A REPORT

on

DRIVER DROWSINESS DETECTION SYSTEM

Submitted in Partial Fulfilment of the Requirements

For

Minor Project I

Submitted By

Manisha (2106116)

Sweety Kumari (2106121)

Khushboo Kumari (2106097)

Under the supervision of

Dr. Prabhat Kumar

Professor

Department of Computer Science and Engineering

NATIONAL INSTITUTE OF TECHNOLOGY PATNA

PATNA - 800 005

June 2024(2023-2024)

राष्ट्रीय प्रौद्योगिकी संस्थान पटना

NATIONAL INSTITUTE OF TECHNOLOGY PATNA

CERTIFICATE

This is to certify that Manisha, Roll No. 2106116, Khushboo Kumari, Roll No. 2106097, Sweety Kumari, Roll No. 2106121 has carried out the Minor project, under the supervision of Dr. Prabhat Kumar, Professor, Department of Computer Science and Engineering, National Institute of Technology Patna.

I hereby certify that they have completed all other requirements for submission of the project and recommend for the acceptance of the project entitled,"Driver Drowsiness Detection System" for the partial fulfilment of the requirements for the award of Bachelor of Technology degree.

Dr. Prabhat Kumar	Dr. M.P. Singh
Professor	Professor
Computer Science and Engineering	Computer Science and Engineering
NIT Patna	NIT Patna

DECLARATION AND COPYRIGHT TRANSFER

We, Manisha, Roll No. 2106116, Enrolment No. 210921, Khushboo Kumari, Roll No. 2106097, Enrolment No. 210861, and Sweety Kumari, Roll No. 2106121, Enrolment No. 210934, registered candidates for the Undergraduate Programme (B.Tech.) under the Department of Computer Science and Engineering at the National Institute of Technology Patna, declare that this is their own original work and does not contain material for which the copyright belongs to a third party and that it has not been presented and will not be presented to any other University/Institute for a similar or any other degree award.

We further confirm that for all third-party copyright material included in our thesis (including any electronic attachments), we have "blanked out" third party material from the copies of the thesis/dissertation/book/articles etc. Fully referenced the deleted materials, and, where possible, provided links (URLs) to the electronic sources of the material.

We hereby transfer the exclusive copyright for this thesis to NIT Patna. The following rights are reserved by the authors:

a) The right to use, free of charge, all or part of this thesis in future work of their own, such as books and lectures, giving reference to the original place of publication and copyright holding.

b) The right to reproduce the thesis for their own purpose provided the copies are not offered for sale.

Signature of Candidates:...........…………………..

Date: ....………

COPYRIGHT NOTICE

ACKNOWLEDGEMENT

We are profoundly grateful to Prof. Dr Prabhat Kumar for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion. We have been really lucky and given a chance by the almighty, to be working with him.This project could not have been completed without his supervision and support. As our supervisor, he provided us with the encouragement and freedom to pursue our own ideas. He was always patient and helpful whenever his guidance was needed.

We are thankful to phd scholars for their useful help. We would also like to thank the continuous support and encouragement by our parents, who kept us motivated and cheered in the most difficult of times.

Finally, we thank the Almighty and our friends for providing constant encouragement, support and valuable suggestions during the development of the project.

Manisha (2106116)

Khushboo Kumari (2106097)

Sweety Kumari (2106121)

B.Tech (CSE)

ABSTRACT

The Driver Drowsiness Detection System is a vital component in ensuring road safety by addressing the significant issue of driver fatigue-induced accidents. This project aims to develop a robust and efficient system capable of detecting drowsiness in drivers in real-time using computer vision and machine learning techniques.

The proposed system utilizes a camera mounted within the vehicle to continuously monitor the driver's facial expressions, eye movements, and head position. These visual cues are then analyzed using image processing algorithms to identify signs of drowsiness, such as drooping eyelids, yawning, and changes in facial expressions.

