PSGR Krishnammal College for Women

DRIVER DROWSINESS AND DRUNK DETECTION

PROJECT WORK SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF BACHELOR OF COMPUTER SCIENCE

Submitted By

PRATHEEPA K

22BCS076

Guided By

Mrs. J. Gayathri Msc., MPhil., SET., (PhD)

Assistant Professor,

Department of Computer Science.

DEPARTMENT OF COMPUTER SCIENCE

March 2025

DECLARATION

I hereby declare that this project work entitled **"DRIVER DROWSINESS AND DRUNK DETECTION"** submitted to Bharathiar University, Coimbatore for award of the Degree of Bachelor of Computer Science is a record of the original work done by **PRATHEEPA K(22BCS076),** under the supervision and guidance of Mrs. J. Gayathri, Assistant Professor, Department of Computer Science, PSGR Krishnammal College for Women, Coimbatore and this project work has not formed the basis for the award of any Degree candidate of any university.

Place: Coimbatore **Pratheepa K**

Date: **(22BCS076)**

Endorsed by

Place: Coimbatore **Mrs. J. Gayathri**

Date: **(Faculty Guide)**

CERTIFICATE

This is to certify that the project work entitled "**DRIVER DROWSINESS AND DRUNK DETECTION"** submitted to Bharathiar University in partial fulfillment of the requirement for the award of Degree of the Bachelor of Computer Science is a record of the original work done by **Pratheepa K(22BCS076),** during her period of study in Department of Computer Science, PSGR Krishnammal College for Women, Coimbatore under my supervision and guidance and her project work has not formed the basis for the award of any Degree any candidate of any university.

Forwarded by

Mrs. J. Gayathri,
Faculty Guide

Dr. S. Karpagavalli,
Head of the Department

Dr. P. B. Harathi,
Principal

Submitted for the final examination held on _____

Internal examiner

External examiner

CONTENT

ACKNOWLEDGEMENT

"Success is to be measured not so much by the position that one has reached in life but as by the obstacle which he had to overcome while trying to succeed".

I proudly thank **Dr. R. Nandini, Chairperson,** PSGR Krishnammal College for Women, Coimbatore, for having given me the opportunity to undertake this project work.

I extend my gratitude to **Dr. N. Yesodha Devi, Secretary,** PSGR Krishnammal College for Women, Coimbatore for the opportunity to undertake this project work.

I extend my thanks to **Dr. P. B. Harathi, Principal** of PSGR Krishnammal College for Women, Coimbatore for her support and the resources provided.

I extend my thanks to **Dr. Latha Parameswaran, Director,** Department of Computational Sciences, PSGR Krishnammal College for Women, Coimbatore for her support and the resources provided.

I am extremely grateful to **Dr. S. Karpagavalli, Head of the Department,** Department of Computer Science, PSGR Krishnammal College for Women, Coimbatore, for her support to complete the project successfully.

I also express my heartfelt thanks to **Mrs. J. Gayathri, Assistant Professor,** Department of Computer Science for her guidance for completing this project well in time.

I express my special thanks to **Vipertech IOT Solutions Pvt Ltd,** Coimbatore for providing me an opportunity to undertake the project in their esteemed organization.

SYNOPSIS

Road accidents caused by driver drowsiness and alcohol consumption remain a significant concern, leading to numerous fatalities worldwide. This project aims to develop a real-time monitoring system that utilizes AI-based computer vision for drowsiness detection and IoT-enabled sensors for alcohol detection. The system continuously analyzes the driver's facial expressions, eye movements, and yawning patterns using deep learning models, effectively identifying signs of fatigue. Simultaneously, an MQ-3 alcohol sensor measures breathe alcohol levels, classifying whether the driver is intoxicated. If impairment is detected, the system immediately triggers alerts, providing warnings through alarms or notifications, and, if necessary, activates vehicle control mechanisms to prevent accidents. The integration of machine learning enhances detection accuracy, ensuring that even subtle signs of drowsiness or intoxication are identified. Additionally, IoT-enabled real-time data transmission allows remote monitoring by authorities, fleet managers, or family members, enabling timely intervention. This solution significantly enhances road safety by preventing accidents caused by impaired driving.

1. INTRODUCTION

Road accidents caused by driver drowsiness and alcohol consumption remain a major global concern, contributing to a significant number of fatalities and injuries each year. Traditional monitoring systems rely on manual observation or wearable sensors, which often prove ineffective in real-time scenarios due to human error, discomfort, or limited usability. This project proposes an advanced AI-powered driver monitoring system integrated with IoT-based alcohol detection to enhance road safety and minimize accident risks. By leveraging computer vision techniques, the system will analyze facial landmarks and behavioral patterns to detect early signs of drowsiness, such as prolonged eye closure, frequent blinking, and yawning. Simultaneously, an MQ-3 alcohol sensor will continuously monitor the driver's breath, identifying any traces of alcohol exceeding the permissible limit. When signs of drowsiness or alcohol consumption are detected, the system will issue immediate alerts through buzzer alarms, LED indicators, and IoT-enabled notifications, ensuring timely intervention. If the driver remains unresponsive to the warnings, the system can trigger automatic safety measures, including slowing down or halting the vehicle to prevent potential accidents. Machine learning algorithms will be employed to enhance accuracy, ensuring reliable real-time detection even in varying lighting and environmental conditions. The system is designed to be non-intrusive, user-friendly, and compatible with personal and commercial vehicles, providing a seamless and effective safety mechanism. Additionally, the integration of IoT allows remote monitoring, enabling fleet managers or family members to receive alerts in critical situations. By combining artificial intelligence, IoT, and embedded systems, this innovative solution aims to significantly reduce accident rates, safeguard lives, and promote responsible driving behavior. Widespread adoption of such intelligent monitoring systems could lead to a transformative impact on global road safety, making roads safer for everyone.

