B. Tech IoT Programming

Project Report

On

# Smart Drive: Real-Time Drowsiness Detection and Alert System Using Vgg DenseNet Hybrid for Driver Safety

by

**Group Leader:**

**Raghvi Gupta (123102177) 123102177@nitkkr.ac.in 9868438110**

**Other Team Members:**

| | |
|---|---|
| Kumar Priyam | (123102180) 123102180@nitkkr.ac.in 9717885693 |
| Bhavit Mittal | (123102183) 123102183@nitkkr.ac.in 9116939936 |
| Priyanka Garhwal | (123102178) 123102178@nitkkr.ac.in 6260786803 |
| Parth Sharma | (123102184) 123102184@nitkkr.ac.in 9205921533 |

**Branch: CS C**

**Subject: IoT Programming (CSPC 209)**

## *Abstract*

This report presents a solution on the increasing accidents and how they can be prevented. The solution includes development of a system for drivers which detects whether they are sleepy or not thus preventing a lot of accidents from happening and ensure Driver Safety.

Our system uses a hybrid of two pretrained deep learning model (VGG16 and DenseNet) based on Convulation Neural Networks(CNN) which has been trained and validated on our dataset. 80% of the dataset is used to train the model while the remaining 20% is used to validate it. This model will give the result on whether the driver is sleepy or not and thereby using ThingSpeak and Wokwi simulator to trigger a buzzer using ESP32.

We used combination of VGG16 and DenseNet as it proved to generate better results than VGG16 and DenseNet individually, we also used some other techniques to improve model efficiency such as Early Stopping, increasing number of epochs, Unfreezing Some layers for better results, using a learning rate scheduler etc which gave the average training accuracy of 97.19% and average validation accuracy of 99.36%.

**Important Keywords**- Deep Learning, CNN, Drowsy Detection, Driver Safety, VGG16, DenseNet, ESP32, ThingSpeak

## 1. Introduction

Falling asleep or drowsy while driving is one of the main causes of road accidents that take place in India. 37% of road accidents in India are caused by driver fatigue and drowsiness according to the Ministry of Road Transport and Highways (MoRTH) Report, 2021. The Indian Foundation of Transport Research and Training (IFTRT) reports 50% of highway accidents are due to fatigue-related crashes involving long distance truck drivers. 11% of global road deaths occur in India, despite the country having only 1% of the world's vehicles as highlighted by World Health Organization (WHO), Global Status Report on Road Safety, 2018. Moreover, economic loss also occurs due to this. The economic loss due to road accidents, including drowsy driving, is estimated at 3-5% of India's GDP as per World Bank Report on Road Safety in India, 2019.

In this fast-paced world, where everyone is in a hurry and suffer from fatigue and spend their maximum times behind the wheel leading to slow reaction times and improper decision making an urgent need of a mechanism can be felt which monitors the driver's fatigue and drowsy states and correspondingly triggers warnings in order to reduce and prevent accidents. This project is this needed and can be viewed as an intelligent solution to the problem stated above.

This project proposes a solution to this problem by building an IoT-based driver fatigue detection system that leverages Convolutional Neural Networks (CNNs) to detect signs of drowsiness and trigger an alarm, ensuring that the driver is alerted before a potential accident occurs. The system combines real-time facial analysis with smart IoT sensors to create a robust, reliable, and scalable solution to a global problem.

This project makes use of Computer Vision techniques, particularly CNNs, to monitor the driver's facial expressions and head posture to detect signs of fatigue or drowsiness such as eye closure, yawning, and head nodding. The CNN model is trained on a dataset exhibiting varying degrees of drowsiness.

Currently, we are focusing on the model and later the model can be integrated with the following architecture:

1. Camera Module: A small infrared (IR) or standard camera will be placed on the vehicle's dashboard that will capture real-time video of the driver's face.
2. Processing Unit (Edge Device): A Raspberry Pi or similar microcontroller that will process the video feed and run the trained CNN model to analyse the driver's drowsiness level.
3. Convolutional Neural Network (CNN): The CNN model is trained to detect drowsiness patterns based on facial features, including eye closure, yawning, and head tilts.
4. Alert Mechanism: When drowsiness is detected, an alert (vibration) will be triggered using the microcontroller.

5. Cloud Storage/Analytics: For long-term data collection, the system can upload fatigue detection logs to the cloud, enabling fleet managers to monitor driver behaviour over time.

The proposed solution will help in reducing the number of accidents caused by driver fatigue and contributing to overall road safety.

## Unique Functionality of the project:

Following points outline the unique functionality of the project:

1. Custom Dataset Creation: A tailored dataset was collected featuring 51 videos of male and female students which allowed us to extract 33,229 pictures in total simulating realistic driving conditions and eye movements (blinking, eye opening/closing, slight head movement).

2. Realistic Scenarios: The dataset focuses on natural behaviors during driving, such as subtle eye behavior and body language, making the detection system more accurate in real-world conditions.

3. Hybrid CNN Model: The project combines two pretrained Convolutional Neural Network (CNN), VGG16 and DenseNet, enhancing the detection process by leveraging transfer learning for better feature extraction and analysis. Their hybrid produces a better accuracy than both of them separately.

| Average | VGG 16 | DenseNet | VGG 16_&_DenseNet |
|---|---|---|---|
| Training Accuracy | 87.40% | 81.68% | 97.19% |
| Validation Accuracy | 92.24% | 91.07% | 99.36% |

4. Unfreezing some layers: In the hybrid of the two models, we unfroze last 4 layers of the model which allowed them to be fine-tuned according to our task and dataset. This was one of the factors responsible for the increase in accuracy.

5. Using a learning rate schedular: Learning rate schedular dynamically adjusts the learning rate during training, which allows for better fine tuning and convergence of the model. Learning rate basically controls how much the model's weights are updated during each iteration. This also contributed in the betterment of the model's results.

## Related Work in Project:

Our project focuses on detecting driver drowsiness; multiple projects have tried to detect driver drowsiness with various methods.

Some systems analyze facial expressions, such as whether the eyes are open or closed, to determine drowsiness. Models like **VGG16**, **VGG19**, and a **novel 4D model** are used for detecting drowsiness. When the eyes remain closed for a long time, the system triggers an alert. Some systems also check for multiple signs of drowsiness, such as blinking frequency, yawning, and head tilting, to improve the model's efficiency. The system is equipped with IoT devices that emit a warning message when drowsiness is detected, and an audible alert is produced.

