilized:

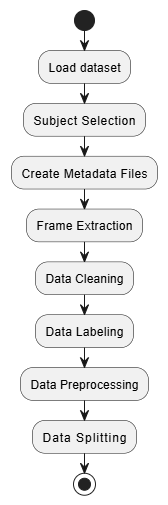
1. **UTA Real-Life Drowsiness Dataset (videos)[https://www.kaggle.com/datasets/rishab260/uta-reallife-drowsiness-dataset]:** Contains video sequences of drivers in real-life scenarios, providing natural variations in facial expressions and drowsiness behavior.
2. **Drowsiness Prediction Dataset[https://www.kaggle.com/datasets/rakibuleceruet/drowsiness-prediction-dataset]:** Includes labeled facial images for awake and drowsy states, supporting classification tasks in the system.
3. **NITYMED Dataset (videos)[https://www.kaggle.com/datasets/nikospetrellis/nitymed]:** Offers video data of drivers under different levels of alertness, enriching temporal features for drowsiness detection.
4. **Driver Inattention Detection Dataset (images)[https://www.kaggle.com/datasets/zeyad1mashhour/driver-inattention-detection-dataset?resource=download]:** Provides still images of drivers exhibiting attentive and inattentive states, helping improve robustness in distinguishing drowsiness cues.

#### 4.1.6 Absent Class

In addition to “Awake” and “Drowsy,” a third class, **Absent**, represents images where no driver is present (empty seats). This class helps the model distinguish between actual driver states and empty seats.These images are essential to avoid false positives and ensure accurate classification. They were captured with a mobile phone under different angles and lighting conditions (day and night), along with additional images collected from the web.



**4.2 Data Preparation**  
The data preparation process is a crucial step to ensure the quality and reliability of the driver drowsiness detection system. It includes selecting suitable subjects, creating metadata files, extracting and cleaning frames, labeling, preprocessing, and finally splitting the dataset into training and testing sets. These steps collectively ensure that the dataset is ready for effective model training and evaluation.



**4.2.1 Subject Selection**  
The datasets were carefully examined to select suitable subjects. In the NTHU-DDD dataset, 21 subjects were selected out of the original 36, while 15 subjects were excluded due to unclear facial features, which could negatively impact the accuracy of the results.

In the NITYMED dataset, 10 subjects were selected out of 21 in order to maintain a balanced distribution of gender (male and female) across the dataset.

This process ensured that the final dataset was both high quality and balanced, making it more reliable for training and evaluation.

**4.2.2 Metadata Creation**  
To ensure clear organization and management of the data, metadata files in CSV format were created for the videos in the NTHU-DDD dataset and for those recorded using a mobile phone. Each metadata file included the following information: video name, total number of frames, frame rate (frames per second), video duration, and video dimensions.

This step helped in understanding and organizing the dataset, making frame extraction easier.

**4.2.3 Frame Extraction**  
Videos from all datasets were split into individual frames so that they could be analyzed and used for model training and testing.

In the NTHU-DDD training set, videos recorded at 30 fps had one frame taken every 30 frames, and videos at 15 fps had one frame taken every 15 frames, reducing repeated frames while keeping enough variety. For the testing set, all videos were 30 fps and longer than the training videos, so one frame was taken every 60 frames to reduce the number of frames while preserving important content.

In the dataset recorded using a mobile phone, one frame was taken every 3 frames. This kept enough variety in the short videos while avoiding too many repeated frames.

For the NITYMED dataset, videos longer than 30 seconds had one frame taken every 25 frames to reduce repeated frames while keeping enough variation, while videos shorter than 30 seconds kept all frames to preserve all the information.

This gave a balanced and representative set of frames for all datasets while reducing repeated frames.

**4.2.4 Data Cleaning**After extracting frames from videos, all frames were carefully checked to ensure they were suitable for training and testing. Frames that were blurry, noisy, or otherwise low quality were removed.

For frames that were already provided as separate images from other datasets, each frame was manually reviewed, and any low quality frames were removed.

This data cleaning process ensured that the final dataset was high quality, consistent, and ready for model training and evaluation, reducing errors caused by poor-quality frames.

**42.5 Data Labeling**  
The frames from the NTHU-DDD dataset were originally labeled, but all frames were carefully reviewed manually after noticing some errors in the original labeling. This ensured that the labels were accurate and reliable.

