Drowsiness Detection System in Real-Time using Edge Computing for Enhanced Driver Safety

Akash Bappy
Biomedical Engineering (ITEE)
University of Oulu, Finland
akash.bappy@student.oulu.fi

Mitun Paul
Biomedical Engineering (ITEE)
University of Oulu, Finland
mitun.paul@student.oulu.fi

Ullas Chowdhury
Software Engineering and Information Systems
University of Oulu, Finland
ullas.chowdhury@student.oulu.fi

Tanvir Shuvo
Biomedical Engineering (ITEE)
University of Oulu, Finland
tanvir.shuvo@student.oulu.fi

Abstract

This paper describes the development and deployment of a real-time system for detecting driver sleepiness., based on edge computing, using the platform of NVIDIA Jetson Orin Nano to improve driver safety. The proposed system is using the deep learning model VGG16 to achieve 99% accuracy in detecting the state of eyes, the most critical component for the accuracy required for drowsiness monitoring. Operating exclusively on edge hardware, the system eliminates latency and cloud-based privacy issues. Therefore, this device would be very applicable on remote or low-connectivity areas. An escalating feedback mechanism using LEDs and a buzzer will carry out graded alerts based on eye closure duration and shall enable timely warnings to the driver. Apart from the existing solutions, which are cloud-based and sensor-heavy, this edge computing solution has several remarkable advantages in terms of cost, portability, and real-time processing. Future improvements will involve adaptive learning, additional sensors that can help improve robustness in a wide range of conditions and, therefore, enable this system to play an effective role in the mitigation of drowsy driving risks.

1 Introduction

1.1 Background

Drowsiness is a transition from an alert state to sleepiness, which sharply reduces focus and increases response time-all factors that create serious danger when performing tasks like driving [1]. It is considered one of the major culprits responsible for road accidents, injuries, death, and economic loss [2]. Based on a 2002 survey carried out by the National Highway Traffic Safety Administration (NHTSA), 37% of drivers reported falling asleep or dosing off while driving at some point in their entire lives [3]. According to the NHTSA's 2008 Traffic Safety Facts publication, drowsy driving contributes to 2.4% of fatal collisions [3]. One study found that between 1999 and 2008, drowsiness-related impaired driving was a factor in 7% of all collisions, 13.1% of non-fatal collisions requiring hospitalization, and 16.5% of fatal collisions. [4]. Since drowsiness is one of the the main causes behind road accidents, to have safety from these sudden incidents or rather reducing the limit of such incidences, the drowsiness detecting systems, and necessary alarm systems are becoming one of the necessities.

Detecting drowsiness and alerting the drivers is crucial for preventing accidents which are caused by fatigued drivers. These systems are generally known as drowsiness ditection systems. They mainly use cameras to observe the state of vehicle drivers during the driving, mainly the eyes to detect the signs of drowsiness. [5, 6, 7];To determine the stages of attentiveness of the drivers, facial features are mostly analyzed using machine learning and magic processing techniques. [6, 7]. Advanced systems however use activity monitoring through EEG signals and also the visual information from EOG to achieve more accurate drowsiness detection [8]. Such systems can classify drowsiness levels as 'awake', 'drowsy' or 'very drowsy' state with the accuracy of 80.6% [8]. When the system detects drowsiness, the systems alert the vehicle drivers [5, 6]. The non-intrusive approach of the camera-based detection system makes them suitable for real life in vehicle situations, enabling detection of drowsiness and alerting the drivers of their states [7].

1.2 Motivation

The main goal of this project is to develop a new, cutting-edge edge computing-assisted sleepiness detection system for drivers. Some of the major advantages of our system based on edge computing technology are as follows: Low Latency: The processing of data in a device itself means that one gets instant feedback devoid of any delay that generally arises with a cloud-based solution.

Real-time Processing: Real-time-operating systems run without breaks, and instant alerts upon detection of drowsy signs will aid in the prevention of accidents.

Dependency Reduction on the Cloud: Edge computing's capacity to operate without an internet connection improves system reliability and enables deployment, even in remote areas. Edge computing's capacity to operate without an internet connection improves system reliability and enables deployment, even in remote areas.

Our solution is based on a high-performance edge device, the Jetson Orin Nano from NVIDIA, which processes video in real time. It is using the deep learning model VGG16 optimized by TensorRT for performance enhancement on edge devices. An instant result from this classification, based on its results, will be provided to the driver through LEDs and a buzzer. This cost-effective and scalable solution will bring a sea change in road safety, resulting in reduced road accidents caused becasue of driver fatigue [9]. The proposed project is a move toward making this advanced technology in drowsiness detection accessible to more drivers and thus may save some lives and prevent injuries on roads around the world.

2 Related Work

2.1 Literature Review

In Finland, driver Fatigue accounts for 26.6% according to the ESRA3 [10] Report. Drowsiness detection systems would come in handy in the quest to improve road safety. These use several techniques whereby different means of monitoring driver behavior act to alert when fatigue signs are detected. Drowsiness Detection Systems use various features such as Steering Yaw angle, Wheel angle, Lane Detection, Brain Signals (EEG),Heart rate (ECG,PPG), Eye State, Yawning, and Visual Feature. In general, there are three types of techniques for drowsiness detection: vehicular, physiological, and behavioral parameter-based techniques.

2.1.1 Vehicular Parameter based Techniques

Vehicle parameter-based approaches to driver fatigue detection involve driving behaviors like changing lanes, variation in speed, or steering pattern through vehicle-component-mounted sensors, such as a steering wheel or brake pedal. Steering angle sensors, for example, detect steering reversals in order to provide an indication of drowsiness but are subject to errors from exogenous factors like road quality. Research shows the drawbacks of such methods since grip force is affected by mood and road conditions, and drowsiness might not necessarily change driving behaviors. New developments are focused on improving fatigue detection to accident prevention. Katyal et al. [11] suggested a lane and eye detection system in real time. that utilizes Hough transforms and the Viola-Jones method; it can successfully detect at night or long-distance driving even when the light condition is poor. Zhenhai et al. [12] presented a drowsiness detection model based on steering wheel angular velocity and achieved higher accuracy by analysis in a time series. Li et al. [13] implemented a real-time system with Steering Wheel Angles (SWA) and Approximate Entropy (ApEn) and reported an accuracy of 78.01%. In another study [14], SWA and yaw angles were used with a neural network Back Propagation and achieved an accuracy of 88.02% for detecting the level of fatigue. Taken together, the studies provide evidence that advanced technologies of fatigue detection have great potential to improve road safety.

2.1.2 Physiological Parameter-based Techniques

Driver drowsiness can be detected by observing physiological parameters, including respiration rate, heart rate, pulse and body temperature. That provides more reliable insight because physical states are reflected in the parameters directly. Fatigue can alter these parameters, such as lowering heart rate and blood pressure, enabling systems to alert drivers before they fall asleep. These methods, although good in effectiveness, usually employ invasive sensors, which give rise to discomfort and other implementation problems. Recent fatigue detection focuses on novel applications of physiological signals. Alzubi et al [15]. presented drowsiness detection using low-cost neuro-signal devices based on EEG; Li and Chung [16] attained 95% accuracy in applying wavelet analysis to Heart Rate Variability (HRV) data with a Support Vector Machine classifier. Rahim et al. [17] proposed an infrared pulse sensor to monitor the fatigue-induced declines in LF/HF ratio, which allows providing timely warnings. To shut down the mobile distractions, Leng et al. [18] proposed a wristband wearable device with PPG and skin response sensors, which could achieve 98.02% driving alerting performance using visual and vibrational cues.

