Drowsy Driver Detection Device Powered by TinyML model

Project Proposal

Supervisor: Prof. Emel Koç

Submitted by

Khaled Alrefai-220212334

Department of Computer Engineering

Submission Date:

21.10.2024

Table of Contents

1. Introduction

In today's world, many factors can lead to driver drowsiness, such as the side effects of medications, extended periods of driving, late work shifts, and more. It takes only a few seconds of inattention to cause fatal accidents, posing a serious threat not only to the drowsy driver but also to others on the road.

While recent advancements in Artificial Intelligence (AI) and Machine Learning (ML) have shown immense potential in enhancing safety, these technologies often require robust computing resources and an internet connection. However, what if we could develop a lightweight, stand-alone device powered by TinyML that performs a single task—detecting driver drowsiness—without the need for internet access or data collection? Such a device could be used by anyone, anywhere, offering a practical solution to a critical problem.

This project aims to design and implement a **Driver Drowsiness Detection System (DDDS)**. The device will monitor the driver's eyes for signs of drowsiness and alert them to restore their attention, helping them reach their destination safely. The design will prioritize minimal hardware and software requirements, with flexibility for future modifications and upgrades depending on project challenges and progress. The selected hardware and software components may change or evolve based on specific technical needs as the project unfolds.

2. Aims and Objectives

The primary objectives of this project are:

- To design and develop a Driver Drowsiness Detection System (DDDS), a car safety device aimed at preventing accidents caused by driver fatigue, thus potentially saving lives.
- To implement a system that accurately detects signs of drowsiness in drivers and alerts them through various means, such as a buzzer, flashing light, or alarm, helping them regain focus and avoid dangerous situations.

The hardware and software components used in the project may change or adapt based on technical challenges encountered during the development process. Solutions such as buzzers, lights, or alarms may be used in whole or in part depending on the progress of the project.

3. Problem Statement

Drowsy driving is a significant issue worldwide, posing serious risks and often leading to catastrophic consequences. The danger of driving while sleepy stems from the combination of fatigue and operating a vehicle, which can result in reduced attention, slower reaction times, and impaired decision-making. While drowsiness commonly occurs when a driver has not had enough sleep, it can also be triggered by factors such as untreated sleep disorders, medications, alcohol consumption, or working night shifts.

One of the greatest dangers of drowsiness is that drivers cannot precisely predict when they will fall asleep, making it particularly hazardous. Even if a driver does not completely fall asleep at the wheel, drowsiness impairs their ability to drive safely. It affects attention and reduces the ability to respond quickly to sudden changes in traffic or road conditions.

In addition to drowsiness, driver distraction also contributes to impaired driving performance. While distractions (e.g., objects or accidents diverting attention) are often triggered by external events, drowsiness results in a gradual withdrawal of attention from the road and traffic requirements. Despite their differences, both driver fatigue and distraction can have similar negative effects, such as slower reaction times and an increased risk of accidents.

The goal of this project is to address this critical issue by designing a system that can detect driver drowsiness early, providing an alert before dangerous levels of fatigue impact driving performance.

4. Literature Review

Research in the field of driver drowsiness detection has shown that eyelid closure is one of the most reliable indicators of sleepiness. Several systems have been developed that focus on detecting driver fatigue through the monitoring of eyelid behavior. Other related behaviors, such as increased blink frequency, slow eyelid movements, repeated blinking, and drooping posture, have also been found to be predictors of drowsiness.

One of the most commonly proposed methods in this area involves using cameras to monitor and analyze these behaviors. By capturing visual data, these systems can predict the onset of drowsiness before it reaches dangerous levels.

In this project, the focus will be on utilizing the eyelid closure technique, analyzing each frame of the driver's image to detect signs of drowsiness. The use of this technique aims to provide accurate and timely alerts to help prevent accidents caused by driver fatigue.

5. Project Scope

This project explores technology that is still in its early research and development stages. Based on the work completed thus far, the project will focus on the following areas of research and analysis:

Tiny Machine Learning (TinyML) for Resource-Constrained Microcontrollers: Investigating how machine learning models can be optimized and deployed on microcontrollers with limited computational and memory resources.