Frames from the other datasets were also manually labeled and verified to make sure they were correct.

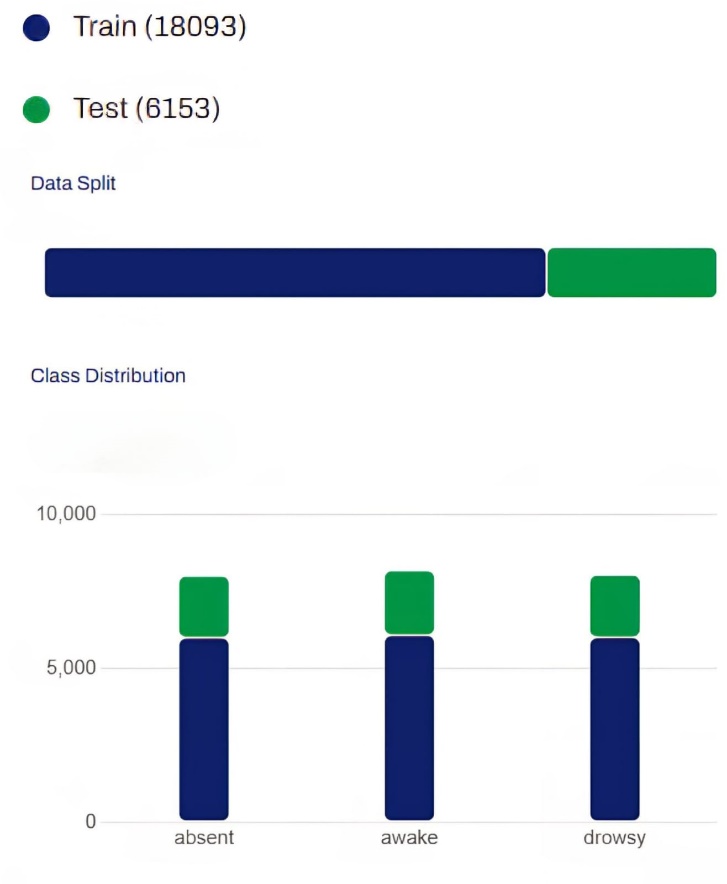
This careful labeling process made sure that the dataset was fully ready for training, with all frames correctly classified into the target classes: Awake, Drowsy, and Absent.

**4.2.6 Data Preprocessing**  
After cleaning and labeling, frames from all datasets were merged into a single dataset and converted to grayscale, simplifying the data while keeping all important facial movements including eyes, mouth, and head so the model could focus on them without being affected by color differences or other conditions.

This preprocessing ensured that all frames were consistent and ready for training and testing.

**4.2.7 Data Splitting**  
After preprocessing, the dataset was divided into training and testing sets. Approximately 75% of the frames were used for training, while the remaining 25% were used for testing. The testing set included different subjects from those in the training set.

This ensured that the model was evaluated on unseen data, providing a reliable measure of its performance.

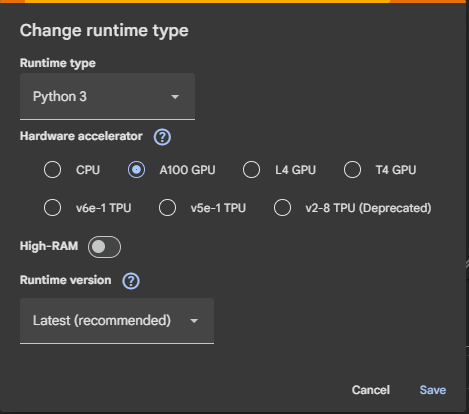


# 4.3 Training process

# The YOLOv11 model was trained using Google Colab Pro. Google Colab [ ], or Colaboratory, is an online platform where you can write and run Python code directly in your browser. It provides access to GPUs, which is extremely helpful for training deep learning models. Colab also allows easy sharing of notebooks and collaboration with others, enabling real-time comments and edits. After training, the YOLOv11 model is able to classify the driver's state into three categories: Awake, Drowsy, or Absent, providing accurate predictions on input images or video frames.

# 

# ****Enabling GPU Acceleration:****To efficiently train YOLOv11 on a large dataset with multiple epochs, GPU acceleration was enabled using the NVIDIA A100, significantly reducing training time and ensuring smooth processing of high-resolution images.