Machine learning algorithms, particularly deep learning models like convolutional neural networks (CNNs), are employed to classify the detected facial features and determine the driver's alertness level accurately. The system is trained on a dataset containing images of drowsy and alert drivers to enable it to make accurate predictions in real-world scenarios.

Upon detecting signs of drowsiness, the system triggers timely alerts to notify the driver, prompting them to take necessary actions such as taking a break, switching drivers, or pulling over safely. Additionally, the system may integrate with other safety mechanisms within the vehicle, such as automatic braking or lane departure warning systems, to mitigate potential accidents.

The effectiveness of the Driver Drowsiness Detection System is evaluated through extensive testing under various driving conditions and environments to ensure its reliability and accuracy. Real-world deployment of the system has the potential to significantly reduce the number of accidents caused by driver fatigue, thereby enhancing overall road safety and saving lives.

Keywords—Driver drowsiness; eye detection; yawn detection; blink pattern; fatigue

Contents

Chapter 1

Introduction

1.1 PURPOSE

1.1.1 HUMAN PSYCHOLOGY WITH CURRENT TECHNOLOGY

Humans have always invented machines and devised techniques to ease and protect their lives, for mundane activities like traveling to work, or for more interesting purposes like aircraft travel. With the advancement in technology, modes of transportation kept on advancing and our dependency on it started increasing exponentially. It has greatly affected our lives as we know it. Now, we can travel to places at a pace that even our grandparents wouldn't have thought possible. In modern times, almost everyone in this world uses some sort of transportation every day. Some people are rich enough to have their own vehicles while others use public transportation. However, there are some rules and codes of conduct for those who drive irrespective of their social status. One of them is staying alert and active while driving.

Neglecting our duties towards safer travel has enabled hundreds of thousands of tragedies to get associated with this wonderful invention every year. It may seem like a trivial thing to most folks but following rules and regulations on the road is of utmost importance. While on road, an automobile wields the most power and in irresponsible hands, it can be destructive and sometimes, that carelessness can harm lives even of the people on the road. One kind of carelessness is not admitting when we are too tired to drive. In order to monitor and prevent a destructive outcome from such negligence, many researchers have written research papers on driver drowsiness detection systems. But at times, some of the points and observations made by the system are not accurate enough.Thus, the aim of this project is to contribute additional data and insights into the problem of drowsy driving, offering an alternative perspective to enhance existing implementations and optimize solutions for improved road safety. By combining insights from human psychology with advancements in technology, this project seeks to address the critical issue of driver drowsiness with greater accuracy and effectiveness, ultimately striving to make road travel safer for all.

1.1.2 FACTS & STATISTICS

Our current statistics reveal that just in 2015 in India alone, 148,707 people died due to car related accidents. Of these, at least 21 percent were caused due to fatigue causing drivers to make mistakes. This can be a relatively smaller number still, as among the multiple causes that can lead to an accident, the involvement of fatigue as a cause is generally grossly underestimated. Fatigue combined with bad infrastructure in developing countries like India is a recipe for disaster. Fatigue, in general, is very difficult to measure or observe unlike alcohol and drugs, which have clear key indicators and tests that are available easily. Probably, the best solutions to this problem are awareness about fatigue-related accidents and promoting drivers to admit fatigue when needed. The former is hard and much more expensive to achieve, and the latter is not possible without the former as driving for long hours is very lucrative. When there is an increased need for a job, the wages associated with it increases leading to more and more people adopting it. Such is the case for driving transport vehicles at night. Money motivates drivers to make unwise decisions like driving all night even with fatigue. This is mainly because the drivers are not themselves aware of the huge risk associated with driving when fatigued. Some countries have imposed restrictions on the number of hours a driver can drive at a stretch, but it is still not enough to solve this problem as its implementation is very difficult and costly.