2.1.3 Behavioral Parameter-based Techniques

Behavioral parameter-based techniques detect drowsiness by observing non-invasive driver behaviors like eye closure ratio (PERCLOS: Percentage of eye Closures), head position, eye blinking, facial expressions and yawning. PERCLOS measures the proportion of time the eyes are closed, while yawning detection tracks changes in mouth geometry. These devices record behavioral traits using cameras and computer vision. However, the precision of these data might be impacted by external factors like road conditions and lighting [19].Eye Tracking and Dynamic Template Matching [20] uses facial detection via the HSI color model and Sobel edge operator to track eyes, determining drowsiness by whether eyes are open or closed, achieving 99.01% accuracy. Mouth and Yawning Analysis [21] tracks yawning using SVM classifiers, improving detection compared to geometric methods, and helps detect driver fatigue early for safety. Assari and Rahmati's [22] Facial Expressions Method detects drowsiness based on facial expressions (eye closure, eyebrow rising, yawning) using infrared light and template matching, with effective performance even with beards or glasses. Yawning Extraction Method [23] use SVM to localize the mouth and Circular Hough Transform to detect yawning, achieving 98% accuracy in real driving conditions. Eye Closure and Head Postures Method [24] combines eye closure duration and head posture (left, right, forward, backward) to detect drowsiness, reaching 80% accuracy. Real-Time Analysis Using Eye and Yawning [25] use eye blinking, yawning, and head movements to detect fatigue in real time, performing well except when drivers wear glasses or have facial hair. Eye Blink Detection Method [26] system calculates eye-blinking frequency to detect drowsiness, achieving 90% efficiency in online and offline videos. Eye Tracking System [27] propose an eye-tracking system using PERCLOS and a top-down model to monitor drowsiness, which works in all lighting conditions. Eye Blink Monitoring Method: [28] use an eye-blink detection technique based on distance calculation to detect drowsiness, achieving 94% accuracy, but it performs poorly in low light.

2.2 Edge Computing and Real-time Processing

Real-time processing and edge computing are essential components of driver drowsiness detection system that offers significant advantages over cloud-based solutions. The importance of these technologies in this safety-critical application cannot be overstated.

2.2.1 Real-time Processing for Drowsiness Detection

The other important requirement in drowsiness detection is that the task inherently embodies time-sensitive characteristics for real-time processing. Amongst some of the most crucial reasons for low latency in such systems, the following are noted: **Immediate Feedback**: Drowsiness may set in very fast. Even the slightest inattention for a few seconds could prove disastrous, and real-time processing ensures that the warning is conveyed to the driver without any delay and could prevent accidents.

Continuous Monitoring: The state of the driver may change very fast, and therefore continuous analysis of physiological and behavioral cues is indispensable in real-time. This is achieved using edge computing that does not rely on unstable network connectivity [29]. **Rapid Response:** In critical drowsy cases, the system must be able to respond in real time by activating, for instance, driver assistance or even emergency stop. Such life-critical applications require real-time processing and decision-making.

2.2.2 Edge Computing in Safety-Critical Applications

Edge computing brings several benefits to safety-critical systems such as driver monitoring: **Less Latency:** Data can be processed on edge devices locally in order to return almost instantaneous results, very critical for timely intervention in drowsiness detection [30]. **Higher Reliability:** Edge computing reduces dependence on network connectivity, ensuring that the system will continue functioning even when internet connectivity is limited or lost [29]. **Improved Privacy**: Edge operation reduces the possibility of data breaches and would help to meet current data protection legislation [30]. **Resource Efficiency**: Computation-intensive operations would be conducted thanks to edge devices without affecting the vehicle's main computer system, enabling more complex algorithms of drowsiness detection. Research has proved the worth of edge computing in drivers' monitoring systems. For example, a study using the NVIDIA Jetson platform reached real-time drowsiness detection with high accuracy and low latency, where video streams are processed locally without dependence on the cloud [31].

2.2.3 Edge Computing vs. Cloud-Based Solutions

When comparing edge computing to cloud-based solutions for drowsiness detection, several key factors come into play:

Table 1: Comparison between Edge Computing and Cloud-Based Solutions

Factor	**Edge Computing**	**Cloud-Based Solutions**
Latency	Low (milliseconds)	Higher (seconds to minutes)
Data Security	High (local processing)	Lower (data transmission risks)
Reliability	High (independent operation)	Lower (network-dependent)
Scalability	Limited by hardware	Highly scalable
Cost	Higher initial investment	Lower upfront costs

Edge computing addresses the challenges of cloud-based systems by: **Minimizing Latency:** Local processing eliminates network delays, crucial for real-time drowsiness detection [30]. **Enhancing Data Security:** Sensitive driver data remains on the edge device, reducing the risk of interception during transmission [29]. **Improving Reliability:** Edge systems continue to function in areas with poor network coverage, ensuring constant monitoring. **Ensuring Responsiveness:** By processing data locally, edge computing provides immediate feedback, critical for driver safety [31]. In conclusion, edge computing and real-time processing are essential for effective drowsiness detection systems. They offer the low latency, high reliability, and enhanced security necessary for this safety-critical application, making them superior to cloud-based solutions in preventing fatigued drivers from causing accidents.

2.3 Comparison with Other Drowsiness Detection Approaches

2.3.1 Hardware Comparison

Traditional drowsiness detection systems traditionally use one or more of the following: **Desktop-based systems:** Even though these avail powerful GPUs, they offer no portability and can hardly afford real-time processing in a vehicle. **In-car cameras:** Even though they are integrated, often their processing power is limited and relies on an external computation provided to them. **Wearable devices:** Often intrusive to wear and sometimes cannot capture all the necessary visual cues. While on the other hand, The Jetson Orin Nano from NVIDIA provides a potent balance of power with portability. It delivers AI performance up to 40 TOPS in a compact size, surpassing many desktop-based solutions in terms of efficiency [32]. An edge computing approach means that video feeds can be analyzed in real time with no need for cloud connectivity, hence addressing the issue of latency, which is so important in drowsiness detection.

2.3.2 Cost and Complexity

At the pricing of US$499, it presents a much more economically viable option compared to custom-built systems or high-end desktop setups. Although somewhat expensive compared to entry-level camera systems, it gives considerably much higher processing capability and flexibility. The integrated nature of the Jetson platform further reduces system complexity because of the marriage of powerful hardware to a comprehensive software stack that is optimized for AI applications[33].

2.3.3 Face and Eye Detection Techniques

Face detection methods compared: Haar Cascade: Faster, less sensitive with changes in conditions. FaceNet: Higher accuracy, computing load is higher. Dlib: Good balance between computational time and accuracy. Facedetect: Edge-optimised and good balance between speed and accuracy. Among those, Facedetect would be the most appropriate to use in our system because it is supported by the edge computing platform Jetson Orin Nano. It allows faster inference times while still providing enough accuracy for real-time drowsiness detection.

Eye detection method comparison: Haar Cascade: Prone to false positives in complex scenes. Dlib: More accurate but computationally expensive. MediaPipe: A very decent trade-off between speed and accuracy; it shows quite robust performance under variable lighting conditions. MediaPipe was adopted in our system owing to its best performance on edge devices. It enjoys faster detection speed and maintains accuracy across various lighting conditions, something quite vital to ensure in on-board applications.

2.3.4 Deep Learning Models

When considering deep learning models for image classification: ResNet [34]: Deep architecture, high precision but maybe at the cost of slower inference. AlexNet [35]: Much less complex in architecture but somewhat faster with lower accuracy relative to the state-of-the-art models. MobileNet [36]: The main reason for using MobileNets is that, since they are applied to mobile devices, they attain a very good trade-off between speed and precision. VGG16 [37]: It has excellent feature extraction capabilities, hence presenting an excellent trade-off between depth and performance. In our project, VGG16 was picked as it has a very good precision with acceptable complexity that can be applied in real-time application. This architecture goes well with feature extraction for drowsiness detection tasks that result in a reliable output without overwhelming the resources of the Jetson Orin Nano. Bringing together hardware power with our carefully handpicked software components means real time, high accuracy, and cost-effectiveness for a drowsiness detection system compared to traditional approaches. This is an edge computing solution to the critical need of low-latency processing in safety-critical applications like driver monitoring.

2.4 Limitations in Existing Works

There exist many limitations with the drowsiness detection systems that are currently in place, and our project tries to cover those. Identifying those gaps and explaining our approach will demonstrate the unique value proposition of our system.

High Cost and Complexity: Current systems mostly depend on either expensive hardware setups or complicated multisensor arrays, neither of which is easily adaptable to widespread adoption [38]. **Cloud Dependence**: Some solutions even depend on cloud services to process the data, bringing latency and reliability concerns into the picture, mainly in areas with poor network connectivity[30]. **Real-Time Processing Limited**: Most systems cannot provide real-time processing since instant drowsiness detection and intervention are both very critical [38] [29]. **Poor Performance in Varying Conditions:** Most of the existing systems are not well designed to handle environmental variations, including low light conditions or even drivers wearing glasses. **Limited Portability:** The desktop or fixed camera systems cannot be portable to meet the requirements of various types of vehicles and driving conditions. **Privacy Issues**: Cloud-based systems that send out critical driver data raise privacy issues and may not conform to data protection laws [30]. **Accuracy and False Positives:**

Most of the systems either have low accuracy or high false alarm rates, leading to alert fatigue and missed detections.