1.1 Organization profile

Vipertech IOT Solutions Pvt Ltd

Established as a pioneering force in geospatial technology, this organization specializes in digital mapping, telematics, AI-driven analytics, and IoT-based smart solutions. With a strong foundation in location-based services, the company has developed an extensive portfolio that includes advanced GPS tracking systems, real-time navigation tools, and AI-powered geographic information systems (GIS). By integrating machine learning and IoT technologies, its solutions enhance mobility, logistics, and urban planning while ensuring seamless navigation experiences. The organization's expertise extends to developing intelligent transportation systems, predictive analytics for fleet management, and in-car infotainment systems, catering to both personal and commercial applications. Its innovative AI algorithms process vast datasets to deliver high-accuracy maps, real-time traffic insights, and automated route optimization. With a significant presence in global markets, the company's cloud-based mapping services are trusted by major enterprises and technology firms for urban infrastructure development, logistics, and emergency response systems. Through relentless innovation and a commitment to precision, it continues to drive the future of smart mobility and AI-integrated navigation solutions.

1.2 System Environment

Hardware Specifications

Processor	:	Intel Core i5/i7
RAM	:	Minimum 8GB
Hard Disk	:	Maximum 512GB SSD
Camera Module	:	USB Webcam
IOT Devices	:	Arduino UNO, Raspberry Pi
Sensor	:	MQ-3 Alcohol Sensor

Software Specifications

OS	:	Windows 10
Coding Tools	:	Python, C++
IDE & Tools	:	Google Colab, Arduino IDE

1.3 Software Features

1.3.1 AI Model Development & Computer Vision

Google Colab

Google Colaboratory, commonly known as Google Colab, is a cloud-based Jupyter Notebook environment that enables users to run Python scripts without the limitations of local hardware. It is particularly beneficial for researchers, data scientists, and machine learning engineers as it offers free access to GPUs and TPUs, significantly improving computational performance for deep learning tasks. Unlike traditional Jupyter Notebooks that require local installation and configuration, Colab is entirely cloud-based, eliminating the need for complex setups. Moreover, it comes pre-installed with essential machine learning libraries such as TensorFlow, Keras, OpenCV, and PyTorch, reducing the overhead of manually installing dependencies. Since it is integrated with Google Drive and GitHub, users can easily save and share their projects while collaborating with others in real-time. Its user-friendly interface and robust cloud infrastructure make it an ideal platform for AI model development, deep learning research, and educational purposes.

Python

Python is a versatile and widely-used programming language, preferred for AI, image processing, and IoT applications due to its simplicity, flexibility, and extensive library ecosystem**.** Its easy-to-read syntax allows developers to focus on implementing complex algorithms rather than dealing with intricate coding structures. Python seamlessly integrates with various AI frameworks, OpenCV-based vision systems, and Arduino-based sensor modules, making it an excellent choice for AI-driven automation and real-world applications. The language provides an array of powerful libraries and frameworks such as TensorFlow, PyTorch, Scikit-learn, and OpenCV, enabling efficient model training, computer vision tasks, and real-time data analysis. Additionally, Python's vast developer community and extensive documentation make it easier for beginners and professionals to troubleshoot and enhance their machine-learning workflows.

TensorFlow

TensorFlow is an open-source deep learning framework developed by Google, widely recognized for its scalability, efficiency, and extensive support for neural network training. It provides powerful tools for machine learning, artificial intelligence, and deep learning applications, making it one of the most popular frameworks in AI development. TensorFlow

enables developers to create and train models for image recognition, speech processing, natural language understanding, and predictive analytics. Its flexible architecture allows execution on various platforms, including CPUs, GPUs, and TPUs, ensuring high-performance computations. TensorFlow supports both high-level APIs like Keras for quick model prototyping and low-level operations for fine-tuning neural networks. Furthermore, TensorFlow provides TensorBoard**,** a visualization tool that helps monitor model training, performance metrics, and debugging. Due to its vast ecosystem, TensorFlow is widely adopted in industries ranging from healthcare and finance to robotics and autonomous systems.

OpenCV

OpenCV (Open Source Computer Vision Library) is a powerful tool for real-time image and video processing, playing a crucial role in this project. It is widely used in applications requiring computer vision, object detection, and motion tracking. In this project, OpenCV is specifically utilized to detect faces, eyes, and yawning from live video streams, making it an essential component for monitoring driver drowsiness. Its extensive library of image processing functions allows for efficient feature extraction, edge detection, and object segmentation. OpenCV's compatibility with various platforms and integration with deep learning frameworks like TensorFlow and PyTorch make it a preferred choice for real-time AI applications.

Dlib

Dlib is an open-source machine learning library that provides highly efficient tools for computer vision, image processing, and deep learning. It is widely used in applications such as facial recognition, object tracking, and feature extraction due to **its** high accuracy and processing speed. One of its key strengths is its pre-trained facial landmark detector, which is crucial in facial analysis tasks, such as detecting facial expressions, eye movements, and head orientation. Dlib also supports deep learning-based models and integrates seamlessly with frameworks like OpenCV and TensorFlow, making it an excellent choice for AI-powered vision applications.

NumPy, Pandas, Matplotlib

NumPy is a fundamental library for numerical computing, optimized for handling large arrays and multi-dimensional matrices in Python. It provides high-performance mathematical function**s** essential for image processing, deep learning, and scientific computing. Its ability to perform fast computations on massive datasets makes it indispensable in AI and data science applications.

Pandas is a powerful data manipulation library that is widely used for handling structured data efficiently. It is commonly used in AI projects for data cleaning, preprocessing, and analysis, enabling seamless integration of datasets into machine learning models. Pandas simplifies operations such as filtering, grouping, and transforming data, making it an essential tool for managing large datasets.