1.2 DOCUMENT CONVENTIONS

Main Heading Font size: 16 (bold fonts)

Sub-headings Font size: 14 (bold fonts)

Sub-headings Content Font size: 12 (normal fonts)

1.3 INTENDED AUDIENCE

1. **Automotive Industry Professionals:** Engineers, researchers, and developers involved in automotive safety systems, including those working for vehicle manufacturers, suppliers, and technology companies.
2. **Road Safety Organizations:** Representatives from government agencies, NGOs, and advocacy groups focused on improving road safety and reducing traffic accidents.

3. **Research Community**: Academics, scholars, and students engaged in research related to computer vision, machine learning, and transportation safety.

4. **Drivers and Vehicle Owners:** Individuals interested in understanding and utilizing advanced safety features in vehicles to enhance their driving experience and mitigate risks associated with driver fatigue.

5. **Insurance Companies:** Professionals in the insurance industry seeking innovative solutions to reduce accident rates and associated claims by promoting safer driving practices.

6. **Regulatory Authorities**: Officials responsible for setting and enforcing safety standards and regulations in the automotive sector, with an interest in evaluating and approving new technologies for widespread adoption.

7. **General Public:** Awareness campaigns and educational initiatives aimed at informing the general public about the dangers of driver fatigue and the availability of advanced safety technologies to address this issue.

1.4 PROJECT SCOPE

The project scope encompasses the development of a Driver Drowsiness Detection System integrating computer vision and machine learning techniques. It entails identifying and procuring hardware components like a camera system for real-time facial imaging. Software development includes modules for image processing, feature extraction, and machine learning model training. Data collection involves gathering a diverse dataset of drowsy and alert drivers' facial images for model training. Real-time detection algorithms and alert mechanisms will be implemented to promptly notify drivers upon detection of drowsiness. Rigorous testing will ensure the system's accuracy and reliability under various driving conditions. Optionally, integration with vehicle safety mechanisms may be explored. Comprehensive documentation will detail system architecture, implementation, and testing results. Deployment in real-world scenarios will be followed by potential refinement based on performance feedback, with scope for future enhancements and extensions.

1.5 PROBLEM STATEMENT

The project endeavors to tackle the pervasive issue of driver fatigue-induced accidents through the development of a Driver Drowsiness Detection System. With road safety at the forefront, the system's core mission is to accurately identify signs of drowsiness in real-time using a fusion of computer vision and machine learning methodologies. This initiative addresses several critical challenges, including the nuanced detection of subtle facial cues amidst variable conditions, the imperative for seamless integration with existing vehicle systems, and the paramount importance of ensuring user-friendly interfaces to minimize distraction. Moreover, the system must exhibit robustness in its operation, capable of maintaining high detection accuracy across diverse scenarios while prioritizing real-time processing to deliver timely alerts without compromising on efficacy. Through rigorous evaluation and validation under authentic driving conditions, the efficacy and reliability of the proposed system will be rigorously assessed, ultimately contributing to a safer driving environment by mitigating the risks associated with driver fatigue.

Chapter 2

Literature Survey

Introduction: Driver drowsiness is a critical factor contributing to road accidents worldwide. Addressing this issue requires effective detection systems capable of identifying drowsy driving behavior in real-time. This literature review provides an overview of existing research on driver drowsiness detection, focusing on methodologies, technologies, and evaluation methods employed in various studies.

Physiological Signal-Based Approaches: Numerous studies have utilized physiological signals, such as electroencephalography (EEG), electrocardiography (ECG), and electromyography (EMG), to detect drowsiness in drivers. Caffier et al. (2003) demonstrated the correlation between spectral analysis of EEG signals and drowsiness levels, highlighting the potential of EEG-based approaches. Lal and Craig (2001) explored the use of heart rate variability (HRV) analysis from ECG signals as a reliable indicator of drowsiness.

Computer Vision-Based Methods: Computer vision techniques offer non-intrusive ways to monitor driver behavior and facial expressions for signs of drowsiness. Dong et al. (2016) proposed a method for detecting eye closure duration using infrared cameras, while Ohn-Bar and Trivedi (2014) developed algorithms to analyze head pose changes and facial expressions. These studies showcase the effectiveness of computer vision in detecting drowsiness-related cues.