Some systems use a CNN-RNN hybrid model and a custom CNN model to detect drowsiness, and the DLIB library is employed for detecting facial key points and calculating the **MAR (Mouth Aspect Ratio)** for yawning detection. Based on the MAR, the system can detect drowsiness. Other systems use the concept of **Eye Closure Detection**, which calculates the **EAR (Eye Aspect Ratio)** to determine drowsiness by monitoring the rate of eye closure. When drowsiness is detected, the system triggers a buzzer to alert the driver and sends a notification to the owner. Physical sensors are not required; if the EAR drops below a threshold of 0.25, indicating that the driver’s eyes remain closed for a long time, the system alerts the driver with a buzzer. They use a **Raspberry Pi 3 Model**, **a Pi Camera Module**, and a **buzzer**.

Some systems utilize **Modified SVM (Support Vector Machines)**, while other models include **KNN**, **Naive Bayes**, **Decision Trees**, **Random Forests**, and **Logistic Regression**. The sensors used include a camera (webcam), and facial landmark detection is carried out using **DLIB** and **OpenCV**.

Below are some reports related to our project and their details are discussed:

1) **Jahan et al [1]**
2) **Phan et al [2]**

3) **Aytekin et al [3]**
4) **Ahmed et al [4]**
5) **Majeed et al [5]**
6) **Srilakshmi et al [6]**
7) **Florez et al [7]**
8) **Satish et al [8]**
9) **Hossain et al [9]**

| **Author** | **Link of published work** | **Year** | **Features** | **Model/ Sensor used** | **Results** | **Drawback in their Project** |
|---|---|---|---|---|---|---|
| | **(Research Paper/ Patent)** | | | | | |
| Jahan et al. | https://www.mdpi.com/2079-9292/12/1/235 | 2023 | 1.Whether eyes are open or closed drowsiness checking in real-time detection.<br>2.For comparing efficiencies of models, used multiple deep learning models.<br>3. when eyes closed for long time alert triggers | **VGG16, VGG19**, and **novel 4D model** | 4D model achieved an accuracy of 97.53% that is more than VGG16 and VGG19.<br>VGG16 - 95.93%, VGG19 - 95.03%<br>• Real-time feasibility: Implemented on a webcam, provide real time alerts | 1.Real-time usage can take a lot of computing power.<br><br>2. Their dataset does not represent real life scenarios as it only contains cropped images of left and right eyes. |

| | | | | | | |
|---|---|---|---|---|---|---|
| Phan et al. | https://www.sciencedirect.com/science/article/abs/pii/S2542660523000288 | 2023 | 1.Multiple Drowsiness Signs: The system checks various signs of drowsiness such as: Blinking frequency Yawning Head tilting etc. 2. IOT Integration: The system is equipped with an IoT module that: Emits warning messages, when drowsiness is detected, it produces audible alerts. 3.Adaptability: This model performs good in different driving conditions and environments. | Models used: **LSTM (Long Short-Term Memory), VGG16, InceptionV3and DenseNet**. Sensor used: **Camera,microphone,IOT device**. | 1.This approach achieves a Very good accuracy up to 98% in detecting driver drowsiness. 2.The system was applicable for various conditions, including scenarios where drivers wore masks and glasses. | 1.Environmental Factors: Surrounding light 2. Computational Resources :Most deep learning models require lot of processing power, especially on embedded systems. |
| Aytekin et al. | intapi.sciendo.com/pdf/10.2478/acss-2022-0009 | 2022 | 1. Classification of Eye and Yawn Status: The system observes whether the driver's eyes are open or closed and whether they are yawning or not. 2.Deep Learning Model: The VGG16 model by using transfer learning detects driver drowsiness. 3.The system observe that there is | Model used: **VGG16 model** (pre-trained model). Sensor used :**In- vehicle Camera**. | This system, using a VGG16 model**,** achieved 91% accuracy in detecting driver drowsiness based on eye | Dataset Limitation**:** Because of limited dataset, the possibility of overtraining of data increases. |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | no need for any physical sensors to the driver's body for taking real time images. | | close or not and yawning or not . | |
| Ahmed et al. | A Deep-Learning Approach to Driver Drowsiness Detection (mdpi.com) | 2023 | 1.The system uses Convolutional Neural Networks (CNN) and VGG16 model, to detect driver drowsiness.<br>2.It categorizes facial expressions related to drowsiness (open eyes, closed eyes, yawning, and no yawning) using a dataset of 2900 images.<br>3.The model monitor the changes in facial expressions and eyeball movements. | **CNN Model**: Custom-built CNN model for driver drowsiness detection.<br><br>**VGG16 Model** (pre-trained model) | 1. The CNN model perform better with an efficiency of 97%, VGG16 had a low efficiency 74% | 1. Used extremely small dataset (2900 images), that might lead to overfitting which leads to poor efficiency. |
| Majeed et al. | Detection of Drowsiness among Drivers Using Novel Deep Convolutional Neural Network | 2023 | 1. To detect driver's drowsiness based on facial expression like yawning and eye movement, it uses custom CNN and a hybrid CNN-RNN model. | 1. **CNN-1**: A convolution neural network is designed to capture physical features. | 1. The custom CNN-1 model achieved a 96.69% accuracy and CNN-2 | 1. It struggles with real world scenarios, where driver's facial orientation |

| | | | | | | |
|---|---|---|---|---|---|---|
| | Model (mdpi.com) | | 2. The DLIB library is used for detecting facial key points and also calculate MAR for yawing detection. | 2. **CNN-2:** CNN with more complex layer and filter 3.**Hybdid CNN-RNN** | model achieved 96.34%,while the hybrid CNN-RNN model achieve 95.24% | might not be continuously in front of camera. |
| Srilakshmi et al. | https://ieeexplore.ieee.org/abstract/document/10104942 | 2023 | Two primary features:<br>**1.** Eye Aspect Ratio (EAR) to detect eye closure.<br>**2.** Mouth Aspect Ratio (MAR)to detect yawning.<br>3.It uses different method for identifying which one give high accuracy. | 1. Modified **SVM (Support Vector Machines)** Other models: **KNN, Naive Bayes, Decision Trees, Random Forests, and Logistic Regression**. Sensor used: **Camera**(web cam),facial landmark detection using **DLIB** and **open cv** | Modified SVM: 95.20% accuracy (best result) Random Forest gives 93.1% accuracy. Logistic Regression gives 93.49% accuracy. Decision Trees gives 90.0% accuracy.<br>K-Nearest Neighbors (KNN) gives 80.82% accuracy<br>Bayes gives 69.18% accuracy | 1.No real-world testing has been done. 2. Accuracy issues: Single-sensor systems generally give inaccurate results. |
| Florez et al. | https://www.mdpi.com/2076- | 2023 | 1.Eye region detection in this it checks whether eyes are open or closed ,it | **ResNet50V2** **VGG16** **InceptionV3** | ResNet50V2 -based CNN model gives accuracy. of | 1. System needs to correct head movement |