2.5 Addressing the Gaps

These shortcomings have been addressed by our project by the following key innovations: **Computations at Edge in Real-Time:** This involves doing all computations at the edge with NVIDIA Jetson Orin Nano. It saves dependence on the cloud, hence massive latency reduction to an extent that real-time drowsiness detection is achieved to ensure immediate intervention. **Balancing Speed and Accuracy:** Face detect-Facedetect, along with the MediaPipe solution for eye tracking, does a great job in balancing speed with accuracy. We have chosen algorithms optimized for performance on edge devices that will empower the system to give a sound performance against variable lighting conditions at drivers with glasses. **Affordability and Portability:** Jetson Orin Nano is cost-effective with powerful performance. Compact in size, it is easy to embed into many types of vehicles. **Improved Privacy:** Due to the fact that processing will be performed entirely on an edge device, sensitive driver information does not necessarily need to be uploaded to the cloud, hence guaranteeing improved data privacy and regulatory compliance. **Adaptive performance:** Our system's approach at the edge enables continuous learning and adaptation to individual driver characteristics that, over time, can increase accuracy and reduce false positives. **Low-light performance:** The algorithms and hardware selected enable this system to perform in variable light conditions-a frequent limitation of camera-based systems. **Reduced Computational Cost:** We can accomplish high performance, without the utilization of costly power-hungry hardware by optimizing algorithms on the Jetson platform. The implementation of these allows our project to grant a more practical, efficient, and effective detection of drowsiness. Our system integrates edge computing with the carefully chosen algorithms and affordable hardware, placing it as an improvement to existent solutions due to its more accessible and reliable wide adaptation to types of vehicles and driving scenarios.

3 Methodology

3.1 System Architecture

The system is can be divided into two primary pipelines: the Model Deployment Pipeline, where the prepared and optimized eye classification model is set for real-time inference, and the Detection and Classification Pipeline, which will process the live video input to detect and classify the state of the eyes. These two combined pipelines will detect driver drowsiness in real time by analyzing live images coming from a webcam. Finally, the system tries to let the driver know whether he is in which state detected through a Feedback System. This can be visualized in Figure 1 as follows:

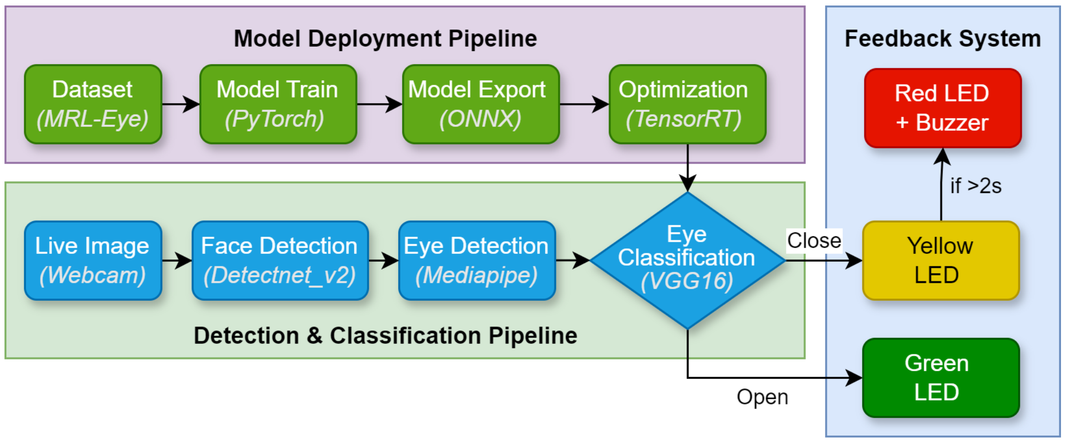

Figure 1: System Architecture for Drowsiness Detection in Real-Time.

3.1.1 Model Deployment Pipeline

Dataset (MRL-Eye):

The MRL Infrared Eye[39] Dataset comprises a total of 84,898 images from 37 subjects (33 males and 4 females). It contains images captured using three various sensors such as Intel IDS Imaging (1280x1024 resolution), RealSense RS 300 (640x480 resolution), and Aptina sensor (752x480 resolution).These images represent various conditions such as bad or good lighting and open or closed eyes some of which are shown in figure 2. The dataset also provides information about whether subjects are wearing any kind of glasses and includes reflections in the eye images as none, small, and large. This makes the dataset quite broad and appropriate for training models that need to generalize across most real-world scenarios, including drowsiness detection and eye state analysis.

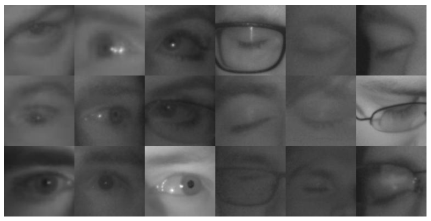
Figure 2: Sample Images from the MRL-Eye Dataset: Rows show train, validation, and test sets; columns show open (first 3) and closed (last 3) eyes.

To enable efficient model creation and assessment, the dataset was split into three separate subsets i.e. training, testing and validation. This dataset comprises two categories which are “Close-Eyes” and “Open-Eyes”. For the model development they were distributed as follows: the training subset contains a total of 50,937 images, with 25,770 images categorized as Open-Eyes and 25,167 images as Close-Eyes. The validation subset has a total of 16,980 images, out of which 8,591 images are Open-Eyes , while the remaining 8,389 images are Close-Eyes. Lastly, the testing subset also had 17,181 images, out of which 8,591 images belonged to the class of Open-Eyes and 8,390 images were Close-Eyes. Overall, the ratio of splitting the dataset was 60:20:20 for training, validation, and test subsets, respectively. The equal distribution of images within the categories into these subsets will ensure that the model is well-trained without facing a high chance of overfitting simultaneously and provides a robust mechanism for validation and testing, hence improvement in generalization and boost the model’s performance in the detection of eye states.

Model Training (PyTorch):

Model training for the classification of the state of an eye was done in PyTorch[40], upon a rich dataset of infrared images of eyes labeled both closed and open. Then, in order to enable appropriate model testing, this collection has been appropriately split between training and validation. Techniques for data augmentation had been used to improve the model’s generalizability and robustness.Techniques for data augmentation had been used to improve the model’s generalizability and robustness. Data augmentation techniques in this work included random resized cropping, where various scales and aspect ratios of the input images were used, and horizontal flipping was done to show mirrored versions of the images to the model for learning from different perspectives in recognizing the eye state. This architecture was chosen for the model since it performed better compared to others in previous tasks related to image classification. The details of model parameters and performance will be further discussed in later sections. This architecture was initialized with weights which were pre-trained from the ImageNet dataset, which allowed the model to make use of features previously learnt. This approach becomes more handy in cases when datasets are limited.

Model Export (ONNX):

Once trained and validated, the model was exported in ONNX[41]. It is a format designed to enhance interoperability between different machine learning frameworks. Thanks to ONNX, it will be easier to share and deploy models more easily in various environments. It became really helpful for leveraging the model’s capabilities in diverse environments. It required the export process to re-target the trained model in PyTorch into ONNX format, preserving both the architecture of the model and its learned parameters.

Model Optimization (TensorRT):

The ONNX model was optimized with TensorRT, an NVIDIA library designed to accomplish the best performance of inference on GPU architectures, and especially in edge devices such as the NVIDIA Jetson Orin Nano[42]. TensorRT implements the latest techniques for model optimization to be more efficient and run faster. One of the key strategies is a reduction in the numerical precision of the computation, that is, from FP32 to FP16, that can dramatically increase model throughput with a good maintenance of accuracy. The reduction in precision is extremely helpful when deep learning applications require high speed, and minor variations of precision may not affect performance overall. Along with reducing precision, TensorRT can also combine several layers of the neural network into one operation. The minimal memory access reduces computational overhead[43], thus enabling faster execution. Merging operations may also enable other forms of optimization in a model, such as utilizing shared memory or reducing storage for activations.