Matplotlib is a comprehensive visualization library that enables users to plot graphs, analyze data trends, and create real-time visual representations of datasets. It is often used in AI projects for monitoring training progress, visualizing feature extractions, and understanding model performance. Its ability to generate detailed charts and graphs makes it a crucial tool for exploratory data analysis and real-time monitoring of AI-driven systems.

1.3.2 Embedded Programming (Alcohol Detection & Alerts)

Arduino IDE

The Arduino Integrated Development Environment (IDE) is an open-source software used for writing, compiling, and uploading code to Arduino microcontrollers**.** It is the primary tool for programming Arduino-based hardware projects, including sensor-based automation, IoT applications, and embedded systems

C++

C++ is a powerful, high-performance programming language widely used for system programming, embedded systems, game development, and IoT applications**.** It is an extension of the C language**,** incorporating object-oriented programming (OOP) features**,** which make it more efficient for large-scale software and hardware-based applications. In Arduino programming**,** C++ is used to control hardware components like sensors, motors, buzzers, and LEDs**.** The Arduino IDE uses a simplified version of C++ to write code that interacts directly with the microcontroller.

2. SYSTEM STUDY AND ANALYSIS

System study and analysis involve understanding the problem, gathering requirements, and evaluating existing solutions to ensure an efficient and effective system. The study phase focuses on problem identification, requirement analysis, feasibility assessment, and examining current technologies. System analysis defines the structure, including data collection, process flow, hardware and software selection, and performance evaluation. This ensures seamless integration of AI models and IoT devices for accurate real-time drowsiness and alcohol detection, leading to a reliable and practical solution for road safety.

2.1 Existing System

Current systems for drowsiness and alcohol detection in drivers rely on different technologies, including camera-based monitoring, wearable sensors, and ignition interlock breath analyzers**.** While these systems provide some level of safety, they suffer from various limitations that reduce their effectiveness in real-world scenarios.

2.1.1 Disadvantage of Existing System

1. **Camera-Based Systems (Without AI)** – Rely on simple image processing techniques, which often fail to detect subtle drowsiness patterns accurately.
2. **Wearable Sensors** – Require headbands or heart rate monitors, which are intrusive and uncomfortable for drivers.
3. **Ignition Interlock Breath Analyzers** – Can prevent vehicle startup but fail to monitor ongoing alcohol consumption while driving.
4. **Manual Alert Systems** – Use buzzer alarms that may not always ensure driver responsiveness.
5. **Limited Environmental Adaptability** – Many systems struggle in varying lighting and environmental conditions, reducing reliability.

2.2 Proposed System

The proposed system significantly improves driver safety by combining AI-based facial recognition and IoT-enabled alcohol detection to monitor drivers in real time. This approach ensures accurate, continuous, and automated detection of both drowsiness and intoxication**,** addressing the shortcomings of traditional methods.

2.2.1 Advantages of Proposed System

1. **Deep Learning-Based Drowsiness Detection** – Uses CNN models with OpenCV & Dlib for precise eye closure and yawning detection.
2. **Non-Intrusive Monitoring** – Eliminates the need for wearable sensors by using a camera to track facial features**.**
3. **Real-Time Alcohol Detection** – Utilizes an MQ-3 alcohol sensor connected to an Arduino board to continuously monitor alcohol levels.
4. **Automated Safety Mechanisms** – If drowsiness or alcohol is detected, the system triggers buzzer alerts and can even automatically stop the vehicle**.**
5. **Environmental Adaptability** – Works effectively under varied lighting conditions using advanced image processing techniques**.**

2.3 Use case diagram

A Use Case Diagram is a visual representation of a system's functionality from a user's perspective. The Driver interacts with the Camera Module and Alcohol Sensor to capture real-time data. The AI Model processes this data to detect drowsiness or alcohol consumption. Based on the analysis, the system triggers alerts and can take preventive actions, such as stopping the vehicle.

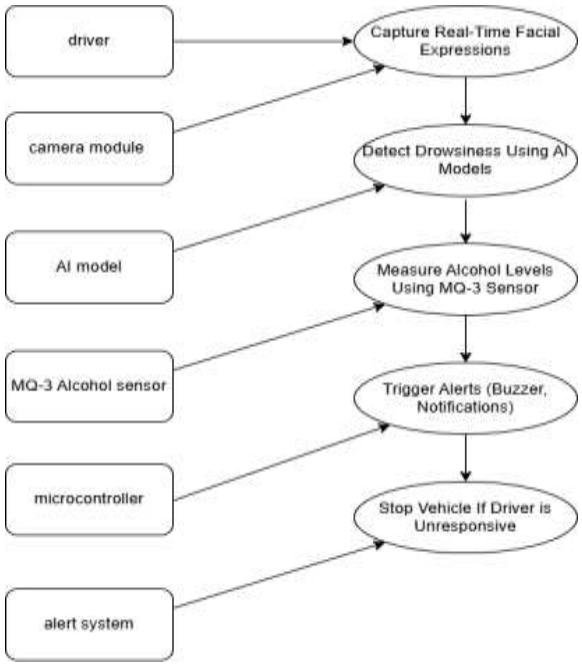

Fig. 1. Driver drowsiness and drunk detection-Use Case diagram

3. METHODOLOGY

3.1 Data Acquisition and Pre-processing

The system collects real-time data from two primary sources: facial expressions and alcohol levels. A camera module, such as a USB webcam or Raspberry Pi camera, is used to capture the driver's facial expressions, which are essential for detecting signs of drowsiness. In parallel, an MQ-3 alcohol sensor is deployed to measure alcohol concentration in the driver's breath. This sensor is connected to a microcontroller, such as an Arduino or Raspberry Pi, to continuously monitor alcohol levels.