| | | | | | | |
|---|---|---|---|---|---|---|
| | 3417/13/13/7849 | | uses ROI extraction (Region of Interest) was used to check the driver's state is drowsy or not. | Sensor: **Camera** | 99.71% . VGG16-based CNN give accuracy of 99.41%, and the InceptionV3-based CNN gives accuracy of 99.31% | because it creates problems in ROI extraction. |
| Satish et al. | https://ieeexplore.ieee.org/abstract/document/9182237 | 2020 | 1.Eye blinking detection using camera-based input, using the Histogram of Oriented Gradients (HOG) algorithm.<br>2. Hand pressure detection using a load cell sensor integrated with an Arduino module.<br>3. The system combines the data from both sensors to detect drowsiness and trigger an alert. | Model used 1.**HOG** (Histogram of Oriented Gradients) for facial and eye movement detection.<br>2. A simple **Arduino-based microcontroller** for hand pressure.<br>Sensor Used: **Camera and Load cell Sensor**. | 1.This system give 96 % of accuracy for eye blinking and hand pressure sensor data and also gives 4 % of false detection rate.<br>2. Hand pressure detection: using a sensor integrated with | 1. It only provides good results when Live video with pressure sensor and eyeblink data is provided otherwise the percentage of false detection is pretty high.<br>2.Limited accuracy without combined data. |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | Arduino Module.<br>3. At the end system combine the data from both sensor to detect the drowsiness | |
| Hossain et al. | https://ieeexplore.ieee.org/abstract/document/8550026 | 2018 | 1. Eye Closure Detection: It calculate the EAR (Eye Aspect Ratio) to determine drowsiness by monitoring the rate of eye closure.<br>2.Alert System : When drowsiness is detected Buzzer alerts the driver and notification is sent to owner<br>3. No physical sensors are required | **Raspberry Pi 3 Model B, Pi Camera Module** (to capture facial Landmark and calculate EAR)<br>Sensors Used: **Pi Camera Module, Buzzer** . | 1. The system detect the drowsiness using EAR, if EAR below threshold 2.5 it means drivers eyes remain closed for long time, the system alert the driver by buzzer.<br>2. The system is determine the difference between | 1. Effectiveness during night driving is less because system cannot detect facial landmarks and calculate EAR .<br>2. The accuracy of system can be affected by environmental condition such as poor lightning and camera position. |

|  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|
|  |  |  |  |  | drowsy eyes and normal blink also. |  |

## Drawbacks in these existing projects:

**Drawbacks in these existing projects:**

**1. Overfitting and overtraining of data:** Many models, like VGG16 and CNN- custom built face a problem of overfitting of data due to small datasets, which limits their performance.

**2. Poor Dataset:** Some of these projects use datasets that do not represent real-life scenarios very well hence the performance of the model plummets.

**3. Techniques:** A wide range of projects has not used techniques that can enhance the efficiency of model considerably. These techniques include – using learning rate scheduler, unfreezing some layers of CNN model etc.

**4. Lack of Real-World Implementation and Testing**: Most of them focus on training model using dataset rather than implementing in real world driving condition or simulator, this arises a problem of how good these systems would perform under real-world complexities, such as different traffic conditions.

## 2. Project

**The setup and technical details for the project is described as follows:**

**Hardware: (to be used in actual implementation in the vehicle)**

- **ESP32 Microcontroller**
    - Type- Wi-Fi-enabled Microcontroller

        - Purpose- It will act as a processing unit for gathering data, communicating with cloud platform and triggering the alarm

    - **Camera Module**
        - Purpose- Capturing real time images and integrating it with the CNN model for real-time processing (uploading to the cloud)

    - **Alert Mechanism**
        - Buzzer: It will emit a sound to wake the driver if drowsiness is detected

**Software**

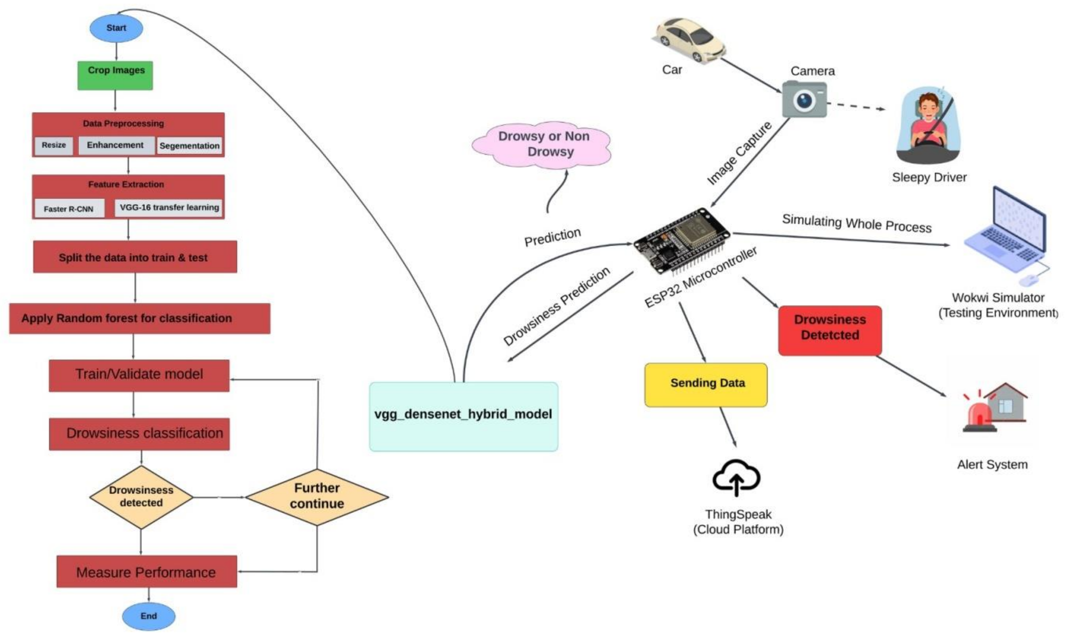
- **Wokwi Simulator**
    - Purpose- To simulate and prototype the circuit (ESP32) to test the functionality before the deployment

- **ThingSpeak**
    - Purpose- To provide the real-time data analytics and fetch the details from real-time processing of frames extracted from camera. It will be integrated with ESP32 using Http protocol to post the data and trigger result.