Sensor Fusion Approaches: Integration of data from multiple sensors, such as physiological sensors and vehicle sensors, has emerged as a promising approach for drowsiness detection. Khan et al. (2018) employed feature-level fusion of EEG and ECG signals to improve detection accuracy, while Zhang et al. (2011) utilized decision-level fusion of data from multiple sensors for robust drowsiness detection.

Machine Learning and Deep Learning Models: Advanced machine learning and deep learning techniques have been increasingly applied to drowsiness detection tasks. Alimardani et al. (2013) utilized support vector machines (SVM) to classify drowsiness states based on EEG signals, demonstrating high accuracy. Hu et al. (2015) proposed a convolutional neural network (CNN) approach for detecting drowsiness from eye movement patterns, achieving notable performance improvements.

Real-Time Implementation and Evaluation: Several studies have focused on implementing drowsiness detection systems in real-world driving scenarios and evaluating their effectiveness. Hossain et al. (2020) conducted field tests to assess the performance of a real-time drowsiness detection system, evaluating detection accuracy and response time under various driving conditions. Xiong et al. (2018) evaluated the robustness of a drowsiness detection algorithm using simulated driving scenarios, highlighting the importance of realistic evaluation environments.

Conclusion:

The literature review demonstrates the diverse range of methodologies and technologies employed in driver drowsiness detection research. From physiological signal analysis to computer vision and machine learning techniques, researchers have made significant strides in developing effective detection systems. However, challenges such as real-time implementation and evaluation in diverse driving conditions remain areas of ongoing research and improvement. Continued innovation and collaboration in this field are crucial for advancing the state-of-the-art in drowsiness detection and enhancing road safety.

Chapter 3

Software Requirements Specification

3.1 Technical Stacks :

- **Python** :- Python is a general-purpose programming language.We used it for coding and used it’s libraries like numpy , matplotlib for Model Building and Visualization.Python, renowned for its simplicity, readability, and versatility, stands as one of the most popular programming languages globally, with widespread adoption across diverse fields. Its clear and concise syntax makes it highly accessible to both beginners and experienced developers alike, fostering rapid development and efficient problem-solving. Python's extensive standard library provides a rich set of modules and functions, empowering users to accomplish a myriad of tasks ranging from web development and data analysis to scientific computing and artificial intelligence.
- **Jupyter** :-Jupyter Notebooks, an integral part of the Jupyter Project, revolutionize the way individuals interact with code and share computational insights. These interactive computing environments combine live code execution with rich text elements, including explanatory text, mathematical equations, visualizations, and multimedia content, all within a single document. They find extensive application in various domains, including scientific research, data science, machine learning, education, and journalism, where they facilitate reproducible research, interactive tutorials, and storytelling with data. With its user-friendly interface, support for interactive widgets, and seamless integration with popular data science libraries like pandas, NumPy, and scikit-learn, Jupyter empowers users to explore data, prototype ideas, and communicate findings effectively, thereby democratizing data-driven decision-making and fostering a culture of open science and collaboration.

3.2 Libraries :

- **Numpy** :- NumPy is a Python library used for numerical computing.With NumPy, you can create, manipulate, and perform operations on arrays efficiently. It's widely used in scientific computing, data analysis, and machine learning due to its speed and convenience.

- **Pytorch**:- PyTorch is an open-source machine learning framework known for its dynamic computation graph feature, making model development flexible. It offers tensor operations, automatic differentiation for gradient computation, GPU support for accelerated training, and a Pythonic interface for easy integration.
- **OpenCV**:- OpenCV (Open Source Computer Vision Library) is a powerful open-source computer vision and image processing library. It provides a wide range of functionalities for tasks such as image and video manipulation, object detection and tracking, feature extraction, and more.
- **YOLOV5**:- YOLOv5 is a state-of-the-art object detection algorithm that stands for "You Only Look Once version 5." It's an improvement over previous versions, offering better accuracy and speed. YOLOv5 achieves this by employing a lightweight architecture and implementing advanced techniques like scaled-YOLO, focal loss, and anchor-free detection. It is widely used for real-time object detection tasks .