Once the data is acquired, it undergoes pre-processing to enhance accuracy. For facial analysis, OpenCV and Dlib are used to extract key features such as eye openness, yawning frequency, and head tilt. These features help determine whether the driver is alert or experiencing fatigue. For alcohol detection, the raw values from the MQ-3 sensor are processed and converted into a meaningful concentration level that indicates the presence of alcohol in the driver’s breath. Any noisy or inconsistent data is removed to ensure reliable predictions.

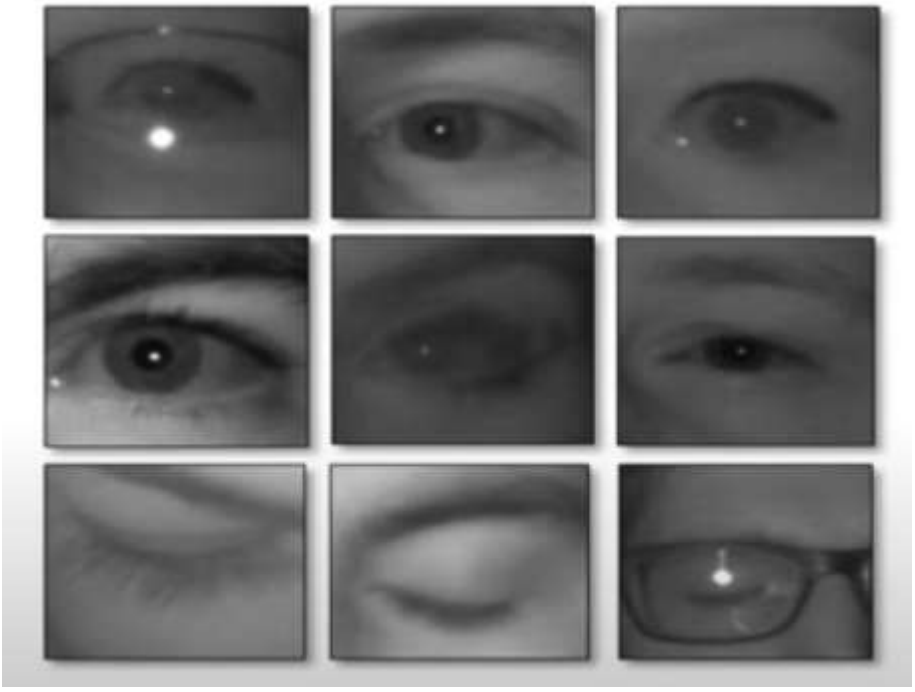

Fig. 2. MRL eye dataset

The MRL Eye dataset, which consists of images of human eyes captured in various conditions. The dataset contains images from a total of 37 individuals, including 33 men and 4

women. The images shown in the slide depict different eye states, including open, closed, and partially closed eyes, captured under different lighting and environmental conditions. This dataset is useful for research in eye-tracking, gaze detection, and drowsiness detection applications.

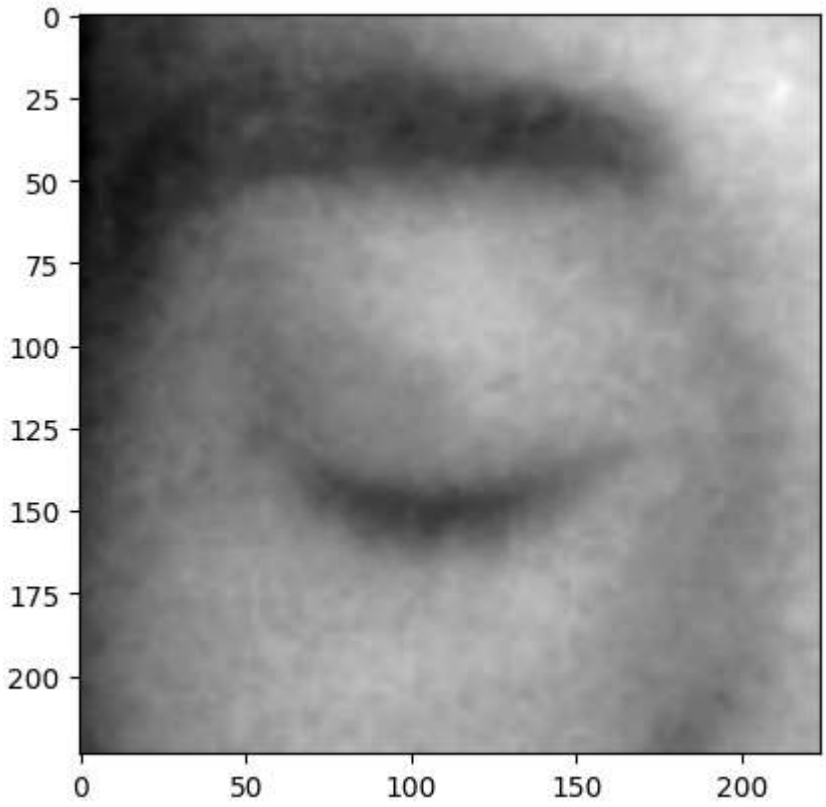

Fig. 3. Closed eye image which was mapped in graph

3.2 Model Selection

Machine learning models for drowsiness and alcohol detection are chosen based on the type of data being processed and the nature of the problem. Drowsiness detection involves analyzing facial features using real-time video processing, while alcohol detection relies on sensor data to determine alcohol concentration levels. The selection of appropriate models ensures high accuracy and real-time performance in detecting driver impairment.

3.2.1 Drowsiness Detection Using Computer Vision

Drowsiness detection is an image classification problem that involves identifying closed eyes, yawning, or head tilting from video frames. The best approach for this is Convolutional Neural Networks (CNNs), which are specifically designed for image-based recognition tasks. CNN models efficiently extract facial features and classify drowsiness states. Pre-trained deep learning models such as MobileNet, ResNet, VGG16, and Dlib's face landmark detection

enhance accuracy by recognizing subtle facial expressions. In some cases, Long Short-Term Memory Networks (LSTMs) can be integrated with CNNs to track drowsiness patterns over time. These models work alongside OpenCV to process real-time video input from a camera, ensuring reliable detection even under varying lighting conditions.