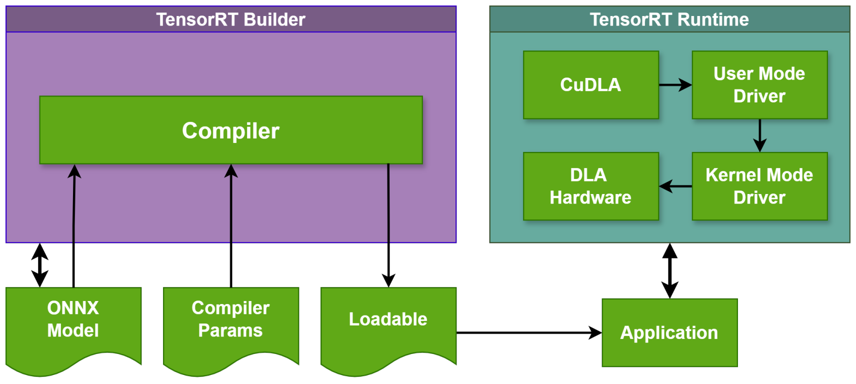

Figure 3: TensorRT Optimization Workflow for ONNX Model Inference.

3.1.2 Detection & Classification Pipeline

Live Feed (Webcam):

First, GStreamer[44] was initialized to interface it with any video capture device. Having found the device, GStreamer enumerates the supported formats and resolutions and selects one that is appropriate for image acquisition. More precisely, by instantiating a proper video source element, which is used for retrieving the video stream from this source; such parameters were set as desired resolution 1280x720 @ 30fps. After that, the video stream enters the decoder, decompressing the image data into the raw format. At last, the frames of the processed items route to an application sink, which allows real-time image retrieval and further processing or analysis within the application.

Face Detection (DetectNet v2):

To detect the driver's face in real time, the FaceDetect model was employed which is based on NVIDIA DetectNet v2[45] detector with ResNet18[46] as a feature extractor. It was chosen due to its speed and accuracy. Various face detection models, including Haar Cascade[47] and Facenet[48], were evaluated for this project. Among them, however, Facedetect had the best trade-off between real-time performance and detection accuracy on the Jetson Orin Nano. The face detection starts by extracting individual frames from the video stream captured by the Webcam, which are then passed through the Facedetect model. This model outputs bounding boxes around detected faces. In cases where it detects multiple faces, the system identifies the biggest face as that of the driver.

Eye Detection (Mediapipe Iris):

After detecting a face, the Mediapipe Iris Model[49] is used eye detection within the detected face region. Different approaches for detecting eyes were compared: Haar Cascade[50] and Dlib [51] among others. The performance of Mediapipe was better in comparison in both accuracy and real-time detection speed. The benefit of Mediapipe for eye region tracking is that it was able to work in different lighting conditions and head angles, which will be very important in a driving environment where drivers might move their heads around a lot. The Mediapipe framework detects eye landmarks in the given face bounding box from Facedetect, extracts the exact coordinates of the eye region, and feeds it to the VGG16 model for classification. Because of the highly efficient processing, Mediapipe provides this system with a high frame rate for smooth real-time drowsiness detection.

Eye Classification (VGG16):

Below is the VGG16[37] model, which has been trained as part of Model Deployment Pipeline, that the system uses to classify an eye state. When the regions of an eye are identified through Mediapipe

Iris model, the underlying mechanics of eye classification will assess the visual attributes of eye regions in every frame to determine if the eyes are closed or opened. After extracting the eye regions, each image will be fed into the model, which extracts necessary information about eyelid position, iris visibility, and many other distinctive details. Confidence scores in the model are computed for "open" and "close" classes of eye states from which the class with the max confidence score is picked as a result of the classification. This system has been embedded with a confidence threshold, just for ensuring that these classifications are reliable to avoid false positives and false negatives of any form. It performs the same classification for each and every frame in real time, hence keeping continuous track of the states of the eyes. Further, in each frame, it evaluates continuously to know for definite whether the eyes are open or closed, providing a strong mechanism in tracking eye movements and possible signs of drowsiness.

3.1.3 Feedback System

The Feedback System provides real-time alerts based on the the driver's eye classification which is shown in Figure 4. Three different colored LEDs-green, yellow, and red-controlled by GPIO[52] pins along with a buzzer use this feedback mechanism in an escalating method to alert the driver. In case both eyes have been classified as "open," it enables the green LED-the GPIO pin 11-which signals that the driver is alert and driving normally. If the driver's eyes are classified as "close" , the yellow LED (GPIO pin 13) will light up to give an early warning. If the eyes don't open within 2 seconds or more, it triggers the system to light on a red LED (GPIO pin 15) and a buzzer (GPIO pin 40) as well, which produces a much stronger alert to allow the driver to take notice of possible drowsiness. This system continuously monitor the driver and provide alerts from subtle to urgent depending on the duration of eye closure. Apart from the eye-based alert system, there will be a physical switch connected to the GPIO pin 7 in order to control of the system manually.

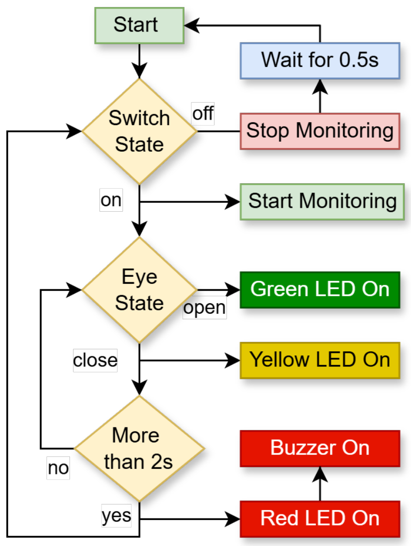

Figure 4: Feedback Alert System Flowchart.

This switch is utilized as an external input to start/stop the main application based on the state of this switch. This switch, when unpressed (HIGH), will automatically trigger the application to run the eye detection and classification system. In case the switch goes LOW-for example, pressed-it will terminate the system application for feedback alerts. Hence, this is manual control so that one will not have hassle in toggling the drowsiness detection application. It also detects any crashes of an application and its automatic restart in case the former crashes, thereby increasing reliability for continuous operation.

3.2 Model Selection and Hyperparameters

3.2.1 Model Selection

In this work, the VGG16[37] model is adopted because it has shown great efficiency in performing tasks of image classification, especially in situations where the change in visual conditions is sensitive, like the eye condition being open or closed. Figure 5 illustrates The architecture consists of 16 layers, precisely 13 convolutional and followed by fully connected layers, which provides a great efficiency in extracting features at higher levels in images. Conclusive fine details could be captured in eye regions by deep convolutional layers, relating to eyelid position, iris visibility, and other minute cues differentiating open eyes from closed ones.

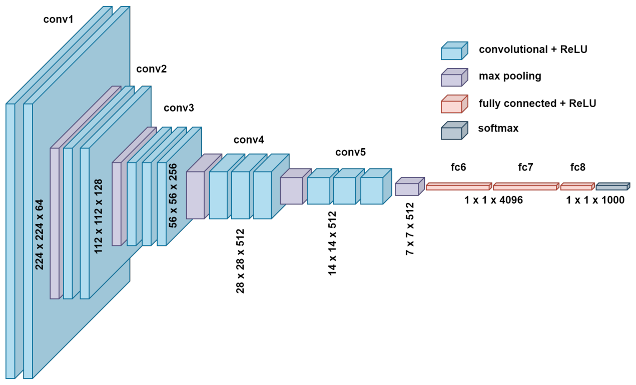

Figure 5: The VGG16 Convolutional Neural Network (CNN) Model's architecture.

Although several different models were tested in this training process, the best tradeoff between accuracy and speed was achieved by VGG16 due to its simplicity and relatively low computational overhead when compared to deeper architectures. That makes it ideal for real-time edge computing where the computational resources are usually in shortage. Additionally, weights pre-trained on huge datasets such as ImageNet enable very effective transfer learning. This diminishes the need for extensive retraining on our drowsiness dataset a great deal, while retaining high accuracy. Thus, VGG16 is a highly reliable choice for our drowsiness detection task to ensure generalization across different lighting conditions and driver behavior.

3.2.2 Hyperparameter Tuning

The training process ran for 35 epochs, using momentum of 0.9 with Stochastic Gradient Descent (SGD)[53] optimizer in order to boost convergence, and weight decay was set to 1e-4 in order to prevent overfitting. A learning rate of 0.001 was used, which allows for a good trade-off between the stability and speed of learning of the process, and batch size was set to 8 in order to assure optimal memory usage when performing the computation of gradients. After each epoch, prevalent metrics were tracked that included accuracy and loss, with the validation set. In case of overfitting, early stopping was allowed to kick in, tracking the validation accuracy. The best performing model was saved in anticipation of future tasks of inference. Efficient data loading was maintained at 2 workers, and the training progress reported every 10 batches. Pre-trained models were used so as to achieve faster learning, hence yielding better performance of the models.