**Machine Learning Model**

- **Convolutional Neural Network (Vgg_DenseNet_Hybrid_Model)**
    - Purpose- Detect driver drowsiness by analyzing images of the driver's face, specifically focusing on eye closure, blinking or yawning.

## 3.1 Conceptual Design Diagram



## 3.2 Explanation of the diagram in steps

The diagram consists of following steps and components:-

1. Car and Camera:-Camera is mounted on the car dashboard to capture the real-time images of the driver, which is used to detect the driver's drowsiness state.

2. Image Capture:-The camera captures real-time images of the driver, which are send to the microcontroller for analysis.

3. ESP32 Microcontroller:-This is a central processing unit that manages data flow. It receives the images, processes the data and works with the deep learning model to check whether driver is feeling sleepy or not.

4. Vgg_densenet_hybrid_model (CNN) Deep Learning Model: -This is a hybrid convolutional neural network (CNN) model which has been used after applying certain parameters for image classification. It will predict whether the driver is drowsy or not based on the image data sent by the microcontroller .

5. Drowsiness Prediction and detection: - The result of the deep learning model is returned to the microcontroller and if the driver is detected to be drowsy, the alert system is triggered.

6. Sending Data: - If drowsiness is detected, signal (currently 1) is sent to the cloud platform (ThingSpeak) to trigger the buzzer else signal (currently 0) is sent which implies that the driver is awake .

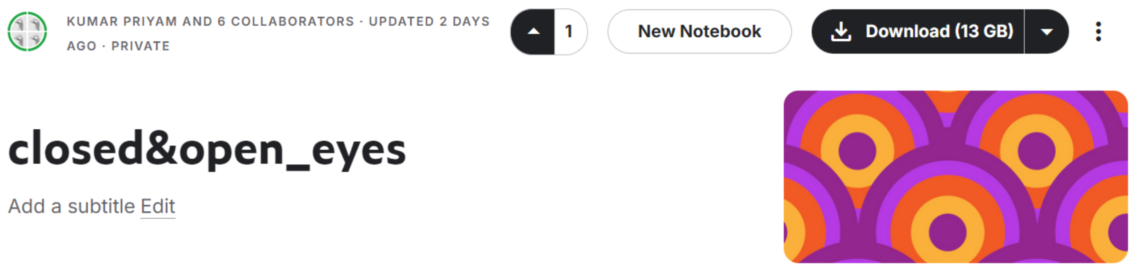
7. ThingSpeak (Cloud Platform):-It is a cloud-based platform used for sending and receiving IoT data. It collects information on detected drowsiness for monitoring or future analysis.

8. Alert System:- When drowsiness is detected, an alert system is triggered to alert or wake up the driver which consists of a buzzer embedded in the car.

9. Wokwi Simulator (Testing Environment): It is a testing environment that simulates the hardware and software interaction before deploying it in real-world conditions. It allows the entire process to be simulated, ensuring the system works as expected.