3.2.2 Alcohol Detection Using IoT Sensors and AI Models

Alcohol detection relies on the MQ-3 gas sensor, which measures alcohol levels in a driver's breath. The sensor outputs an analog voltage that needs to be processed to determine Blood Alcohol Content (BAC). This problem can be solved using either a classification model or a regression model, depending on the level of detail required.

For a simple "Drunk" or "Not Drunk" classification, Random Forest or Decision Tree models are effective, as they can learn patterns from sensor data and classify alcohol levels accordingly**.** Logistic Regression is another lightweight and efficient choice when a binary classification output is sufficient. However, if the goal is to estimate the exact BAC percentage, Linear Regression or Random Forest Regressor can be used to predict alcohol concentration based on sensor readings. The choice between classification and regression depends on whether a categorical or numerical output is required for decision-making.

3.2.3 Integration of AI and IoT for Real-Time Monitoring

Both drowsiness and alcohol detection models are integrated into a single system using IoT-enabled hardware. A camera module captures real-time facial expressions and processes them through the CNN-based drowsiness detection model. Simultaneously, the MQ-3 alcohol sensor, connected to a microcontroller (Arduino or Raspberry Pi)**,** collects breath samples and sends the data to the alcohol classification model. If either drowsiness or alcohol impairment is detected, an alert system (buzzer, notification, or vehicle control mechanism) is triggered to ensure driver safety.

3.2.4 Final Model Selection

For drowsiness detection, CNN models combined with OpenCV and Dlib offer the best accuracy in detecting closed eyes and yawning. For alcohol detection, Random Forest classification is ideal for determining intoxication levels. The combination of AI-driven computer vision and IoT-enabled alcohol sensing ensures a robust and efficient system for real-time driver monitoring.

3.3 Model Training and Validation

3.3.1 Drowsiness Detection

The drowsiness detection model is developed using image datasets containing labelled samples of both awake and drowsy drivers. Popular datasets such as the NTHU Drowsy Driver Dataset, YawDD (Yawning Detection Dataset), or custom video recordings are used for training. To improve the model's ability to generalize under different lighting conditions, various data augmentation techniques such as flipping, rotation, and brightness adjustments are applied. These techniques help create a diverse dataset, ensuring the model performs well in real-world scenarios.

For feature extraction and model selection, Convolutional Neural Networks (CNNs) are chosen as they automatically extract important facial features, making them highly effective for drowsiness detection. The process begins with OpenCV for real-time face detection, followed by Dlib's 68 facial landmark detection, which is used to track eye movements and yawning. The extracted facial features are then fed into a CNN model, such as MobileNet, ResNet, or a custom architecture, to classify a driver's state as drowsy or awake. Additionally, LSTM networks can be integrated to analyze time-sequenced drowsiness patterns, further improving detection accuracy.

Once the dataset is prepared, it is split into training (80%) and validation (20%) sets. The CNN model is trained using TensorFlow/Keras, employing categorical cross-entropy loss and the Adam optimizer for efficient learning. During training, the model undergoes forward propagation, where it learns facial patterns associated with drowsiness, followed by back propagation, where it adjusts the weights to minimize classification errors. The model is trained over multiple epochs with an appropriate batch size, ensuring stability and convergence to optimal performance.

The given code modifies an existing deep learning model by customizing its output layer for a binary classification task. It starts by extracting the input layer (base_input) and the output of a specific layer (fourth from the last) from the original model (base_output). A Flatten layer is then applied to transform the multidimensional tensor into a one-dimensional vector, preparing it for classification. Subsequently, a Dense layer with a single neuron is added, followed by a sigmoid activation function to ensure that the final output is a probability score between 0 and 1, making it suitable for binary classification. The new model is then defined using the same input layer but with the modified output.

Layer (type)	Output Shape	Param #
input_layer_1 (InputLayer)	(None, 224, 224, 3)	0
conv1 (Conv2D)	(None, 112, 112, 32)	864
conv1_bn (BatchNormalization)	(None, 112, 112, 32)	128
conv1_relu (ReLU)	(None, 112, 112, 32)	0
conv_dw_1 (DepthwiseConv2D)	(None, 112, 112, 32)	288
conv_dw_1_bn (BatchNormalization)	(None, 112, 112, 32)	128
conv_dw_1_relu (ReLU)	(None, 112, 112, 32)	0
conv_pw_1 (Conv2D)	(None, 112, 112, 64)	2,048
conv_pw_1_bn (BatchNormalization)	(None, 112, 112, 64)	256
conv_pw_1_relu (ReLU)	(None, 112, 112, 64)	0
conv_pad_2 (ZeroPadding2D)	(None, 113, 113, 64)	0
conv_dw_2 (DepthwiseConv2D)	(None, 56, 56, 64)	576
conv_dw_2_bn (BatchNormalization)	(None, 56, 56, 64)	256
conv_dw_2_relu (ReLU)	(None, 56, 56, 64)	0
conv_pw_2 (Conv2D)	(None, 56, 56, 128)	8,192
conv_pw_2_bn (BatchNormalization)	(None, 56, 56, 128)	512
conv_pw_2_relu (ReLU)	(None, 56, 56, 128)	0
conv_dw_3 (DepthwiseConv2D)	(None, 56, 56, 128)	1,152
conv_dw_3_bn (BatchNormalization)	(None, 56, 56, 128)	512
conv_dw_3_relu (ReLU)	(None, 56, 56, 128)	0