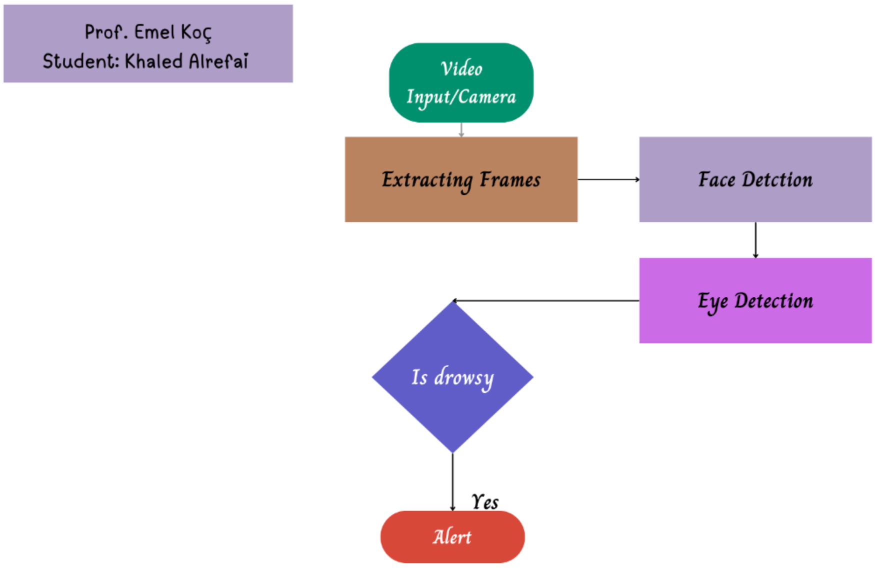
Microcontroller Selection for Machine Learning and AI: Analyzing different types of microcontrollers suitable for AI and machine learning applications, focusing on their performance, power efficiency, and compatibility with TinyML models.

Interfacing Multiple Microcontrollers: Exploring the best methods for connecting two microcontrollers—one responsible for handling low-level components (e.g., sensors, cameras) and programmed using C, and the other dedicated to processing input data using a machine learning model, programmed in Python. The communication between the two controllers will be a critical part of the design and may involve protocols like UART, SPI, or I2C.

The specific hardware and communication protocols selected for this project may evolve as the design challenges and project progress unfold, allowing for flexibility in implementation.

6. Methodology

The general architecture of the Driver Drowsiness Detection System (DDDS) is designed for ease of operation. The system primarily functions by capturing video of the driver's face using a camera. It measures the blinking rate of the eyes and issues an alert if the driver shows signs of drowsiness. Additionally, the system may utilize the PERCLOS (Percentage of Eyelid Closure Over the Pupil Over Time) technique, which detects gradual eyelid closure instead of just rapid blinks. Based on the PERCLOS score, the system will trigger an alarm to alert the driver.



Software and Libraries:

In the development of the system, several Python libraries will be employed to implement machine learning models and process the video input:

- **OpenCV (CV2):** An open-source library for machine learning, computer vision, and image processing. It will be used to analyze video streams from the camera and detect facial landmarks such as eyes and eyelid closure.

- **Keras**: A high-level neural network library based on Python, integrated with TensorFlow. It simplifies the development of machine learning models and is highly adopted for running on both CPUs and GPUs.
- **NumPy**: A Python library used for working with arrays, matrices, and linear algebra functions, which will assist in processing video frames and other data operations.
- **Matplotlib**: A low-level graph plotting library used for visualizing data such as PERCLOS scores and other performance metrics.

Approach:

Multiple research papers and methods related to drowsiness detection have been reviewed from sources like IEEE, Google Scholar, ScienceDirect, and ResearchGate to identify the most effective approach for this project.

The system employs machine vision to monitor the driver's face. A webcam is positioned to capture the driver's face in real time.
After detecting the face, the system focuses on the eyes to determine their state (open or closed). The algorithm assumes the driver is drowsy if the face is not detected in several consecutive frames.

The project will involve creating and training a machine learning model capable of detecting drowsiness by analyzing eye patterns.
Python will serve as the primary language for implementing this system, focusing on processing the video input and alerting the driver when signs of drowsiness are detected.
The use of TinyML will ensure the model operates efficiently on resource-constrained microcontrollers.
Depending on the progress and challenges encountered during development, the hardware and software components may be adjusted to ensure optimal performance.

7. Feasibility Study

The Driver Drowsiness Detection System (DDDS) project is feasible, but several risks and resource requirements need to be considered and mitigated throughout the development process.