## 3.2 Working Example

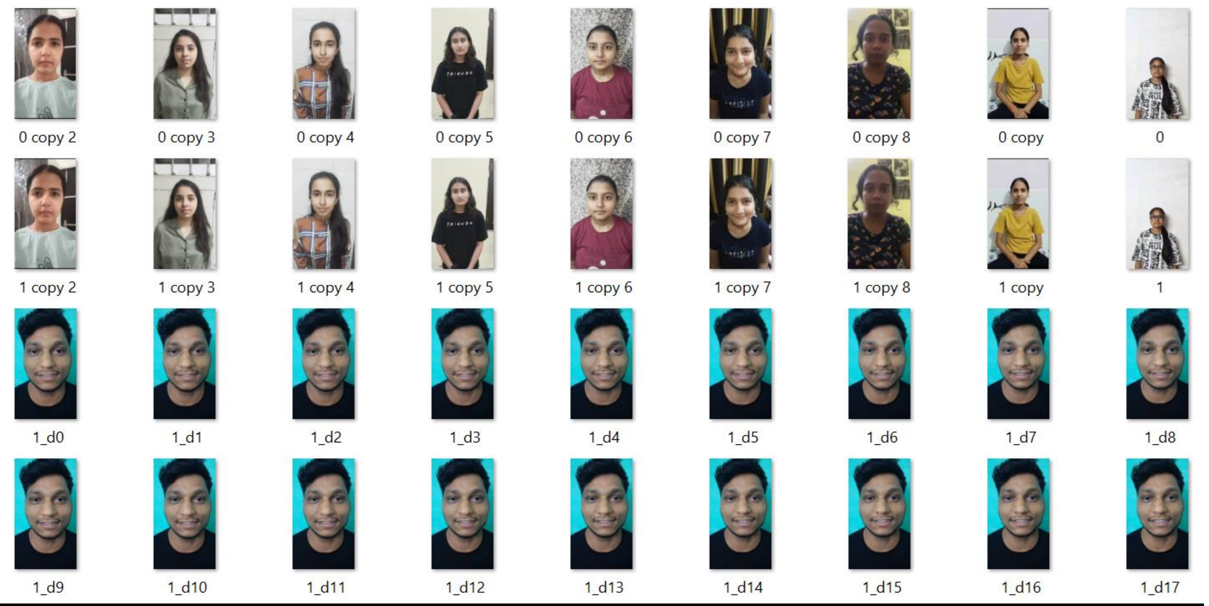
**<u>Dataset:</u>**



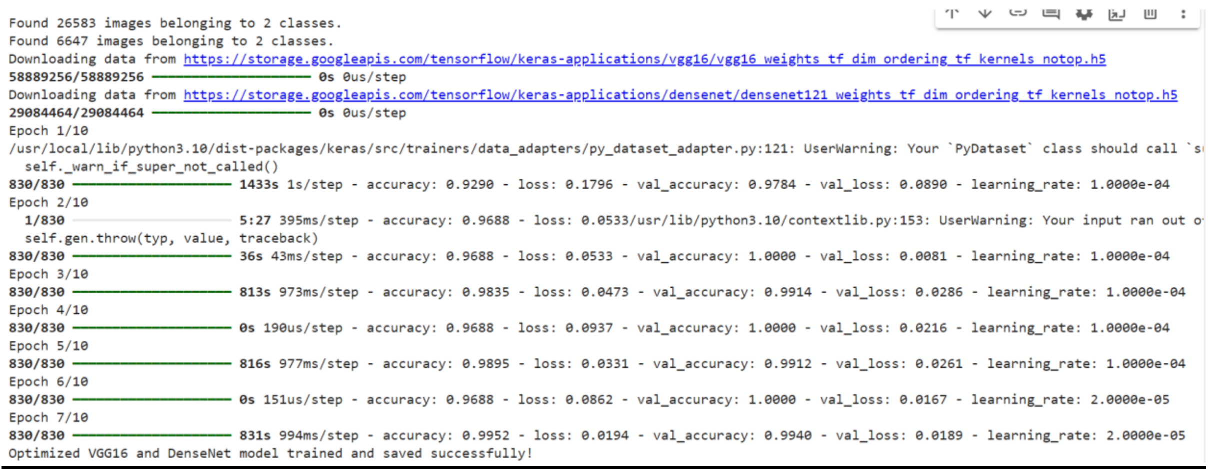
KUMAR PRIYAM AND 6 COLLABORATORS · UPDATED 2 DAYS AGO · PRIVATE

closed&open_eyes

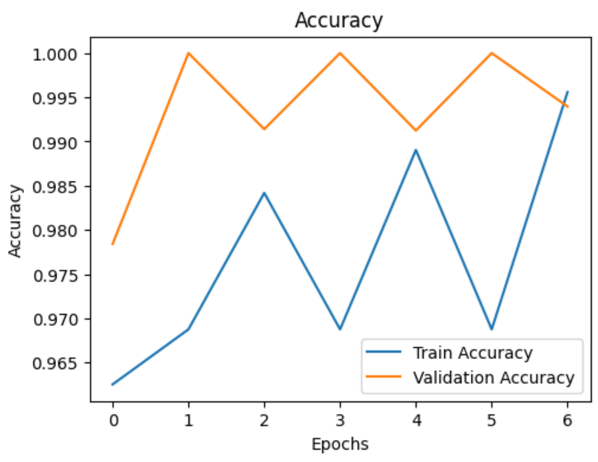
Add a subtitle Edit

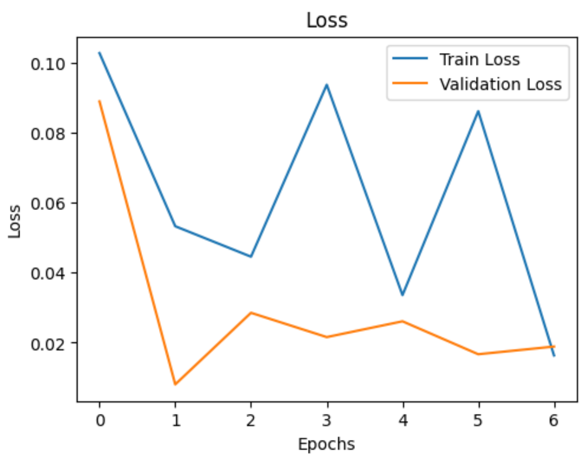


## Vgg_DenseNet_hybrid_model:

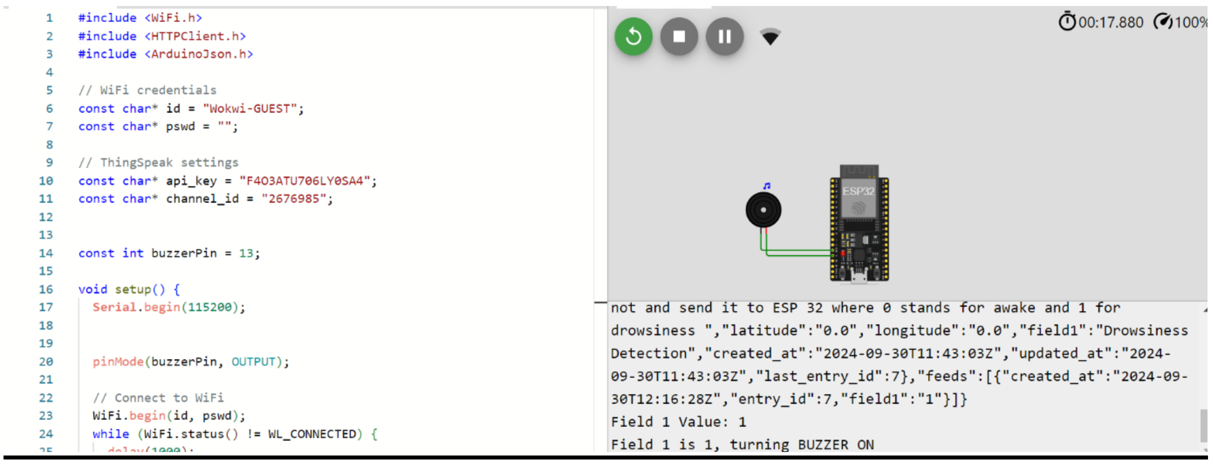


```
Found 26583 images belonging to 2 classes.
Found 6647 images belonging to 2 classes.
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5
58889256/58889256 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/densenet/densenet121_weights_tf_dim_ordering_tf_kernels_notop.h5
29084464/29084464 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
Epoch 1/10
/usr/local/lib/python3.10/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: UserWarning: Your `PyDataset` class should call `s
  self._warn_if_super_not_called()
830/830 ━━━━━━━━━━━━━━━━━━━━ 1433s 1s/step - accuracy: 0.9290 - loss: 0.1796 - val_accuracy: 0.9784 - val_loss: 0.0890 - learning_rate: 1.0000e-04
Epoch 2/10
  1/830 ━━━━━━━━━━━━━━━━━━━━ 5:27 395ms/step - accuracy: 0.9688 - loss: 0.0533/usr/lib/python3.10/contextlib.py:153: UserWarning: Your input ran out o
  self.gen.throw(typ, value, traceback)
830/830 ━━━━━━━━━━━━━━━━━━━━ 36s 43ms/step - accuracy: 0.9688 - loss: 0.0533 - val_accuracy: 1.0000 - val_loss: 0.0081 - learning_rate: 1.0000e-04
Epoch 3/10
830/830 ━━━━━━━━━━━━━━━━━━━━ 813s 973ms/step - accuracy: 0.9835 - loss: 0.0473 - val_accuracy: 0.9914 - val_loss: 0.0286 - learning_rate: 1.0000e-04
Epoch 4/10
830/830 ━━━━━━━━━━━━━━━━━━━━ 0s 190us/step - accuracy: 0.9688 - loss: 0.0937 - val_accuracy: 1.0000 - val_loss: 0.0216 - learning_rate: 1.0000e-04
Epoch 5/10
830/830 ━━━━━━━━━━━━━━━━━━━━ 816s 977ms/step - accuracy: 0.9895 - loss: 0.0331 - val_accuracy: 0.9912 - val_loss: 0.0261 - learning_rate: 1.0000e-04
Epoch 6/10
830/830 ━━━━━━━━━━━━━━━━━━━━ 0s 151us/step - accuracy: 0.9688 - loss: 0.0862 - val_accuracy: 1.0000 - val_loss: 0.0167 - learning_rate: 2.0000e-05
Epoch 7/10
830/830 ━━━━━━━━━━━━━━━━━━━━ 831s 994ms/step - accuracy: 0.9952 - loss: 0.0194 - val_accuracy: 0.9940 - val_loss: 0.0189 - learning_rate: 2.0000e-05
Optimized VGG16 and DenseNet model trained and saved successfully!
```