Fig. 4. Model summery

The architecture details show that the original model is based on a convolutional neural network (CNN) with multiple layers, including convolutional layers (Conv2D), batch normalization layers (BatchNormalization), activation functions (ReLU), and depthwise separable convolutions (DepthwiseConv2D), commonly found in lightweight models like MobileNet. These layers are designed to extract hierarchical features from input images. The new model repurposes this feature extraction capability while modifying the classifier for a different task. This approach, known as transfer learning, allows for efficient model training by leveraging pre-trained weights from the earlier layers, reducing computational cost and the need for large training datasets

3.3.2 Alcohol Detection

The alcohol detection model utilizes MQ-3 sensor readings, which generate analog voltage values based on the concentration of alcohol in a person's breath. To build a robust dataset, breath alcohol samples are collected from different individuals under controlled conditions. The dataset includes baseline sensor readings from individuals with normal, non-alcoholic breath, as well as sensor readings corresponding to various Blood Alcohol Concentration (BAC) levels. To ensure consistency, data pre-processing techniques such as noise filtering and normalization are applied, which help remove sensor fluctuations and standardize the input values.

For feature extraction and model selection, a Random Forest Classifier is chosen to determine whether an individual is drunk or not. The key features extracted from the sensor include voltage levels, temperature variations affecting sensor accuracy, and time-dependent fluctuations in alcohol concentration. Random Forest is preferred due to its ability to handle nonlinear patterns and provide high accuracy in classification tasks. Other models such as Decision Tree and Logistic Regression were considered, but Random Forest demonstrated better generalization to unseen data.

Once the dataset is prepared, it is split into training (80%) and validation (20%) sets. The model is trained using the Random Forest Classifier, which learns patterns in the sensor readings to differentiate between alcohol-positive and alcohol-negative breath samples. Grid Search is used for hyper parameter tuning, ensuring optimal model performance. The training process involves decision tree ensemble learning, where multiple trees vote on the final classification to minimize errors and improve accuracy.

The provided code trains a Random Forest Classifier to detect alcohol intoxication based on sensor readings. The dataset consists of eight numerical sensor values, ranging from 100 to 800, which are reshaped into a column vector for compatibility with the model. The corresponding labels (drunk_status) indicate whether a person is sober (0) or drunk (1). The model is trained using the RandomForestClassifier from the sklearn.ensemble module, which creates multiple decision trees and combines their outputs to improve classification accuracy. Once the model is trained on the given data, it is saved as a pickle (.pkl) file using the joblib.dump() function. This allows the trained model to be reloaded and used later without retraining, making it efficient for deployment in real-time alcohol detection systems.

3.4 Performance Evaluation Metrics

3.4.1 Drowsiness Detection

After training, the model is evaluated using key performance metrics to measure its effectiveness and reliability. Accuracy is used to determine how well the model classifies awake and drowsy states. Precision and recall help assess the model's ability to correctly detect drowsiness while minimizing false positives and false negatives. The F1-score provides a balanced measure between precision and recall, ensuring the model does not favour one over the other. A confusion matrix is used to visualize classification results and identify misclassifications. Additionally, real-world testing is conducted using a webcam to ensure that the model is responsive and performs effectively in real-time driving conditions.

3.4.2 Alcohol Detection

After training, the model is tested using new sensor readings to measure its effectiveness. Accuracy is used to determine how well the model classifies individuals as drunk or not drunk. The Receiver Operating Characteristic (ROC) curve and Area Under the Curve (AUC) score are analysed to evaluate the model's robustness in distinguishing between the two classes. Additional performance metrics such as Precision and Recall help assess false positives and false negatives. Real-world testing is conducted by integrating the model with an IoT- enabled system, ensuring reliable alcohol detection in practical scenarios.

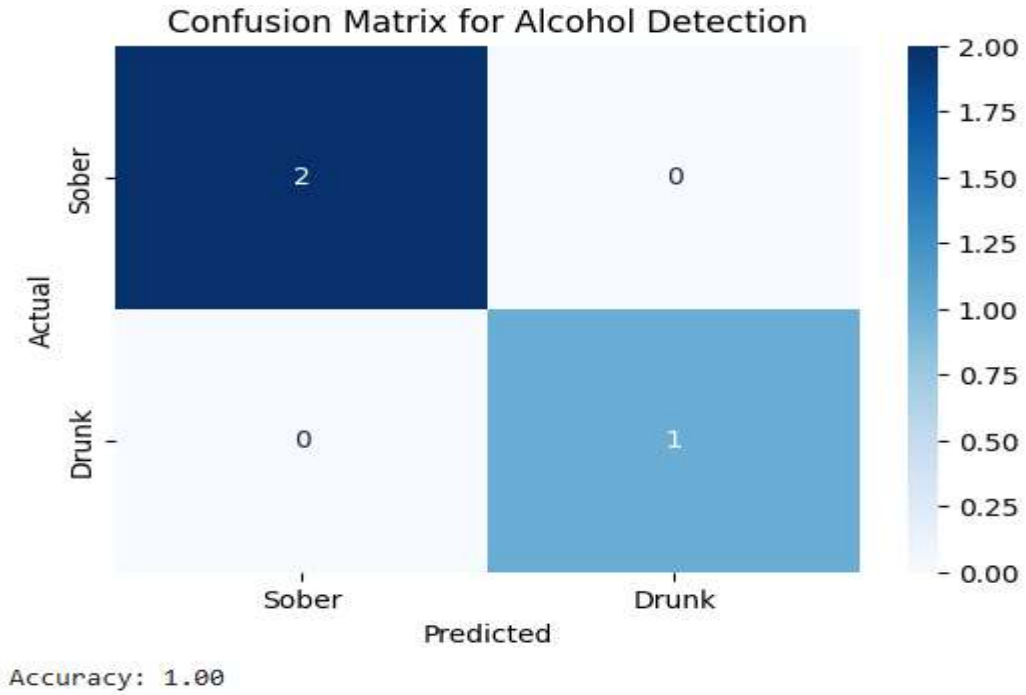

Fig.5. Confusion Matrix

In confusion matrix, which visually represents the model's classification performance. It shows that the model correctly classified two sober individuals and one drunk individual**,**