**Check GPU Status using NVIDIA SMI:** After enabling GPU acceleration, the NVIDIA System Management Interface (NVIDIA SMI) was used to verify the GPU status. This command-line tool provides detailed information about the GPU, including memory usage, temperature, and active processes, ensuring that the A100 GPU is properly recognized and ready for training the YOLOv11 model.

**Mount Google Drive:**Google Drive was mounted in the Colab environment to directly access the dataset and to store the training results, including model weights and logs. This approach allowed seamless data access and ensured that all outputs were safely saved for later use without relying on temporary Colab storage.

**Extract Dataset Files:** The dataset was extracted and organized into a clear hierarchical folder structure. The training set contains three subfolders corresponding to the classes: Awake, Drowsy, and Absent. Similarly, the testing set also contains the same three subfolders. This organization allows the YOLOv11 model to efficiently access and process images for both training and evaluation.

**Install and Load YOLO Library from Ultralytics:** The Ultralytics YOLO library was installed in Google Colab (pip install ultralytics) to enable the use of YOLOv11. A pre-trained YOLOv11 model was loaded directly for classification, simplifying the training workflow and ensuring compatibility with the dataset.

**Train YOLOv11 Model (including Validation):**A YOLOv11n classification model (yolo11n-cls.pt) was trained on a custom driver drowsiness dataset with three classes. The dataset contains 18,093 training images and 6,153 testing images. The model was trained for 100 epochs with a batch size of 32 and an image size of 640×640 on a GPU. Training progressed smoothly, showing a steady decrease in training loss and an increase in testing top-1 accuracy, reaching around 99% by epoch 50. The model architecture comprises 86 layers with approximately 1.5 million parameters, and pretrained weights were partially transferred to accelerate convergence. All training logs and results were saved to Google Drive.

**Data Augmentation During Training**

During the training process, the YOLOv11 framework automatically applied data augmentation techniques to the input images to improve the model’s stability and ability to generalize. Data augmentation artificially increases the variety of the training dataset by creating modified versions of existing images through random transformations.

In this project, YOLOv11 applied augmentation automatically on each batch during training. These transformations included scaling, cropping, brightness adjustment, and color changes. Such operations help the model become less sensitive to changes in lighting, camera angle, and driver appearance.

By training the model with these variations, data augmentation reduced overfitting and helped YOLOv11 learn general features that perform well on new, unseen data. As a result, the model could accurately recognize Awake, Drowsy, and Absent states even when the driver’s face appeared under different lighting conditions, angles, or expressions.

**Choice of Epochs and Training Progress Or Determining the Number of Epochs**

In deep learning models, an epoch represents one complete cycle in which the model processes the entire training dataset once. During each epoch, the model updates its internal parameters to minimize the difference between the predicted and actual outputs. Increasing the number of epochs allows the model to progressively improve its performance by learning more detailed features, while training for too many epochs may cause overfitting, where the model becomes too specialized to the training data and performs poorly on new inputs.

In this project, the YOLOv11 model was trained for 100 epochs using Google Colab Pro, which provided high performance GPU acceleration for efficient computation. During training, both training and validation losses were continuously monitored to ensure proper learning progress.

At the beginning of training, the training loss decreased rapidly from approximately 0.49 to around 0.20, indicating that the model was effectively learning to recognize facial cues related to driver drowsiness. As training progressed, the validation loss gradually stabilized near 0.03, while the classification accuracy exceeded 98%, reflecting strong generalization on unseen data. Around epoch 80, the loss and accuracy curves started to level off, showing that the model had learned most of the important patterns. Training the model for up to 100 epochs made the results more stable and reduced small changes in performance. It also showed that the model did not overfit and could still work well with new data.

Moreover, the close values between the training loss and validation loss during the later stages of training indicate that the model achieved a strong balance between learning and generalization. This similarity shows that the YOLOv11 model did not overfit the training data, as its performance on unseen validation samples remained consistent. The model successfully learned the essential facial and behavioral features needed to identify driver states effectively without memorizing the training examples.

This consistent reduction and balance between training and validation losses confirm that the model effectively learned to generalize key visual patterns, such as eye closure, yawning, and head position, which are important indicators of driver drowsiness. The choice of 100 epochs was suitable for YOLOv11, since large deep learning models usually need many training rounds to reach stable results and learn strong and clear features from the data.