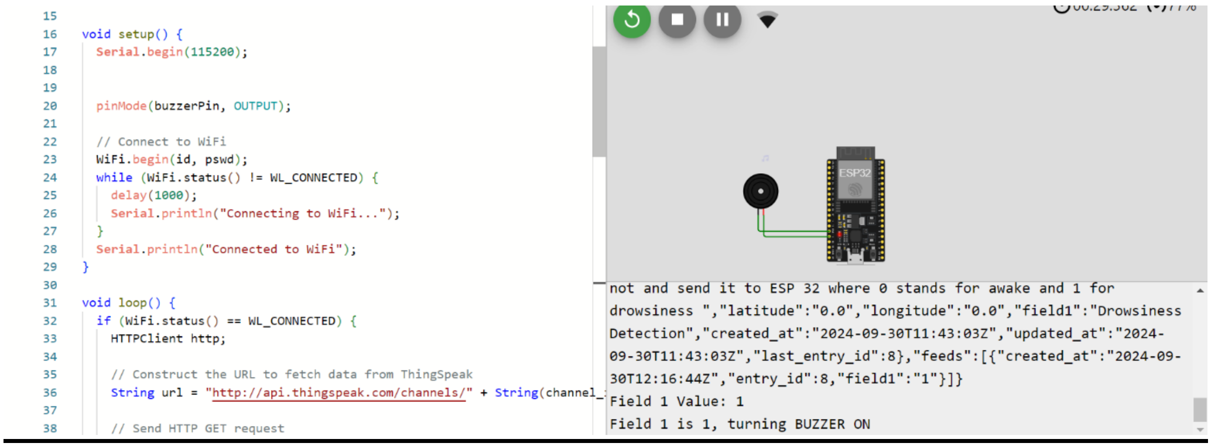


## Wokwi Simulation:



```cpp
#include <WiFi.h>
#include <HTTPClient.h>
#include <ArduinoJson.h>

// WiFi credentials
const char* id = "Wokwi-GUEST";
const char* pswd = "";

// ThingSpeak settings
const char* api_key = "F4O3ATU706LY0SA4";
const char* channel_id = "2676985";


const int buzzerPin = 13;

void setup() {
  Serial.begin(115200);


  pinMode(buzzerPin, OUTPUT);

  // Connect to WiFi
  WiFi.begin(id, pswd);
  while (WiFi.status() != WL_CONNECTED) {
```

```
not and send it to ESP 32 where 0 stands for awake and 1 for
drowsiness ","latitude":"0.0","longitude":"0.0","field1":"Drowsiness
Detection","created_at":"2024-09-30T11:43:03Z","updated_at":"2024-
09-30T11:43:03Z","last_entry_id":7},"feeds":[{"created_at":"2024-09-
30T12:16:28Z","entry_id":7,"field1":"1"}]}
Field 1 Value: 1
Field 1 is 1, turning BUZZER ON
```

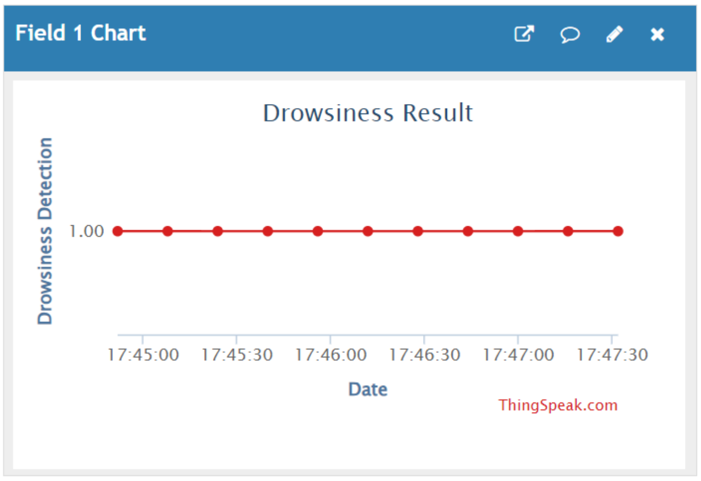


```cpp

void setup() {
  Serial.begin(115200);


  pinMode(buzzerPin, OUTPUT);

  // Connect to WiFi
  WiFi.begin(id, pswd);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }
  Serial.println("Connected to WiFi");
}

void loop() {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;

    // Construct the URL to fetch data from ThingSpeak
    String url = "http://api.thingspeak.com/channels/" + String(channel_

    // Send HTTP GET request
```

```
not and send it to ESP 32 where 0 stands for awake and 1 for
drowsiness ","latitude":"0.0","longitude":"0.0","field1":"Drowsiness
Detection","created_at":"2024-09-30T11:43:03Z","updated_at":"2024-
09-30T11:43:03Z","last_entry_id":8},"feeds":[{"created_at":"2024-09-
30T12:16:44Z","entry_id":8,"field1":"1"}]}
Field 1 Value: 1
Field 1 is 1, turning BUZZER ON
```

## ThingSpeak:

Entries: 10



## 4. Experimental Setup

### 4.1. Implementation Details:

The Camera will capture the driver’s face and then will be converted into frames. Then, the extracted images will be analyzed for drowsiness and the result would be sent to the cloud server ThingSpeak which will futher send signal to Wokwi simulator. Then with the help of ESP32 the alarm will be raised if needed.