3.3 Prototype Setup

NVIDIA Jetson Orin Nano is powerful machine in a small form factor build with 6-core processor of ARM Cortex-A78AE and 512 CUDA cores on Ampere Architecture. Additionally it has 8 GB of LPDDR5 DRAM-68 GB/s bandwidth that provided the required computation power for image processing tasks in real-time such as drowsiness detection. A 512GB M.2 SSD was used for storage, allowing fast reading and writing speeds to efficiently handle such a large dataset and model checkpoints. To provide a high-resolution live video input, highly important for the accuracy of eye-tracking and classification, the Logitech Brio 4K webcam was used. This further improves the eye-state detection and classification in the system through a high-resolution feed, enhancing the performance of the model in real time. The whole system was powered by a 1300mAh LiPo battery, making it portable without any need for an external power source in vehicles.

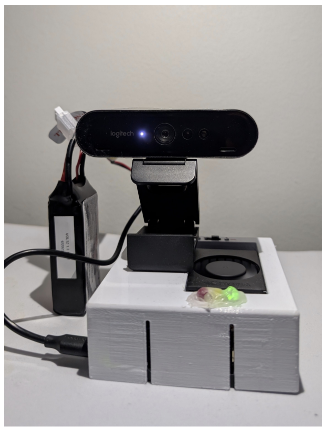

Figure 6: Prototype Image containing Jetson Orin Nano, Webcam, Lipo Battery, LED, Buzzer and switch.

The feedback mechanism, including generic LED indicators and a buzzer, could be driven directly from the internal

5V supply of the Jetson Orin Nano; hence, no external power source was needed for these peripherals. It could be controlled with a generic switch connected via the GPIO pins to the Jetson board to turn the system on or off depending on needs. Meanwhile, it would drive LEDs and a buzzer connected to the 40-pin expansion header to provide real-time feedback for the driver based on the deep learning model output. All of these were enclosed with a 3D printed case to make it portable and easy to use which is shown in Figure 6. The system was therefore able to ensure, thanks to this compact and efficient arrangement, that operation was possible in real-time-that is to say, accurate and immediate feedback could be obtained in vehicle environments.

4 Result

4.1 Training Evaluation

Figure 7 shows the training of a VGG16 model, where for each iteration, training and validation loss as well as training and validation accuracy are shown. The training loss is seen to be around 0.2 at the start while the validation loss is around 0.06, so there is scope for improvement. As the training process advances, up to 35 epochs, training loss drops smoothly down to about 0.045, while validation loss falls down almost to 0.025 with the help of improved model performance. Training accuracy starts at about 90%, and continues steadily towards 99%, representing model improvement. On validation accuracy as well, there is quite a high figure: starting from about 98.00% with a constant trend above it. In general, the graph shows that the training session was successful because losses go down and accuracies go up, therefore, showing that the model learned the training data and performs on validation data, sign of good training process with potential high performance on unseen data.

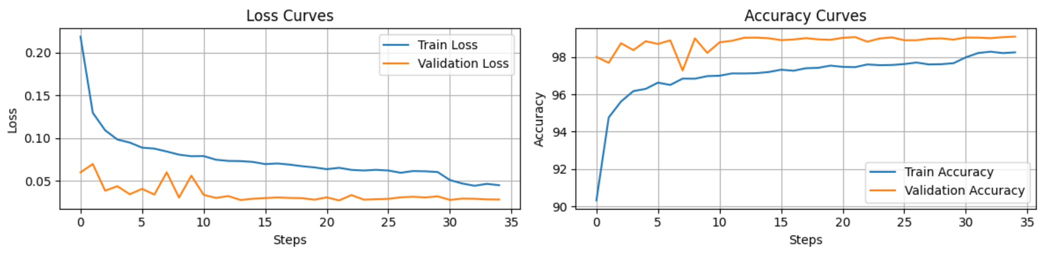

Figure 7: Loss and Accuracy Curves for VGG16 Model Training with MRL-Eye Dataset.

4.2 Performance Evaluation

We tried to focus on both accuracy and speed to evaluate the performance of the mode and VGG16 was able to balance it will. For instance, it achieved a high accuracy of 99% in classifying open and closed eyes, and precision, recall, and F1-score were also 0.99, indicating good balance in detecting drowsiness without false positives as presented in Table2. Figure 8 presents the confusion matrix for VGG16, where the total numbers of predictions are divided into 8,288 true negatives (correctly closed eyes), 8,543 true positives (correctly open eyes), 102 false positives, and 48 false negatives. The matrix yields an important quality of this model to differentiate between open and closed eyes, which is one of the critical factors in drowsiness detection. Among other significant factors for this model, real-time performance also relied on latency, which averaged at 14.4 ms per image.

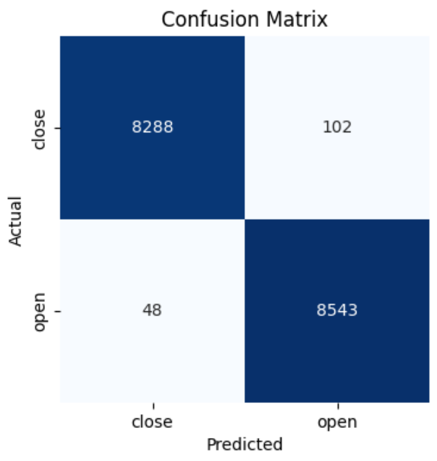

Figure 8: Confusion Matrix for VGG16 Classificatoin Model

Table 2: Performance Metrics of Various Deep Learning Models

Name	**Accuracy**	**Precision**		**Recall**		**F1 Score**		**Latency (ms)**
		Open	**Close**	**Open**	**Close**	**Open**	**Close**	
Alexnet	97%	0.94	0.94	0.99	0.99	0.97	0.97	4.5
Efficientnet_b5	98%	0.98	0.97	0.97	0.98	0.98	0.98	17.2
Googlenet	97%	0.97	0.98	0.98	0.97	0.97	0.97	2.8
mobilenet_v2	96%	0.97	0.95	0.95	0.97	0.96	0.96	2.2
Resnet_18	98%	0.98	0.98	0.98	0.98	0.98	0.98	2.1
Resnet_50	98%	0.99	0.98	0.99	0.98	0.98	0.98	5.3
Vgg 16	**99%**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**14.4**
VGG19	99%	0.99	0.99	0.99	0.99	0.99	0.99	16.2

4.2.1 Performance Comparison

Table 2 summarizes the performances of the different deep learning models for fatigue detection task that includes accuracy, precision, recall, F1 score, and latency considering the class "Open" and "Close". The ResNet18, ResNet50, EfficientNet-B5, GoogLeNet, MobileNet v2, and VGG16 have been fine-tuned to realize real-time drowsiness detection. The best performance among them was with the VGG16 model, which was able to do the classification with an accuracy of 99%, with high precision and recall for both states: "Open" and "Close", each with a value of 0.99, while latency is well balanced at 14.4 ms per image. Further optimization of this model using TensorRT on the NVIDIA Jetson Orin Nano platform succeeded in obtaining throughput of approximately 70 FPS while being energy-efficient, hence the model is suitable for edge AI applications that are power-conscious and require real-time performance. ResNet18 and ResNet50 also provided 98% accuracy, but with faster processing-2.1 ms per image-at a little less accuracy compared to VGG16. The EfficientNet-B5, while accurate, was slow, at 17.2 ms, limiting its applicability for rapid inference tasks. The current analysis identifies the VGG16 as one of the most consistent and reliable models that can be used for real-time, edge-based drowsiness detection.

5 Discussion

5.1 Model Overview

Particularly the VGG16 model is used because of its well-known accuracy and real-time processing. By identifying the eys states for both open and closed and achieves 99% accuracy hence lowering false alarms. The system has a good reaction time of 14.4 milliseconds which it can use to make the drivers alert very quickly. This system also respects the driver's privacy as it does the processing on the device rather than in the cloud. In testing, VGG16 outperformed others like ResNet18 and EfficientNet-B5 by offering a better balance of accuracy and speed. Unlike other systems that rely on cloud processing or physiological data, this system works on-device and doesn't need constant internet access, making it more versatile. Although low light and facial obstructions like sunglasses or masks can sometimes reduce accuracy, integrating infrared (IR) cameras or similar tools could help address these situations. The system offers a practical, portable, cost-effective, early-warning solution, with real driver safety. Its design helps lower accident risks by giving timely alerts and works well even in remote areas with limited connectivity. With future updates, this system could maintain high accuracy across varied conditions, making it a valuable tool for safer road travel.