**Result of Training**

During the training process of the YOLO model, a configuration file named args.yaml is automatically generated. This file contains all the parameters and settings that control how the model is trained. It serves as a complete record of the experiment and ensures the ability to reproduce the results.

The main parameters in this configuration include:

task: Defines the type of task for the model, here it is set to classify, meaning the model performs image classification.

**mode**: Specifies the operation mode, which is train in this case.

**model**: Indicates the pre-trained YOLOv11 model used as a starting point (yolo11n-cls.pt).

**data**: The path to the dataset used for training and evaluation.

**epochs**: The total number of training epochs (100 in this project), representing full passes over the training data.

**batch**: Batch size (32), controlling how many images are processed before updating model weights.

**imgsz**: Image size (640×640 pixels) for input images during training.

**save**: Enables saving the trained model and associated outputs.

**device**: Specifies the hardware used for training '0' refers to GPU 0.

**project**: The directory path where training outputs, including weights and logs, are saved.

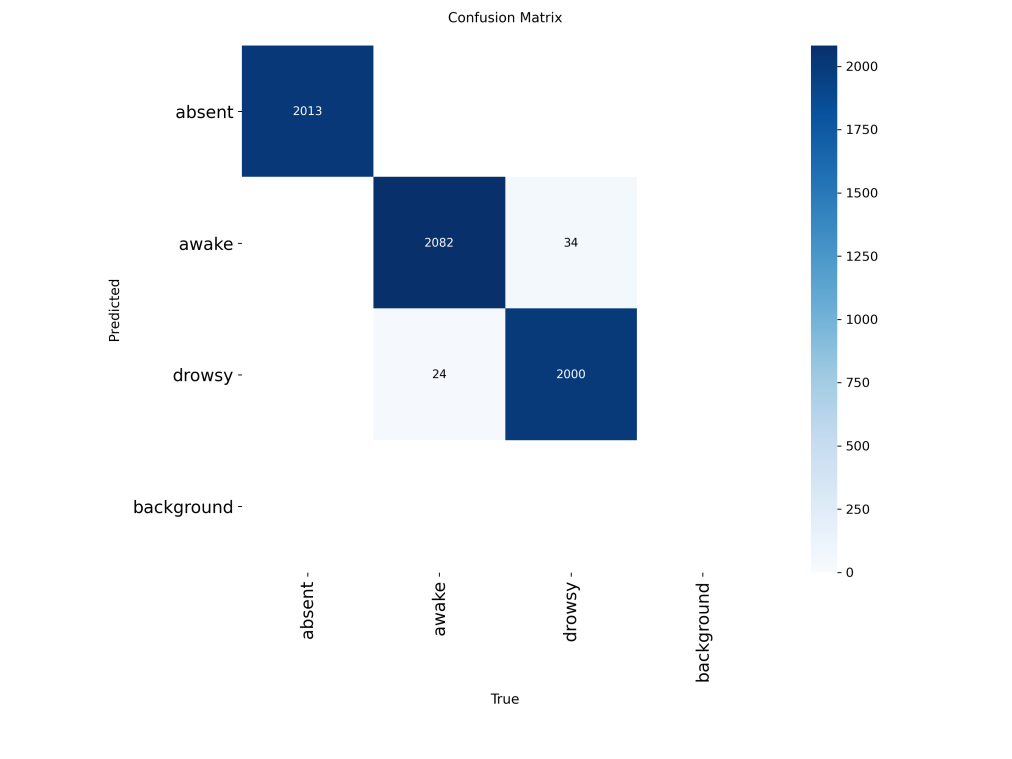
**name**: Name of the experiment, used to organize results in subfolders.

**optimizer**: Optimization algorithm for updating model weights, auto allows YOLO to select the best optimizer.

**plots**: Enables generation of plots for training loss and accuracy.

**lr0**: Initial learning rate for weight updates.

**auto\_augment**: Activates automatic data augmentation (randaugment) to improve generalization and reduce overfitting.



This matrix highlights the model's strong performance in predicting driver states. The confusion matrix is a tool used to evaluate the performance of a classification model by summarizing the correct predictions in comparison to the actual labels.