**Table 1: Implementation Details**

| **Component** | **Details** | **Functionality** | **Integration** |
|---|---|---|---|
| ESP32 Microcontroller | Wifi Enabled | Processing of data, CNN inference, and communication with cloud services | Connected to Wi-Fi to fetch data from ThingSpeak, controls the alert system. |
| Camera | - | Capturing real-time images of the driver’s face for drowsiness detection. | Allows access of real-time images to the CNN model. |
| CNN Model | vgg_densenet_hybrid_model | Analyzing images of the driver’s face to detect drowsiness | Provides signal to ThingSpeak about driver drowsiness |
| ThingSpeak Cloud | Data logging, analytics, and visualization. | Collecting data from the ESP32, logging driver drowsiness detection, and sends alerts. | Connected via HTTP protocol, displays real-time monitoring data and stores event logs. |

|  |  |  |  |
|---|---|---|---|
| Wokwi Simulator | Simulation | Simulating ESP32 circuits to prototype and test the design | Simulates the ESP32's interactions Triggered by the ESP32 based on CNN model output |
| Buzzer | Small Piezo Buzzer | Providing audio alerts when the driver shows signs of drowsiness. | Triggered by the ESP32 based on CNN model output |

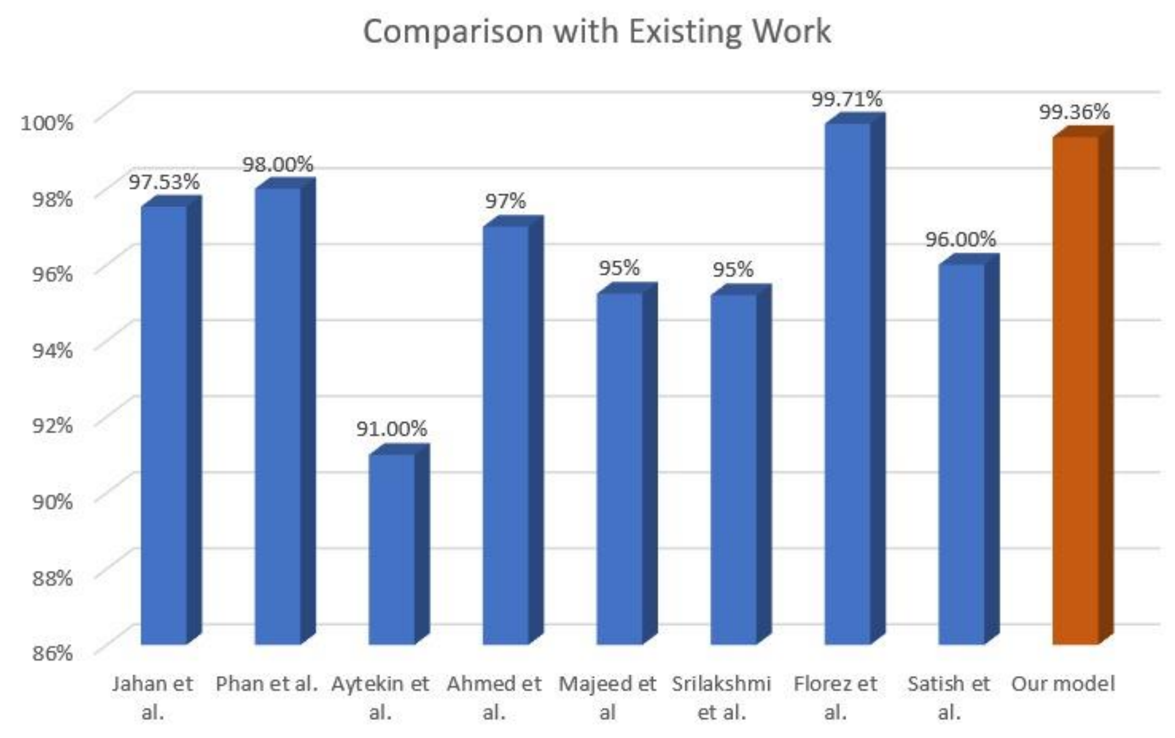
## 4.2. Experimental Results

| **Frame** | **Result** | **Buzzer State** |
|---|---|---|
| | **Driver is Awake 0.505893 18** | **OFF** |
| | **Driver is Awake 0.03609495 49** |  |
| | **Driver is Awake 0.30575123 79** |  |

| | | |
|---|---|---|
| | **Driver is Awake 0.9445113 17** | **OFF** |
| | **Driver is Awake 0.7772418 43** | |
| | **Driver is Awake 0.8824509 71** | |
| | **Drowsiness Detected 0.012284032 79** | **ON** |
| | **Drowsiness Detected 3.161284e-05 80** | |

| | | |
|---|---|---|
| | **Drowsiness Detected 2.2603268e-05 143 raise alarm** | |
| | **Drowsiness Detected 0.00039693434 134** | |
| | **Drowsiness Detected 0.00029022808 135** | **ON** |
| | **Drowsiness Detected 0.0006394791 197 raise alarm** | |

## 6. Comparison with the Existing Work



- Jahan et al[1] used deep learning models for detecting drowsiness by training VGG16, VGG19, and 4D on MRL dataset and achieved accuracies of 95.93% (VGG16), 95.03%(VGG19) and 97.50%(4D Model).The dataset they used includes left and right eye images.

- Phan et al[2] proposed an approach to detect driver drowsiness using IoT and deep neural networks improved from LSTM, VGG16, InceptionV3, and DenseNet. They used transfer learning technique combined with multiple drowsiness signs to detect the accuracy of the drowsiness detection which came out to be 98%.

- Aytekin et al[3] applied the pre-trained VGG16 model which uses transfer learning to determine whether the eyes are closed or open , whether the subject is yawning or not ultimately determining drowsiness of driver. The authors achieved an accuracy of 91%.

- Ahmed et al [4]evaluated the level of driver fatigue based on changes in a driver's eyeball movement using a convolutional neural network (CNN).They trained their model on 2900 images , VGG16 achieved efficiency of only 74% and CNN model achieved a good efficiency of around 97%.

- Majeed et al[5] study proposed a deep neural network architecture for drowsiness detection. Authors used the DLIB library to locate key facial points to calculate the mouth aspect ratio (MAR), further training their model on a small dataset and classifying in 'yawning' and 'no_yawning' classes. Their results demonstrated that the proposed CNN model achieved an average accuracy of 96.69%.

- Srilakshmi et al [6]used several algorithms to detect drowsiness, including K-Nearest Neighbors (KNN), Support Vector Machines (SVMs), Naive Bayes, Random Forests, Logistic Regression, and Decision Trees. Among the different algorithms they used the best results were given by Modified SVM(95.20%) which are used for both classification and regression.

- In Florez et al[7] paper ROI extraction (Region of Interest) was used to check whether eyes are open or closed. They trained 3 deep learning models InceptionV3, Resnet50V2 and VGG16 on NITYMED dataset. They achieved exceptional accuracies, 99.71% with Resnet50V2, 99.41% with VGG16 and 99.31% with InceptionV3.

- Satish et al.[8] used Python machine learning and Arduino along with camera and load cell sensors. The facial and eye detection was done by OpenCV and HOG algorithms. Driver face was captured, the eye retina detection and facial feature extraction were done and blinking values were calculated then threshold values were set. Secondly, the Aurdino module was used to integrate with elastomeric sensors for real-time calculation of driver hand pressure on the car steering wheel and the threshold value was set. The result from both methods were taken as input for taking the final decision and alerting the driver. The accuracy in the result was 96% when Live video with pressure sensor and eyeblink data were used .