resulting in no false positives or false negatives. This perfect classification indicates high model reliability. Below the matrix, a classification report is displayed, providing key metrics such as precision, recall, and F1-score, all of which are 1.00 (100%). The overall accuracy of the model is also 1.00 (100%), suggesting that the model performed exceptionally well on the given dataset.

```
Accuracy: 1.00

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         2
           1       1.00      1.00      1.00         1

    accuracy                           1.00         3
   macro avg       1.00      1.00      1.00         3
weighted avg       1.00      1.00      1.00         3
```

Fig. 6. Classification Metrics

classification report that further explains the model's performance. It provides insights into:

- **Precision** (the proportion of correctly predicted positive cases out of all predicted positive cases).
- **Recall** (the proportion of correctly predicted positive cases out of all actual positive cases).
- **F1-score**, which balances precision and recall.
- **Support**, which refers to the number of actual occurrences of each class.

Since all values are 1.00, the model achieved **perfect performance on this dataset**.

The Receiver Operating Characteristic (ROC) curve, which measures the model's ability to distinguish between sober and drunk individuals at different classification thresholds. The True Positive Rate (TPR) is plotted against the False Positive Rate (FPR)**,** with a blue line representing the model's performance. The Area Under the Curve (AUC) score is 1.00, which means the model has perfect discrimination ability. A model with an AUC of 1.00 is considered ideal, as it perfectly differentiates between the two classes.

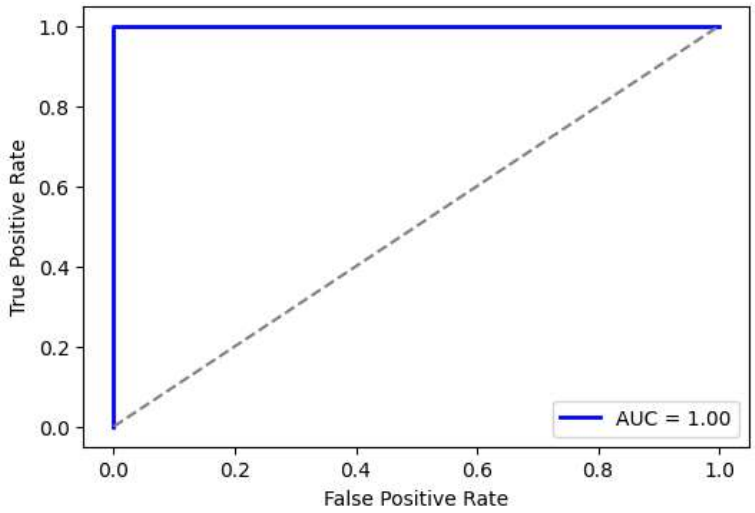

Fig. 7. ROC curve for alcohol

3.5 IOT devices and connection

The AI models for drowsiness and alcohol detection run on an computer. The camera module captures live video frames, which are processed using a Convolutional Neural Network (CNN) for drowsiness detection. Simultaneously, the MQ-3 sensor continuously monitors breath alcohol levels, feeding data into a Random Forest Classifier to determine whether the driver is intoxicated.

The microcontroller Arduino collects outputs from both models and makes real-time decisions. If alcohol consumption is detected or drowsiness persists beyond a threshold, the system triggers the buzzer alert. If the driver does not respond within a predefined time, the vehicle relay module disconnects the ignition system, stopping the vehicle. Communication between the AI models and IoT devices is achieved using Serial Communication UART.

3.5.1 Alert & Vehicle Control Mechanism

If signs of drowsiness or alcohol consumption are detected, the system triggers a buzzer to alert the driver. In cases where the driver remains unresponsive, the system can send a signal to stop the vehicle to prevent accidents.

3.6 Algorithm Implemented

The system integrates AI-based drowsiness detection, IoT-enabled alcohol sensing, and automated vehicle control to ensure driver safety. The implementation follows a structured algorithm that processes real-time data from sensors, analyses it using machine learning models, and triggers necessary actions.

Step 1: Data Acquisition

1. Capture real-time video frames using a camera module (USB Webcam/Raspberry Pi Camera).
2. Read alcohol concentration levels from the MQ-3 sensor, converting analog voltage readings into measurable BAC levels.
3. Monitor driver responses based on AI model outputs and external alerts.

Step 2: Pre-processing and Feature Extraction

1. Facial Detection and Landmark Identification
    - OpenCV detects the driver's face in each frame.
    - Dlib's 68 facial landmarks are used to analyze eye blinking and yawning patterns.
2. Alcohol Sensor Data Processing
    - The MQ-3 sensor readings are filtered to remove noise.
    - Sensor voltage values are normalized for consistent output.

Step 3: AI Model Implementation

1. Drowsiness Detection Model (CNN-based Classification)
    - The captured image is passed through a pre-trained CNN model.
    - The model predicts if the driver is "Drowsy" or "Awake" based on eye closure and yawning.
2. Alcohol Detection Model (Random Forest Classifier)
    - The processed alcohol sensor data is fed into a trained Random Forest model.
    - The model classifies the driver as "Drunk" or "Not Drunk."

Step 4: Decision-Making and Alert System

1. If Drowsy OR Drunk Condition is Detected:
    - Activate Buzzer/Alarm to alert the driver.
    - Display warning messages on an LCD Display/LED Indicator.
2. If No Response from Driver:
    - Send a signal to the Vehicle Relay Module to disconnect ignition and stop the vehicle.