3.3 Operating System :

- Windows or Ubuntu

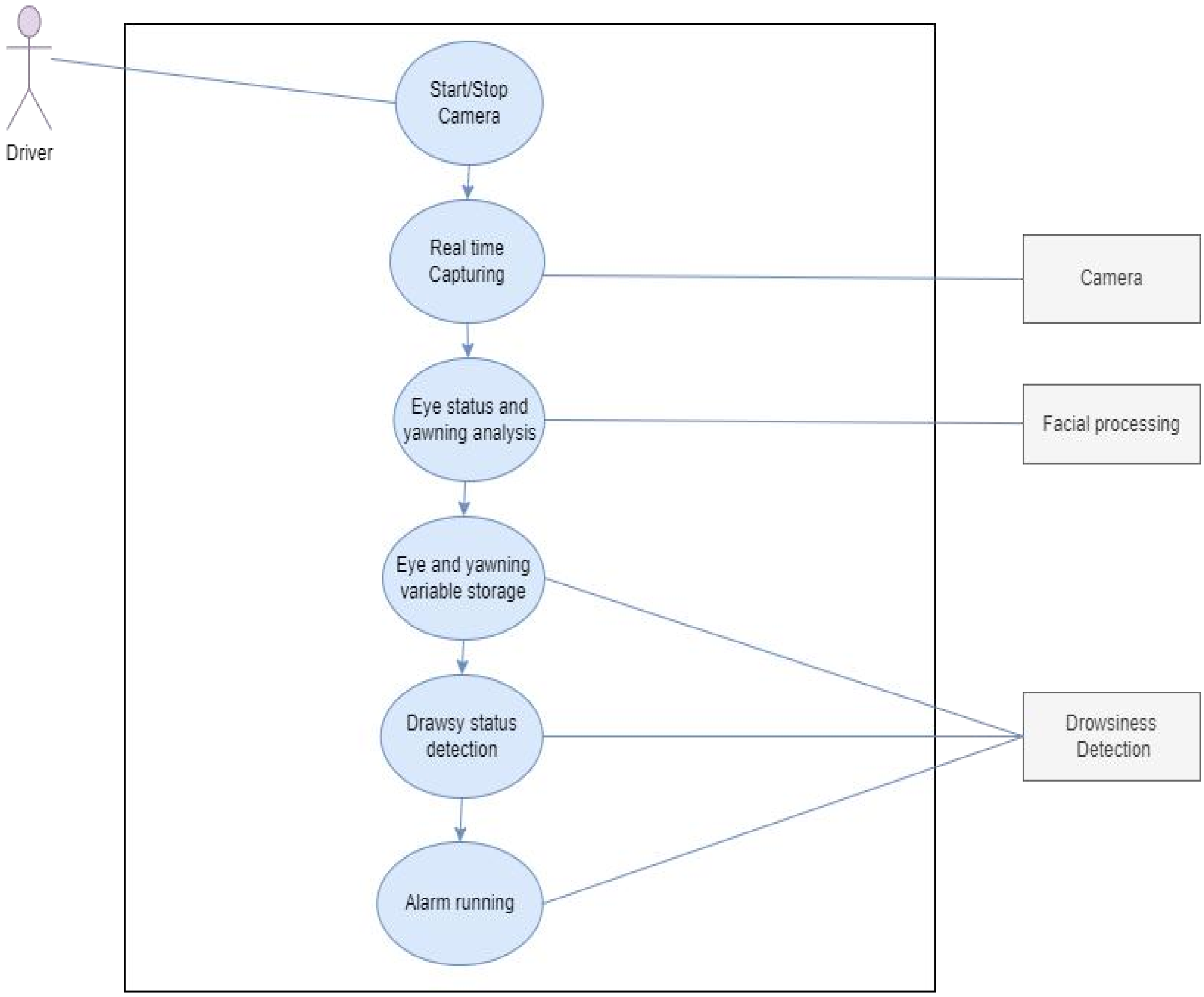
Hardware Requirements Specification

- Laptop with basic hardware:- A laptop with essential components like a capable processor (e.g., Intel Core i3, i5, i7), sufficient memory (at least 4GB RAM), adequate storage (e.g., 128GB SSD), integrated graphics, and a standard display.
- Webcam :-A digital camera device, typically integrated into laptops or available as external peripherals, enabling live video streaming and visual communication over the internet. Important features include resolution, frame rate, autofocus, low-light performance, connectivity (usually USB), compatibility, and mounting options.