7.1 Risks Involved

7.1.1 Technical Risks

Integrating multiple microcontrollers and ensuring smooth communication between them may pose technical challenges. This could potentially lead to delays in the project timeline.

Risk Mitigation:

I plan to conduct thorough testing to identify the best implementation strategy. I have alternative solutions and backup designs for potential hardware challenges, which will be tested and refined to ensure seamless integration.

7.1.2 Data Security

Handling sensitive user data is a common security concern in such systems.

Risk Mitigation:

This project does not collect or store any user data, as TinyML is employed for the detection task without learning over time. As a result, there are no risks related to data collection or privacy breaches.

7.1.3 Competitive Landscape

There may be competitors offering similar drowsiness detection devices or systems, which could impact the market positioning of this project.

Risk Mitigation:

I will focus on making this device a unique and competitive option by prioritizing lightweight design, low energy consumption, and affordability. These features will ensure the system stands out in the market while continually innovating to maintain a competitive edge.

7.1.4 Availability Risk

Certain components required for the system may not be available in the region, potentially affecting the project's progress.

Risk Mitigation:

I have developed multiple hardware and software designs to achieve the same functionality. For example, if a specific microcontroller is unavailable, I can use alternatives with similar capabilities. I will also adopt an iterative approach, starting with basic models (e.g., 0.1V) and upgrading incrementally (e.g., to 0.2V and beyond) as needed.

7.2 Resource Requirements

7.2.1 Computing Resources

The project will require computing resources for both hardware and software development. This includes access to development workstations, design tools, machine learning models, and various open-source platforms essential for the development of the DDDS.

7.2.2 Time

Time allocation is critical for the successful completion of the project. I will collaborate with Prof. Emel Koç to establish a clear project timeline, outlining specific phases for development, testing, and iterative improvements. The project schedule will be closely monitored to ensure timely completion.

8. Functional and Non-Functional Requirements

- **8.1 Functional Requirements**

  Drowsiness Detection: The device must reliably detect driver drowsiness by analyzing eye movement and closure through a camera feed.

  Alert System: The device must issue an immediate alert (such as a buzzer, flash, or alarm) when drowsiness is detected, ensuring the driver's attention is restored promptly.

- **8.2 Non-Functional Requirements**

  Performance: The system should process real-time data efficiently to avoid any delays in detection and alerts.

  Accuracy: The detection algorithm must be highly accurate to minimize false positives and negatives, ensuring reliable operation during long drives.

  Low Power Usage: The device must be energy-efficient to extend usage without frequent recharging, especially in cars with limited power resources.

  Cost-Effective: The system should be designed with affordable hardware to make it accessible for widespread use without sacrificing functionality or performance.

9. Tools/Technology

- **9.1 Hardware Requirements**
  Raspberry Pi: Serves as the main processing unit to run machine learning models and manage camera input.
  Surveillance Camera: Captures real-time video of the driver's face to monitor eye movement and detect drowsiness.
  Microcontrollers: (e.g., Atmega32) used for low-level control and integration of input components.
  Alert Units: Includes buzzers, light flashes, or screens to notify the driver when drowsiness is detected.
- **9.2 Software Requirements**
  Python (Version 3): Main programming language for developing machine learning algorithms and processing camera input.
  OpenCV: An open-source computer vision library used for image processing tasks, including facial detection and eyelid analysis.
  C Programming Language: Used for low-level programming of microcontrollers to handle hardware interactions and real-time data processing.
  MicroPython: A lean implementation of Python designed for microcontrollers, facilitating the development of lightweight applications.

10. Expertise

I possess valuable experience and expertise that align well with the technologies and tools needed for the Drowsy Driver Detection System (DDDS) project. My background includes transforming projects into real-world businesses, notably through my involvement with CareAnywhere, a mobile car repair shop launched in Dubai in early 2023. Additionally, I have experience in Android application development on Upwork, utilizing Java and Kotlin from 2022 to early 2023.

Summary of Previous Learning:

- Probability & Statistics for Machine Learning & Data Science - DeepLearning.AI (Issued Aug 2024).
- Using Python to Access Web Data - University of Michigan (Issued Jun 2024).
- Python Data Structures - University of Michigan (Issued Apr 2023).
- Edge Computing Fundamentals - LearnQuest (Issued Feb 2024).
- Introduction to Artificial Intelligence (AI) - IBM (Issued Jun 2023, with Honors).