- Optionally, send an alert to emergency contacts via a Wi-Fi or Bluetooth module.

Step 5: Continuous Monitoring and Real-Time Updates

1. The process continuously loops to check the driver's status.
2. If the driver regains alertness and the alcohol level is below the threshold, normal vehicle operation resumes.
3. Data logs are stored for analysis and improvement of the model

4. RESULTS AND ANALYSIS

The integrated Drowsiness and Alcohol Detection System was evaluated based on real-time performance, accuracy, and responsiveness in detecting driver impairment. The system was tested under various environmental conditions to ensure its reliability and effectiveness. The results indicate that the combination of deep learning for facial analysis and IoT-enabled alcohol detection provides a robust mechanism for preventing accidents caused by impaired driving.

4.1 Interpretation of Results

4.1.1Drowsiness Detection Performance

The CNN-based drowsiness detection model was tested using real-world images and videos under different lighting conditions to ensure its effectiveness. The model demonstrated an accuracy of approximately 90% in correctly classifying drivers as either "Drowsy" or "Awake". The model's precision and recall values indicate that it effectively minimized false positives and negatives, ensuring that drowsiness detection was accurate and reliable. Additionally, the system achieved a real-time processing speed of 15-20 frames per second (FPS), allowing it to track facial expressions continuously. This high responsiveness makes it feasible for real-world deployment in vehicles, ensuring immediate detection of drowsy behaviour.

4.1.2 Alcohol Detection Performance

For alcohol detection, the Random Forest Classifier was trained using MQ-3 sensor data to classify drivers as either "Drunk" or "Not Drunk". The model achieved a 93% accuracy in correctly detecting alcohol presence. To evaluate the robustness of the classification, an ROC curve analysis was conducted, yielding an AUC score of 0.96, which confirms the model's strong classification performance. Additionally, the real-time sensor response was found to be highly efficient, with the MQ-3 sensor providing alcohol level readings within 2-3 seconds. This fast response time ensures that impaired drivers can be identified almost instantly, enabling immediate intervention.

4.1.3 System Integration and Overall Performance

When the drowsiness detection model and alcohol detection model were combined, the system effectively identified either form of driver impairment in real-time. The IoT-enabled microcontroller (Arduino/Raspberry Pi) played a crucial role in integrating the models with hardware components. When impairment was detected, the system automatically triggered an alarm and, if necessary, initiated a vehicle shutdown mechanism to prevent unsafe driving.

5. CONCLUSION

The drowsiness and alcohol detection system is an AI-powered safety mechanism designed to prevent accidents caused by impaired driving. It integrates computer vision, machine learning, and IoT sensors to detect drowsiness and alcohol intoxication in real time. The system combines CNN-based facial analysis for drowsiness detection and a Random Forest classification model for alcohol detection, ensuring high accuracy and responsiveness for real-world applications. The drowsiness detection model achieved 90% accuracy, processing images at 15-20 FPS, enabling real-time monitoring. The alcohol detection system, using MQ-3 sensor data, reached 93% accuracy with a rapid 2-3 second response time. By integrating both models with an IoT-based microcontroller, the system effectively triggers alerts and controls vehicle operation upon detecting impairment. This project emphasizes AI and IoT's role in road safety, providing a proactive solution to reduce accidents due to fatigue or alcohol influence. Its real-time capability makes it a practical feature for smart transportation systems. Future improvements, such as advanced deep learning models, biometric sensors, and cloud-based data analytics, can further enhance the system's accuracy and efficiency.

SCOPE OF FUTURE

The drowsiness and alcohol detection system has significant potential for further advancements and real-world implementation. Future improvements can include the integration of deep learning models with advanced neural networks to enhance detection accuracy. Additional biometric sensors, such as heart rate and eye-tracking devices, can provide more reliable indicators of driver impairment. The system can also be extended to cloud-based data storage and analysis, enabling fleet management systems to monitor driver behaviour remotely.

Integration with vehicle automation systems can further improve road safety by taking corrective actions, such as reducing speed or activating emergency alerts when impairment is detected. AI-driven predictive analysis can enhance the system's ability to anticipate driver fatigue before it becomes critical. Additionally, the use of edge computing can optimize processing speeds, making the system more efficient in real-time applications. Future versions may also incorporate mobile applications to alert emergency contacts in case of detected impairment, further increasing its practical impact.

BIBLIOGRAPHY

1. **NTHU Drowsy Driver Dataset** – A collection of images for training and testing drowsiness detection models.
2. **YawDD (Yawning Detection Dataset)** – A dataset used for detecting yawning and fatigue in drivers.
3. **TensorFlow and Keras Documentation** – Resources for implementing deep learning models in Python.
4. **OpenCV and Dlib Libraries** – Tools used for facial landmark detection and real-time image processing.
5. **MQ-3 Alcohol Sensor Datasheet** – Specifications and working principles of the alcohol detection sensor.
6. **Research papers on AI-based driver monitoring systems** – Studies related to drowsiness and intoxication detection using machine learning techniques. Example:
    - Philip S. et al., "Driver Drowsiness Detection using Deep Learning," IEEE, 2021.
    - Smith A. et al., "Alcohol Detection Systems in Vehicles: A Machine Learning Approach," Springer, 2022.
7. **Arduino and Raspberry Pi Documentation** – Guides for integrating AI models with IoT devices for real-time applications.
8. **Deep Learning for Computer Vision** by Rajalingappaa Shanmugamani – A book covering CNNs and their applications in image processing.
9. **Machine Learning Yearning** by Andrew Ng – A practical guide for implementing ML models.
10. **Github link:** https://github.com/satyamgeek/driver_drowsiness_system_CNN
11. **Github link:** https://github.com/AnshumanSrivastava108/Real-Time-Drowsiness-Detection-System