**Breakdown of the Matrix**

**1- Absent (True) Predictions:**

* Correctly predicted as Absent: 2013 instances
* Incorrectly predicted as Awake: 0 instances
* Incorrectly predicted as Drowsy: 0 instances

**2- Awake (True) Predictions:**

* Correctly predicted as Awake: 2082 instances
* Incorrectly predicted as Absent: 0 instances
* Incorrectly predicted as Drowsy: 34 instances

**3- Drowsy (True) Predictions:**

* Correctly predicted as Drowsy: 2000 instances
* Incorrectly predicted as Absent: 0 instances
* Incorrectly predicted as Awake: 24 instances

**Sum of each column:**

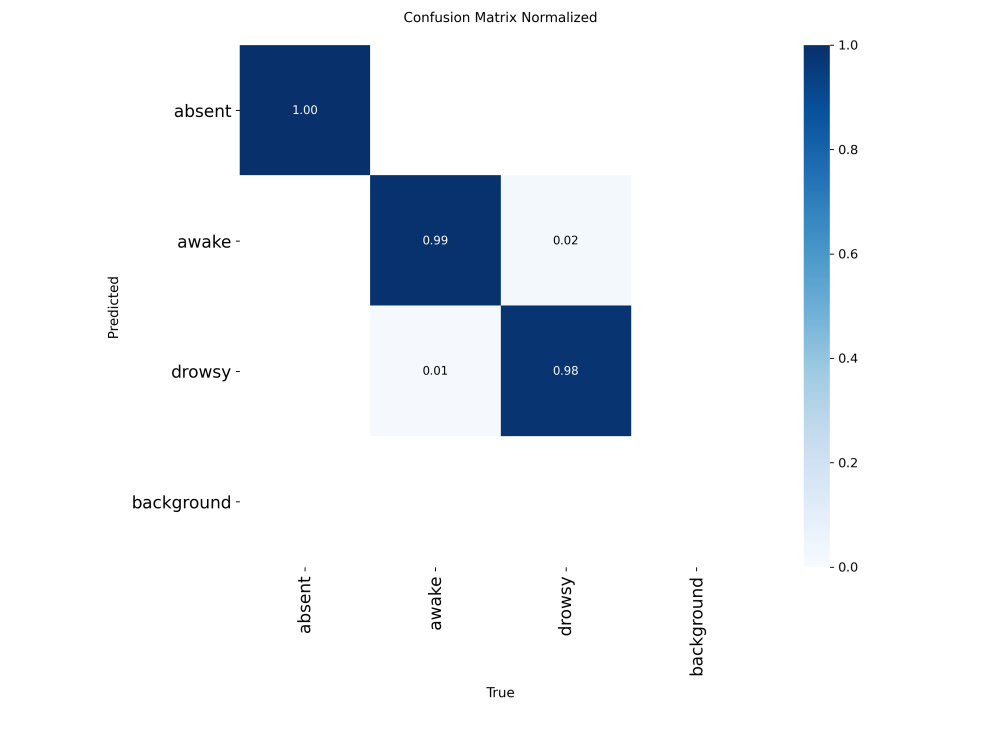
* Absent column: 2013 + 0 + 0 + 0 = 2013
* Awake column: 0 + 2082 + 24 + 0 = 2106
* Drowsy column: 0 + 34 + 2000 + 0 = 2034

**Interpretation**

* High Accuracy:
* The model perfectly predicts the Absent class with 2013 correct predictions.
* Most predictions for Awake (2082) and Drowsy (2000) are correct.

**Conclusion**

* The model accurately classifies the majority of driver states.
* Absent class is perfectly detected.



This figure shows the normalized confusion matrix for the YOLO model on the driver drowsiness detection dataset. Normalization expresses each cell as a percentage of the true class, which helps to better understand the relative performance of the model across classes.

**Analysis:**

* Absent: Perfectly classified (100%), with no confusion with other classes.
* Awake: Mostly correct )99%( correctly predicted, with a slight misclassification as Drowsy.
* Drowsy: Mostly correct )98%( correctly predicted, with a small misclassification as Awake.

**Interpretation:**

* The model indicates high overall accuracy.
* Slight confusion occurs between Awake and Drowsy, which is expected due to the similarity in these driver states.

The normalized confusion matrix provides a clear visual representation of the model’s performance in relative terms.

**Evaluation Metrics: Precision, Recall, and F1-Score**

To better assess the performance of the YOLOv11 model on the driver drowsiness detection dataset, three standard evaluation metrics were calculated for each class: Precision, Recall, and F1-Score. These metrics provide a detailed understanding of the model’s ability to correctly classify each driver state.