5.2 Limitations

5.2.1 Environmenta Limitatons

The main disadvantages to this system of drowsiness detection are environmental factors and the situation that may make this system less accurate and reliable. Some of the limitations mentioned above include low light problems, difficulties dealing with multiple faces or occlusions, and possible effects from environmental factors such as vibration or temperature.

This system works less when there is not enough light, making the detection rather inaccurate. While driving in daytime or mild lighting, the system works commendably; practically, driving involves a lot of nighttime or driving through tunnels where visibility might be low, and the system may not be able to keep track of your eyes that well. Moreover, when the intensity of the light is poor, the camera may fail to easily recognize minor events like blinking or changes in the state of the eyes. In this respect, there may be increased false alarms or late notifications. This, in theory, could be overcome by adding an IR camera to ensure performance irrespective of the consistency of the light; it might be subjected to extra hardware modifications with associated extra costs. Presence of Several Faces or occlusions: When there are more than two people in the car, or the driver's face is half occluded by sunglasses or a mask. Second, when several faces come into view within a similar radius around the vehicle, the single camera on the system may fail to identify and track the proper driver to cause sham alarms. Moreover, face identification might be hard to achieve properly or eye tracking impeded by things like sunglasses or facial hair. Adding another layer of identification for the users or driver-specific tracking by way of seat-based detection would make things a bit more reliable in cases where there are multiple riders, ensuring alerts actually go to the correct driver.

5.2.2 Software-related limitations

Limitations in the software of this drowsiness detection system come from limitations in the VGG16 model, the TensorRT optimizations, and dataset generalizability. Although, VGG16 can recognize states of an eye quite effectively, such a deep architecture may lead to longer inference times compared to lighter models. Because VGG16 has been designed for general object classification, adaptation to drowsiness detection might not be perfect; therefore, a more customized model would probably reach similar accuracy with much less computational load. TensorRT optimizations have enhanced runtime processing efficiency on the Jetson Orin Nano. However, because of dependence upon NVIDIA and inconsistencies in hardware, performance across different devices may be jeopardized, where the differences in inference speed and reliable results are one big concern. Besides, the model's performance is highly sensitive to the diversity in the training dataset, especially the MRL Eye Dataset. Although this dataset has a number of eye images, they may not be representative and diverse enough for various eye shapes, ethnicities, light conditions, and environmental factors that will make the system generalize poorly. Without sufficient inclusions of these real-world variations, the model performance will degrade when applied to different drivers or settings, an important consideration in applications of drowsiness detection. Overcoming some of these limitations by exploring other models and increasing the diversity of the dataset can result in a more flexible, accurate, and reliable system during real-world application, along with some cross-platform optimizations.

5.2.3 Hardware-related limitations

The drowsiness detecting system is powered by the NVIDIA Jetson Orin Nano; however, there are many downsides to power consumption, portability, processing power, and scalability that may stand in the way of widespread usage. While its high performance fits well in edge computing, on the other hand, the large power requirements of the Nano make recharging very frequent, and it would be rather impractical inside vehicles that do not have a dedicated power source. Its size and dependence on external hardware make it difficult to implement in a wide variety of car models, whereby the system requires conspicuous changes that may be too expensive for consumers. The processing power is enough for real-time drowsiness detection but might become insufficient if further features or more complex models were implemented. This limits scalability when demand for multifunctional AI systems in cars is increasing. In addition, the price of Jetson Orin Nano is more expensive compared to simpler integrated solutions that could reach the mass market. Its dependence on the NVIDIA ecosystem also reduces compatibility across multiple platforms, which may further be a barrier in its adoption for most vehicle manufacturers on different hardware ecosystems. These various hardware limitations will have to be overcome, at the minimum, through research into smaller, more power-efficient AI modules, and optimization of production costs to make the system accessible and scalable for wide automotive use.

5.3 Future Works

5.3.1 Eye pattern analysis

Further analysis of eye pattern can now be done, which may further create an enhanced ability for the detection of tiredness. Advanced blink and saccade research may monitor how long a person can keep himself blinking and the movement of eyes to avoid early signs of tiredness. Pupil size changes are usually associated with fatigue, so monitoring the dilation of pupils under different lighting conditions may reveal much about alertness. More complex models can improve the accuracy of these systems utilizin training for more complex patterns in eye movements that signal drowsiness and, therefore, make them adaptable for different types of drivers.

5.3.2 Pose Estimation

Pose estimation could add much to driver monitoring by detecting early signs of fatigue beyond simply eye closure, such as tilting the head, slouching, or poor posture, which constitute very low-key signals of drowsiness. This added layer of observation could increase safety and make it possible for the system to catch not just fatigue but also distraction, such as looking away from the road frequently. Besides, pose estimation decreases the occurrence of false positives in Mult passenger environments owing to the capacity for differentiating driver-specific movements from the acts of passengers themselves. Furthermore, pose tracking could be more robust under low light conditions and occluded views by eyeglasses or sunglasses, incorporating posture and eye movement for a stronger sense of what the driver is doing. Integrating pose estimation with eye tracking increases not only the total robustness for different conditions that may appear during driving but also makes it a multirelevant and holistic toolset for drowsiness detection and therefore for traffic safety.

5.3.3 Improvements to the existing system

Key upgrades to the system will make it more effective in overcoming some limitations in its current setting. Correctly tracking the eye even in dark conditions using IR or NIR sensors would cater to low-light detection. More advanced technologies for depth sensing, such as LIDAR, will be important for distinguishing the driver from other passengers, while unique driver identifiers-like seat-based sensors or RFID-may provide enhanced focus on the driver alone. Besides that, the employment of automotive-grade vibration-resistant hardware and adding cooling solutions for the Jetson Orin Nano would further improve stability over temperature variations. Model tuning onto lightweight architectures such as MobileNetV3 would provide reduced power consumption without loss of accuracy, while adaptive learning algorithms may allow personalized drowsiness detection based on patterns driven by individual drivers, hence reduced false positives. These would further enhance system reliability, hence much more adaptable and fitted for real driving environments in the real world.

5.3.4 System expansion

It could even add a far greater range of functionalities to the system that would significantly enhance driving safety and adaptability. The most critical improvement would be the integration with in-vehicle safety systems, whereby such a system could provide an in-car visual and audible warning or, in more advanced models, even slow down or stop the car upon the detection of drowsiness. Integrated with vehicle controls, it would become possible to adjust speed or make lane position adjustments in real time and, therefore, lower the possibility of driving while drowsy. It would further send alerts to other devices connected to it, such as a smartphone or to fleet management devices, so that supervisors can also be constantly aware of what is taking place. Other functionalities would include data logging and analysis for identifying trends with the aim of providing personalized recommendations on rest or route adjustments, and hence improve the general driving conditions, especially for long-haul drivers. Further, this data will help fleet managers in optimizing drivers' schedules and personalize the detection of fatigue over time. This would integrate into other systems, such as lane-keep assistance, to provide a suite of driver assistance with coordinated alerts and corrective measures. These are features that turn the system into an active, integrated safety platform with long-term driver wellness.

5.3.5 Enhancements model performance

Improvement in model performance could make the model very accurate, flexible, and responsive to different driving conditions and user profiles. Increasingly diverse training datasets regarding eye shapes, facial features, and ethnic backgrounds would improve generalization and reduce bias, thereby improving accuracy across users. Real-life detection of drowsiness would be further enhanced with data relating to a variety of lightings, head positions, and environments. The incorporation of drivers' real-world data from ages with various accessories, like glasses and hats, will add robustness across appearances. Efficient deep learning model evaluations, including but not limited to MobileNet and EfficientNet, would optimize performance at the edge in real time, tuning a good balance of accuracy with less computation power. ResNet, which uses residual connections, might improve feature extraction of subtle fatigue cues without heavy processing.This will also reduce model size for edge devices, including Jetson Orin Nano, by using model optimization techniques like quantization, knowledge distillation and pruning. Detection thresholds may also be optimized for each driver to decrease false positives using adaptive learning algorithms, enabling customized responses.