- Hossain et al[9] used a Pi camera to capture the images of the driver's eye and the entire system was incorporated using Raspberry-Pi. If the eye closure ratio deteriorated from the standard ratio, the driver was alerted. The threshold value for the EAR (the eye closure ratio) was considered 0.25 and any value above this would meant that the driver’s eyes were open. However while ,carrying out the experiment at night, the system failed to calculate EAR values correctly.

- Our model uses Convolutional Neural Networks(CNN) like many of them have used but with the help of our custom made dataset we trained VGG16 and DenseNet models separately and combined. The hybrid model outperformed individual models which showed that their combination along with some other techniques gives an improved model with an average accuracy of 99.36%.

# 7. Conclusion and Future Work

## Conclusion:

In conclusion, our research on detecting whether the driver is drowsy or not provides a proactive approach to minimizing road accidents due to sleepiness thereby enhancing road safety. By training a vgg_densenet_hybrid ML model we accuracy predict the driver drowsiness with our validation accuracy being 99.36%. We have demonstrated the real time monitoring and alert system in preventing critical accidents. The integration of Wokwi simulator and buzzer alert system ensures that driver is informed in a timely manner which would enable the driver to take corrective actions before accidents occur. Furthermore, the use of ThingSpeak for data logging offers a valuable resource for continuous analysis and improvements. Overall, this project highlights the importance of utilizing machine learning and IoT technologies to address real-world problems like drowsy driving, contributing to safer roads and reduced accident rates.

## Future Work:

- **Improvement of CNN Model Accuracy:** We will train the model on a more diverse dataset and try other CNN model types to enhance real-time performance and at the same time maintaining accuracy.
- **Real Time Data Feedback:** We will work on implementing real-time data feedback to alert nearby emergency services or loved ones in case of extended periods of detected drowsiness.
- **Enhanced Alert System:** Expand the alert system by incorporating smart wearable devices (like smartwatches) to deliver notifications and vibrations to drivers in real time.

- **Managing Computational Power:** Working on reducing computational power needed to process the images in real-time so that there is no lag in alerting the driver.
- **Improving Efficiency in lack of light:** As our dataset only includes videos recorded in daylight it might not be able to give efficient results in lack of light.

These future improvements can enhance the functionality, accuracy, and user experience of the system, making it a more robust and reliable tool for preventing accidents caused by drowsy driving.

## References

**[1]** Jahan, I.; Uddin, K.M.A.; Murad, S.A.; Miah, M.S.U.; Khan, T.Z.; Masud, M.; Aljahdali, S.; Bairagi, A.K. 4D: A Real-Time Driver Drowsiness Detector Using Deep Learning. Electronics 2023, 12, 235. https://doi.org/10.3390/ electronics12010235 Academic Editor: Dimitris Apostolou. Received: 31 October 2022 Revised: 12 December 2022 Accepted: 23 December 2022 Published: 3 January 2023

**[2]** Anh-Cang Phan, Thanh-Ngoan Trieu, Thuong-Cang Phan ·

**[3]** Alper Aytekin1*, Vasfiye Mençik2, 1Yıldız Technical University, İstanbul, Turkey 2Dicle University, Diyarbakır, Turkey. author's e-mail: alper.aytekin@std.yildiz.edu.tr, ©2022 Alper Aytekin, Vasfiye Mençik.

**[4]** Ahmed, M.I.B.; Alabdulkarem, H.; Alomair, F.; Aldossary, D.; Alahmari, M.; Alhumaidan, M.; Alrassan, S.; Rahman, A.; Youldash, M.; Zaman, G. ADeep-Learning Approach to Driver Drowsiness Detection. Safety 2023, 9, 65. https://doi.org/10.3390/ safety9030065 Academic Editor: Raphael Grzebieta Received: 31 May 2023 Revised: 30 August 2023 Accepted: 31 August 2023 Published: 13 September 2023.

**[5]** Majeed, F.; Shafique, U.; Safran, M.; Alfarhood, S.; Ashraf, I. Detection of Drowsiness among Drivers Using Novel Deep Convolutional Neural Network Model. Sensors 2023, 23, 8741. https://doi.org/10.3390/s23218741 Academic Editor: Felipe Jiménez Received: 10 September 2023 Revised: 20 October 2023 Accepted: 24 October 2023 Published: 26 October 2023.

**[6]** T. Srilakshmi Department of Computer Science and Engineering, Prasad V Potluri Siddhartha Institute of Technology Vijayawada- 52007, India tslakshmi@pvpsiddhartha.ac.in Lakshmi Ramani Burra Department of Computer Science and Engineering, Prasad V Potluri Siddhartha Institute of Technology Vijayawada- 52007, India blramani@pvpsiddhartha.ac.in, Harshavardhan Reddy Student, Department of Computer Science and Engineering, Prasad V Potluri Siddhartha Institute of Technology rharsha102002@gmail.com Mohana Vamsi Thota Student, Department of Computer Science and Engineering, Prasad V Potluri Siddhartha Institute of Technology Vijayawada- 52007, India mohanavamsi15@gmail.com, Yaswanthi Potluri Department of Computer Science and Engineering, Prasad V Potluri Siddhartha Institute of Technology Vijayawada- 52007, India potluriyaswanthi@gmail.com Rishita Gundimeda Student, Department of Computer Science and Engineering, Prasad V Potluri Siddhartha Institute of Technology Vijayawada- 52007, India rishitagundimeda902@gmail.com.

**[7]** Florez, R.; Palomino Quispe, F.; Coaquira-Castillo, R.J.; Herrera-Levano, J.C.; Paixão, T.; Alvarez, A.B. A CNN-Based Approach for Driver Drowsiness Detection by Real-Time Eye State Identification. Appl. Sci. 2023, 13, 7849. https://doi.org/10.3390/ app13137849 Academic Editor: Hui Yuan Received: 1 June 2023 Revised: 29 June 2023 Accepted: 30 June 2023 Published: 4 July 2023.

**[8]** K. Satish, A. Lalitesh, K. Bhargavi, M. Sishir Prem and Anjali.T International Conference on Communication and Signal Processing, July 28 - 30, 2020, India.

**[9]** Md. Yousuf Hossain Department of Computer Science and Engineering BRAC University Dhaka, Bangladesh yousuffahim8@gmail.com , Fabian Parsia George Department of Computer Science and Engineering BRAC University Dhaka, Bangladesh fabian.george96@gmail.com.