Precision measures the percentage of correctly predicted instances among all instances predicted for a class.

Recall measures the percentage of correctly predicted instances among all actual instances of a class.

F1-Score is the harmonic mean of precision and recall, providing a single value that balances both metrics.

**Results:**

* **Absent**: Achieved perfect results, with precision, recall, and F1-Score all equal to 100%, indicating the model reliably detected empty seats.
* **Awake**: Achieved a precision of 98.9%, recall of 98.4%, and F1-Score of 98.6%, showing that most awake instances were correctly classified with only slight confusion with the Drowsy class.
* **Drowsy**: Achieved a precision of 98.3%, recall of 98.8%, and F1-Score of 98.5%, reflecting that the model correctly recognized nearly all drowsy instances, with infrequent misclassification as Awake.

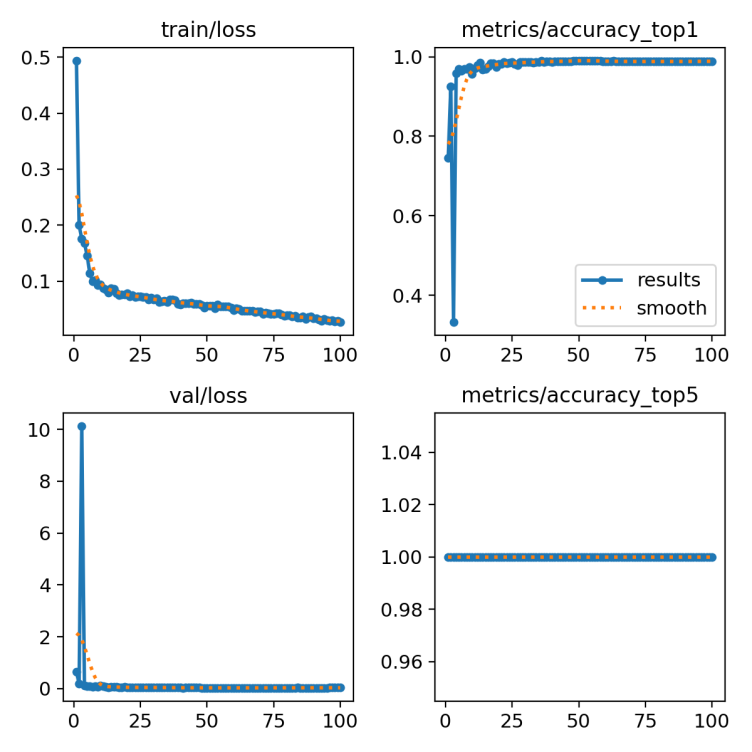
**Interpretation:**

The results indicate that the model performs remarkably well across all driver states.

The Absent class was perfectly classified, while the Awake and Drowsy classes achieved high precision and recall with slight confusion between them.

**Conclusion:**

Overall, the F1-Scores above 98% for all classes indicate that the YOLOv11 model maintains strong and balanced performance, making it highly suitable for real-time driver drowsiness detection.



This figure illustrates the performance of the YOLO model during training through four key curves:

Train Loss: Shows how the training loss decreases over epochs, indicating that the model is learning to fit the training data.

Validation Loss (Val Loss): Reflects the model's performance on unseen validation data. A decreasing validation loss suggests good generalization.

Training Accuracy (Metrics/Accuracy\_top1): Measures the percentage of correctly predicted instances in the training set. Increasing accuracy indicates that the model is learning effectively.

Top-5 Accuracy (Metrics/Accuracy\_top5): Since the dataset contains only 3 classes, Top-5 accuracy is always 1 (100%), meaning the true class is always within the top 5 predictions.

**Observations:**

* The training and validation losses converge, indicating stable learning without overfitting.
* Top-1 accuracy steadily increases, showing improvement in precise predictions.

**Interpretation:**

* The model shows strong learning behavior, with decreasing losses and increasing top-1 accuracy.
* Convergence between training and validation curves indicates good generalization to unseen data.

**Conclusion from Training**

The training process of the YOLOv11 model for driver drowsiness detection showed high performance and stability. Over 100 epochs, the model successfully learned to recognize key visual cues, such as eye closure, yawning, and head position, which are critical indicators of driver states. Both training and validation losses decreased steadily and converged, indicating effective learning without signs of overfitting.