5.3.6 Exploring alternative technologies

Adaptation to various technologies could leverage the system to further its key accessibility or functionality for real-world applications. Low-power machine learning models, such as those optimized via TinyML or compact versions of MobileNet, might let the system run on microcontrollers and normal car electronics, which further allows low-cost and scalable implementation. The incorporation of dedicated AI edge chips, such as Intel's Movidius Myriad X or Google's Coral Edge TPU would therefore significantly enhance drowsiness detection while keeping the power consumption low-great assurance that accuracy and latency are high with minimal vehicle computing resources used. Sensor fusion, to be derived from input by accelerometers, LIDAR, or thermal cameras, would build reliability in the most challenging conditions-low light or obstructions. Second, it might explore cloud-edge hybrid settings for continuous connectivity for model updates and infrequent data synchronization with the cloud for fleet management activities. These will further make the system more energy-efficient, adaptable, and scalable in a wide deployment basis of automotive applications.

6 Conclusion

Based on edge computing, this project successfully shows a high-tech way to find drivers who are tired. The NVIDIA Jetson Orin Nano and VGG16 model is used to find drivers who are sleepy very accurately and in real time. The system is very good at telling the difference between eye states (99% accuracy), so there aren't many fake positives or negatives. It also has a reliable alert system that makes sure drivers get feedback right away. At 14.4 milliseconds per frame, this fast reaction time is very important for avoiding accidents that could happen because of delays caused by driver fatigue. This makes the system a useful tool for improving driver safety.

The technology also makes things safer and more stable by handling data on the device itself instead of in the cloud. This helps a lot when you're not close to the internet or can't connect well. Edge computing not only facilitates continuous real-time monitoring but also mitigates the security risks associated with cloud reliance, making the system adaptable for varied driving environments. While the system's robustness may slightly reduce in low-light or obstructed views, potential enhancements like infrared (IR) cameras or LIDAR could further extend its effectiveness.

To improve the system's flexibility to different driving situations and driver characteristics, future versions should diversify the training dataset and incorporate technologies like adaptive learning and posture estimation. This sleepiness detection system shows the wider potential of edge-based safety solutions in the car industry and bridges high AI capabilities with practical application, making roads safer.

7 Contribution Distribution

Akash Bappy: System Architecture Design, Face detection (Detectnetv2), Eye Detection (Mediapipe), Model Training and Tuning (Vgg16,Vgg19), Model Conversion and Optimization, Report Drafting (Methodology, Results), Report Finalization

Mitun Paul: Feedback Mechanism Implementation, Face detection (Haar Cascade), Eye Detection (Dlib), Model Training and Tuning (Resnet18,Resnet50), Report Drafting (Related Works)

Ullas Chowdhury: Prototype Hardware Assembly, Face detection (Facenet), Model Training and Tuning (Alexnet,Efficientnet_b5), Report Drafting (Discussion, Conclusion)

Tanvir Shuvo: Eye Detection (Haar Cascade), Model Training and Tuning (Googlenet,mobilenet_v2), Report Drafting (Introduction)

References

[1] Jeremy D. Slater. “A definition of drowsiness: One purpose for sleep?” In: *Medical Hypotheses* 71.5 (2008), pp. 641–644. ISSN: 0306-9877. DOI: `https://doi.org/10.1016/j.mehy.2008.05.035`. URL: `https://www.sciencedirect.com/science/article/pii/S0306987708002946`.

[2] Arun Sahayadhas, Kenneth Sundaraj, and Murugappan Murugappan. “Detecting Driver Drowsiness Based on Sensors: A Review”. In: *Sensors* 12.12 (2012), pp. 16937–16953. ISSN: 1424-8220. DOI: `10.3390/s121216937`. URL: `https://www.mdpi.com/1424-8220/12/12/16937`.

[3] Brian C. Tefft. *The Prevalence and Impact of Drowsy Driving*. Technical Report. Washington, D.C.: AAA Foundation for Traffic Safety, 2010. URL: `https://aaafoundation.org/prevalence-impact-drowsy-driving/`.

[4] Brian Tefft. “Prevalence of motor vehicle crashes involving drowsy drivers, United States, 1999-2008”. In: *Accident; analysis and prevention* 45 (Mar. 2012), pp. 180–6. DOI: `10.1016/j.aap.2011.05.028`.

[5] Uzair Ghole et al. “Drowsiness detection and monitoring system”. In: *ITM Web of Conferences*. Vol. 32. EDP Sciences. 2020, p. 03045.

[6] Prakhar Pratap et al. “DROWSINESS DETECTION SYSTEM USING MACHINE LEARNING”. In: *NIET Journal of Engineering and Technology* (2018). URL: `https://api.semanticscholar.org/CorpusID:260588015`.

[7] H. Ueno, M. Kaneda, and M. Tsukino. “Development of drowsiness detection system”. In: *Proceedings of VNIS’94 - 1994 Vehicle Navigation and Information Systems Conference*. 1994, pp. 15–20. DOI: `10.1109/VNIS.1994.396873`.

[8] Antoine Picot, Sylvie Charbonnier, and Alice Caplier. “On-Line Detection of Drowsiness Using Brain and Visual Information”. In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 42.3 (2012), pp. 764–775. DOI: `10.1109/TSMCA.2011.2164242`.

[9] Yaman Albadawi, Maen Takruri, and Mohammed Awad. “A Review of Recent Developments in Driver Drowsiness Detection Systems”. In: *Sensors* 22.5 (2022). ISSN: 1424-8220. DOI: `10.3390/s22052069`. URL: `https://www.mdpi.com/1424-8220/22/5/2069`.

[10] Vias institute. *Finland – ESRA3 Country Fact Sheet*. ESRA3 survey (E-Survey of Road users’ Attitudes) [Fact sheet]. Version 2 (01/2024). 2023. URL: `https://www.esranet.eu/storage/minisites/esra2023countryfactsheetfinland.pdf`.

[11] Y. Katyal, S. Alur, and S. Dwivedi. “Safe driving by detecting lane discipline and driver drowsiness”. In: *Proc. IEEE Int. Conf. Adv. Commun., Control Comput. Technol.* May 2014, pp. 1008–1012.

[12] G. Zhenhai et al. “Driver drowsiness detection based on time series analysis of steering wheel angular velocity”. In: *Proc. 9th Int. Conf. Measuring Technol. Mechatron. Automat. (ICMTMA)*. Jan. 2017, pp. 99–101.

[13] Z. Li et al. “Online detection of driver fatigue using steering wheel angles for real driving conditions”. In: *Sensors* 17.3 (Mar. 2017), p. 495.

[14] Z. Li et al. “Automatic detection of driver fatigue using driving operation information for transportation safety”. In: *Sensors* 17.6 (May 2017), p. 1212.

[15] H. S. AlZu’bi, W. Al-Nuaimy, and N. S. Al-Zubi. “EEG-based driver fatigue detection”. In: *Proc. 6th Int. Conf. Develop. Syst. Eng. (DESE)*. Dec. 2013, pp. 111–114.

[16] G. Li and W.-Y. Chung. “Detection of driver drowsiness using wavelet analysis of heart rate variability and a support vector machine classifier”. In: *Sensors* 13.12 (Dec. 2013), pp. 16494–16511.

[17] H. A. Rahim, A. Dalimi, and H. Jaafar. “Detecting drowsy driver using pulse sensor”. In: *J. Technol.* 73.3 (Mar. 2015), pp. 5–8.

[18] L. B. Leng, L. B. Giin, and W.-Y. Chung. “Wearable driver drowsiness detection system based on biomedical and motion sensors”. In: *Proc. IEEE Sensors*. Nov. 2015, pp. 1–4.

[19] L. Barr, S. Popkin, and H. Howarth. *An evaluation of emerging driver fatigue detection measures and technologies*. Tech. rep. FMCSA-RRR-09-005. Washington, DC, USA: Federal Motor Carrier Safety Administration, 2009.

[20] W.-B. Horng et al. “Driver fatigue detection based on eye tracking and dynamic template matching”. In: *Proc. IEEE Int. Conf. Netw., Sens. Control*. Vol. 1. Mar. 2004, pp. 7–12.

[21] M. Saradadevi and P. Bajaj. “Driver fatigue detection using mouth and yawning analysis”. In: *Int. J. Comput. Sci. Netw. Secur.* 8.6 (June 2008), pp. 183–188.