In-Progress Learning:

- Practical Quantum Computing with IBM Qiskit for Beginners.
- Introduction to Quantum Information - Korea Advanced Institute of Science and Technology (KAIST).
- Introduction to Computer Vision and Image Processing – IBM.

11. Milestones

Milestone	Duration
Collection of Literature	Two Weeks
Study of Literature	Two Weeks
Analysis of Proposed Scheme	One Month
Preparation of Scheme/Model	One Month
Implementation of Scheme/Model	One Month
Analysis and Simulation	One Month
Result Formulation	Two Weeks
Final Write-up & Thesis Submission	Two Weeks

12-References

[1] **DDD TinyML: A TinyML-Based Driver Drowsiness Detection Model Using Deep Learning** by **Norah N. Alajlan** and **Dina M. Ibrahim, https://www.mdpi.com/1424-8220/23/12/5696.**

[2] **S. Ding, Z. Yuan, P. An, G. Xue, W. Sun and J. Zhao, "Cascaded Convolutional Neural Network with Attention Mechanism for Mobile EEG-based Driver Drowsiness Detection System,"** ***2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*****, San Diego, CA, USA, 2019, pp. 1457-1464, doi: 10.1109/BIBM47256.2019.8982938..**

[3] **Microcontrollers for Machine Learning and AI**
By yida, https://www.seeedstudio.com/blog/2019/10/24/microcontrollers-for-machine-learning-and-ai/?srsltid=AfmBOoq_tqfSFW1hB6tFuE_2auWNb7ci-SoNYaB9Mhlwmd0gZ4st9fnb.

[4] brandmed. "Is Firebase the Best Back-End Choice for a Flutter App?" Brandmed.com, brandmed.com/blog/development/is-firebase-the-best-back-end-choice-for-a-flutter-app. Accessed 12 Oct. 2023.

[5] **Get Started With Machine Learning on Arduino, by** Sandeep Mistry and Dominic Pajak Last revision10/21/2024, https://docs.arduino.cc/tutorials/nano-33-ble-sense/get-started-with-machine-learning/.

[6] Tiny Machine Learning for Resource-Constrained Microcontrollers, Riku Immonen,

Timo Hämäläinen, First published: 10 November 2022, https://doi.org/10.1155/2022/7437023.

[7] **TinyML: Machine Learning on ESP32 with MicroPython, by Tomas Posted on Jun 26, 2021 Updated on Jul 1, 2021, https://dev.to/tkeyo/tinyml-machine-learning-on-esp32-with-micropython-38a6**.

[8] Elena Magán et al., "Driver Drowsiness Detection by Applying Deep Learning Techniques to Sequences of Images," Applied Sciences, vol. 12, no. 3, pp. 1-25, 2022. https://doi.org/10.3390/app12031145.

[9] Suarez Buitrago, Eduard Antonio, "Drowsiness Detection System in Car Drivers Using Image Processing Techniques and Machine Learning," University of Pamplona – Faculty of Engineering and Architecture, http://repositoriodspace.unipamplona.edu.co/jspui/handle/20.500.12744/743.

[10] A. Caso, J. Rey de Castro, and E. Rosales-Mayor, "Sleep Habits and Traffic Accidents in Inter-Provincial Bus Drivers of Arequipa, Peru," Peruvian Journal of Experimental Medicine and Public Health, vol. 31, no. 4, pp. 707-711, 2014. https://europepmc.org/article/med/25597722.

[11] Mohammad Hijji et al., "FADS: An Intelligent Fatigue and Age Detection System," Mathematics, vol. 11, no. 5, pp. 1-16, 2023. https://www.mdpi.com/2227-7390/11/5/1174.

[Github]

- **TinyML project for driver drowsiness detetction. https://github.com/nourahnasser15/TinyML/tree/main?tab=readme-ov-file.**

- **drowsiness_control_sys_ADAS. https://github.com/dabourt/drowsiness_control_sys_ADAS**

- **Driver-Drowsiness-Detection-and-Accident-Prevention-System. https://github.com/Menna-Allah-21/-Driver-Drowsiness-Detection-and-Accident-Prevention-System.**

- **Driver Alertness Detection. https://github.com/shashankholla/DriverAlertnessSystem-SIH/tree/master**

- **Driver Drowsiness Detection System for Road Safety. https://github.com/ThuraAung1601/drowsiness-detection**

- **Emlearn Machine learning for microcontroller and embedded systems. https://github.com/emlearn/emlearn**