Evaluation using the confusion matrix and standard metrics (Precision, Recall, and F1-Score) revealed that the Absent class was perfectly classified, while the Awake and Drowsy classes achieved values above 98%, with only slight confusion between them. These results confirm that the model generalized well to unseen data.

Automatic data augmentation further enhanced the model's robustness, enabling it to handle variations in lighting, facial orientation, and expressions. The best-performing weights were saved as best.pt, representing the epoch with the highest validation accuracy, and are ready for deployment in real-time detection.

Overall, the training process proved that the YOLOv11 model is accurate, stable, and capable of real-time driver drowsiness detection, making it suitable for practical applications.

# Reference list

[1] World Health Organization. (2023, December). Road traffic injuries. World Health Organization. <https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries>

[2] Palestinian Central Bureau of Statistics. (2023, September 4). Casualties in road traffic accidents in Palestine by governorate and type of injury, 2022. <https://www.pcbs.gov.ps/statisticsIndicatorsTables.aspx?lang=en&table_id=2037>

[3] Amoadu, M., Ansah, E. W., & Sarfo, J. O. (2023). Psychosocial work factors, road traffic accidents and risky driving behaviours in low-and middle-income countries: a scoping review. IATSS research, 47(2), 240-250. <https://doi.org/10.1016/j.iatssr.2023.03.005>

[4] AAA Foundation for Traffic Safety. (2024, March). Drowsy driving in fatal crashes, United States, 2017–2021. [https://aaafoundation.org/drowsy-driving-in-fatal-crashes-united-states-2017-2021 /](https://aaafoundation.org/drowsy-driving-in-fatal-crashes-united-states-2017-2021%20/)

[5] Fundación MAPFRE, "Fatigue detection system," Fundación MAPFRE, 2023. [Online]. Available: <https://www.fundacionmapfre.org/en/education-outreach/road-safety/adas-systems/types/fatigue-detection-system>

[6] M. Arefnezhad, M. K. Daneshvar, and R. Boostani, “PERCLOS estimation from EEG signals using dynamic regression models,” \*Scientific Reports\*, vol. 12, no. 1, 2022. [Online]. Available: <https://www.nature.com/articles/s41598-022-05810-x>

[7] M. Awais, N. Badruddin, and M. Drieberg, “Driver drowsiness detection using physiological signals: A review,” \*Sensors\*, vol. 17, no. 3, p. 495, Mar. 2017. [Online]. Available: https://[www.mdpi.com/1424-](http://www.mdpi.com/1424-) 8220/17/9/1991

[8] M. Arefnezhad, B. Sadeghi, and A. A. Pouyan, “Driver drowsiness detection based on steering behavior,” \*Journal of Medical Signals and Sensors\*, vol. 9, no. 1, pp. 45–52, 2019. [Online]. Available: <https://doi.org/10.3390/s17030495>

[9] T. Soukupova and J. Cech, “Real-time eye blink detection using facial landmarks,” in \*Proc. 21st Computer Vision Winter Workshop\*, Rimske Toplice, Slovenia, 2016. [Online]. Available:

<https://vision.fe.uni-lj.si/cvww2016/proceedings/papers/05.pdf>

[10] Ahmed, M. I. B., Alabdulkarem, H., Alomair, F., Aldossary, D., Alahmari, M., Alhumaidan, M., ... & Zaman, G. (2023). A deep-learning approach to driver drowsiness detection. *Safety*, *9*(3), 65. https://[www.mdpi.com/2313-576X/9/3/65](http://www.mdpi.com/2313-576X/9/3/65)

[11] Madane, A., Singh, A., Fargade, S., & Dongare, A. (2025). *Real-time driver drowsiness detection using deep learning and computer vision techniques*. International Journal of Scientific Research in Science,Engineering and Technology, 12(3), 222–229. <https://doi.org/10.32628/IJSRSET2512335>

[12] Gomaa, M., Mahmoud, R., & Sarhan, A. (2022, August). A CNN-LSTM-based deep learning approach for driver drowsiness prediction. Journal of Engineering Research, 6. :

<https://digitalcommons.aaru.edu.jo/cgi/viewcontent.cgi?article=1209&context=erjeng>

[13] C.-H. Chou, "NTHU Drowsy Driver Detection Dataset (NTHU-DDD)," National Tsing Hua University, [Online]. Available: <http://cv.cs.nthu.edu.tw/php/callforpaper/datasets/DDD/>