[22] M. A. Assari and M. Rahmati. “Driver drowsiness detection using face expression recognition”. In: *Proc. IEEE Int. Conf. Signal Image Process. Appl. (ICSIPA)*. Nov. 2011, pp. 337–341.

[23] C. Yan et al. “Video-based classification of driving behavior using a hierarchical classification system with multiple features”. In: *Int. J. Pattern Recognit. Artif. Intell.* 30.5 (2016), Art. no. 1650010.

[24] I. Teyeb et al. “A novel approach for drowsy driver detection using head posture estimation and eyes recognition system based on wavelet network”. In: *Proc. 5th Int. Conf. Inf., Intell., Syst. Appl. (IISA)*. July 2014, pp. 379–384.

[25] N. Kuamr, N. C. Barwar, and N. Kuamr. “Analysis of real time driver fatigue detection based on eye and yawning”. In: *Int. J. Comput. Sci. Inf. Technol.* 5.6 (2014), pp. 7821–7826.

[26] R. Ahmad and J. N. Borole. “Drowsy driver identification using eye blink detection”. In: *Int. J. Comput. Sci. Inf. Technol.* 6.1 (Jan. 2015), pp. 270–274.

[27] T. P. Nguyen, M. T. Chew, and S. Demidenko. “Eye tracking system to detect driver drowsiness”. In: *Proc. 6th Int. Conf. Automat., Robot. Appl. (ICARA)*. Feb. 2015, pp. 472–477.

[28] A. Rahman, M. Sirshar, and A. Khan. “Real time drowsiness detection using eye blink monitoring”. In: *Proc. Nat. Softw. Eng. Conf. (NSEC)*. Dec. 2015, pp. 1–7.

[29] Y. Albadawi, M. Takruri, and M. Awad. “A Review of Recent Developments in Driver Drowsiness Detection Systems”. In: *Sensors* 22.5 (2022), p. 2069. DOI: `https://doi.org/10.3390/s22052069`.

[30] Q. Abbas and A. Alsheddy. “Driver Fatigue Detection Systems Using Multi-Sensors, Smartphone, and Cloud-Based Computing Platforms: A Comparative Analysis”. In: *Sensors* 21.1 (Dec. 2020), p. 56. DOI: `10.3390/s21010056`.

[31] S. Díaz-Santos et al. “Driver Identification and Detection of Drowsiness while Driving”. In: *Applied Sciences* 14.6 (2024), p. 2603. DOI: `https://doi.org/10.3390/app14062603`.

[32] NVIDIA. *NVIDIA Jetson Orin Nano Sets New Standard for Entry-Level Edge AI and Robotics with 80x Performance Leap*. [Online; accessed 30-Oct-2024]. 2023. URL: `https://nvidianews.nvidia.com/news/nvidia-jetson-orin-nano-sets-new-standard-for-entry-level-edge-ai-and-robotics-with-80x-performance-leap`.

[33] NVIDIA. *Solving Entry-Level Edge AI Challenges with NVIDIA Jetson Orin Nano*. [Online; accessed 30-Oct-2024]. 2023. URL: `https://developer.nvidia.com/blog/solving-entry-level-edge-ai-challenges-with-nvidia-jetson-orin-nano/`.

[34] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: `1512.03385 [cs.CV]`. URL: `https://arxiv.org/abs/1512.03385`.

[35] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. “ImageNet classification with deep convolutional neural networks”. In: *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*. NIPS’12. Lake Tahoe, Nevada: Curran Associates Inc., 2012, pp. 1097–1105.

[36] Andrew G. Howard et al. “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications”. In: *CoRR* abs/1704.04861 (2017). arXiv: `1704.04861`. URL: `http://arxiv.org/abs/1704.04861`.

[37] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: `1409.1556 [cs.CV]`. URL: `https://arxiv.org/abs/1409.1556`.

[38] Biying Fu et al. "A Survey on Drowsiness Detection – Modern Applications and Methods". In: (2024). [Online; accessed 30-Oct-2024]. URL: `https://arxiv.org/html/2408.12990v1#S10`.

[39] Radovan Fusek. "Pupil Localization Using Geodesic Distance". In: *Advances in Visual Computing*. Ed. by George Bebis et al. Cham: Springer International Publishing, 2018, pp. 433–444. ISBN: 978-3-030-03801-4.

[40] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. URL: `https://proceedings.neurips.cc/paper_files/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf`.

[41] Ayush Shridhar, Phil Tomson, and Mike Innes. "Interoperating Deep Learning models with ONNX.jl". In: *Proceedings of the JuliaCon Conferences* 1.1 (2020), p. 59. DOI: `10.21105/jcon.00059`. URL: `https://doi.org/10.21105/jcon.00059`.

[42] Eunjin Jeong, Jangryul Kim, and Soonhoi Ha. "TensorRT-Based Framework and Optimization Methodology for Deep Learning Inference on Jetson Boards". In: *ACM Trans. Embed. Comput. Syst.* 21.5 (Oct. 2022). ISSN: 1539-9087. DOI: `10.1145/3508391`. URL: `https://doi.org/10.1145/3508391`.

[43] Yuxiao Zhou and Kecheng Yang. "Exploring TensorRT to Improve Real-Time Inference for Deep Learning". In: *2022 IEEE 24th Int Conf on High Performance Computing Communications; 8th Int Conf on Data Science Systems; 20th Int Conf on Smart City; 8th Int Conf on Dependability in Sensor, Cloud Big Data Systems Application (HPCC/DSS/SmartCity/DependSys)*. 2022, pp. 2011–2018. DOI: `10.1109/HPCC-DSS-SmartCity-DependSys57074.2022.00299`.

[44] Yongpeng Zhang and Frank Mueller. "GStream: A General-Purpose Data Streaming Framework on GPU Clusters". In: *2011 International Conference on Parallel Processing*. 2011, pp. 245–254. DOI: `10.1109/ICPP.2011.22`.

[45] Ho Chuen Kam et al. "Effective real-time face mask detection on NVIDIA edge devices". In: *HKIE Transactions* 29.2 (2022), pp. 120–128. DOI: `10.33430/v29n2thie-2021-0029`.

[46] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: `10.1109/CVPR.2016.90`.

[47] Li Cuimei et al. "Human face detection algorithm via Haar cascade classifier combined with three additional classifiers". In: *2017 13th IEEE International Conference on Electronic Measurement Instruments (ICEMI)*. 2017, pp. 483–487. DOI: `10.1109/ICEMI.2017.8265863`.

[48] Florian Schroff, Dmitry Kalenichenko, and James Philbin. "FaceNet: A unified embedding for face recognition and clustering". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2015. DOI: `10.1109/cvpr.2015.7298682`. URL: `http://dx.doi.org/10.1109/CVPR.2015.7298682`.

[49] Artsiom Ablavatski et al. *Real-time Pupil Tracking from Monocular Video for Digital Puppetry*. 2020. arXiv: `2006.11341 [cs.CV]`. URL: `https://arxiv.org/abs/2006.11341`.

[50] Norma Latif Fitriyani, Chuan-Kai Yang, and Muhammad Syafrudin. "Real-time eye state detection system using haar cascade classifier and circular hough transform". In: *2016 IEEE 5th Global Conference on Consumer Electronics*. 2016, pp. 1–3. DOI: `10.1109/GCCE.2016.7800424`.

[51] Davis E. King. "Dlib-ml: A Machine Learning Toolkit". In: *J. Mach. Learn. Res.* 10 (Dec. 2009), pp. 1755–1758. ISSN: 1532-4435.

[52] Agus Kurniawan. "NVIDIA Jetson Nano I/O Programming". In: *IoT Projects with NVIDIA Jetson Nano: AI-Enabled Internet of Things Projects for Beginners*. Berkeley, CA: Apress, 2021, pp. 63–83. ISBN: 978-1-4842-6452-2. DOI: `10.1007/978-1-4842-6452-2_5`. URL: `https://doi.org/10.1007/978-1-4842-6452-2_5`.

[53] Nikhil Ketkar. “Stochastic Gradient Descent”. In: *Deep Learning with Python: A Hands-on Introduction*. Berkeley, CA: Apress, 2017, pp. 113–132. ISBN: 978-1-4842-2766-4. DOI: `10.1007/978-1-4842-2766-4_8`. URL: `https://doi.org/10.1007/978-1-4842-2766-4_8`.