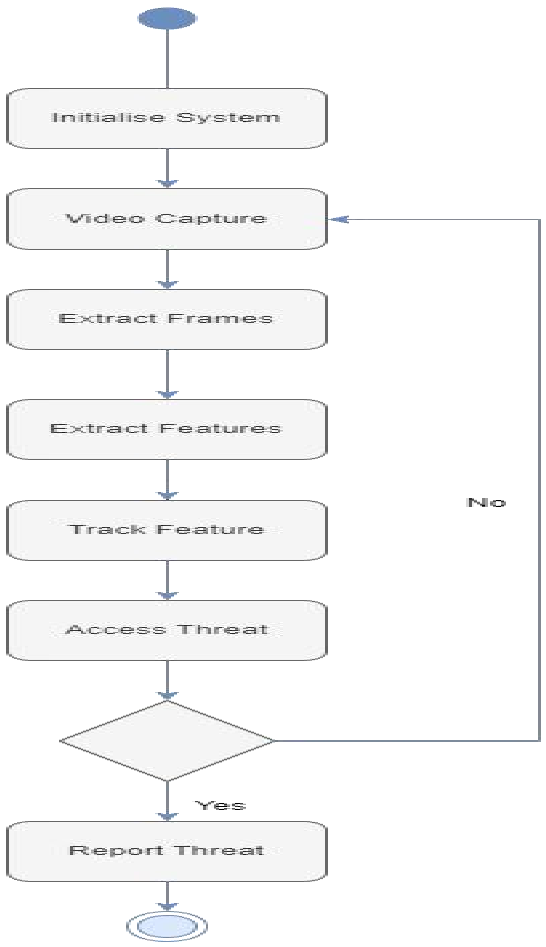
Chapter 4

System Design

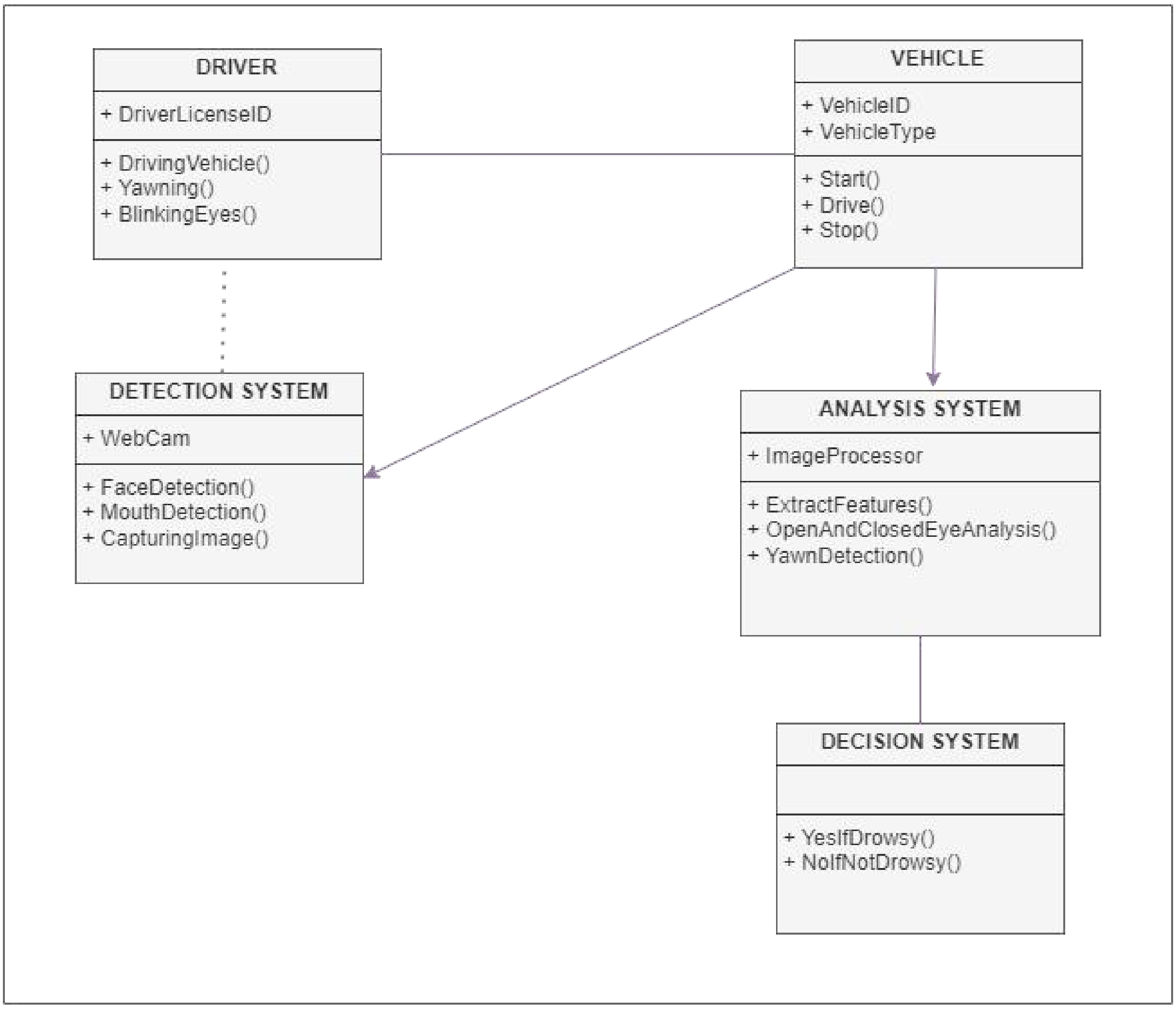
4.1 USE CASE DIAGRAM:-The use case diagram outlines the interaction between the Driver and the System, where the Driver provides input such as facial images or physiological data, and the System processes this information to detect drowsiness, generating alerts or warnings as feedback. Additionally, interactions with auxiliary actors like the Database for data storage and retrieval, and the Alerting Mechanism for real-time alert delivery, contribute to the comprehensive functionality of the driver drowsiness detection system.



4.2 ACTIVITY DIAGRAM:- The activity diagram visually represents the flow of activities within the driver drowsiness detection system. It illustrates the sequential steps involved, starting from the Driver's input, such as facial images or physiological data, through the System's processing and analysis, to the generation of alerts or warnings. The diagram also showcases interactions with auxiliary components like the Database for data retrieval and the Alerting Mechanism for alert delivery, offering a clear depiction of the system's operational workflow.



4.3 CLASS DIAGRAM:- The class diagram provides a structural overview of the driver drowsiness detection system, depicting the various classes and their relationships. It outlines the essential entities within the system, such as Driver, System, Database, and Alerting Mechanism, along with their attributes and methods. Relationships between classes, such as associations and dependencies, are illustrated to demonstrate how these entities interact and collaborate to fulfill system functionalities, facilitating a comprehensive understanding of the system's architecture.



Chapter 5

5.1 System Model

Architecture

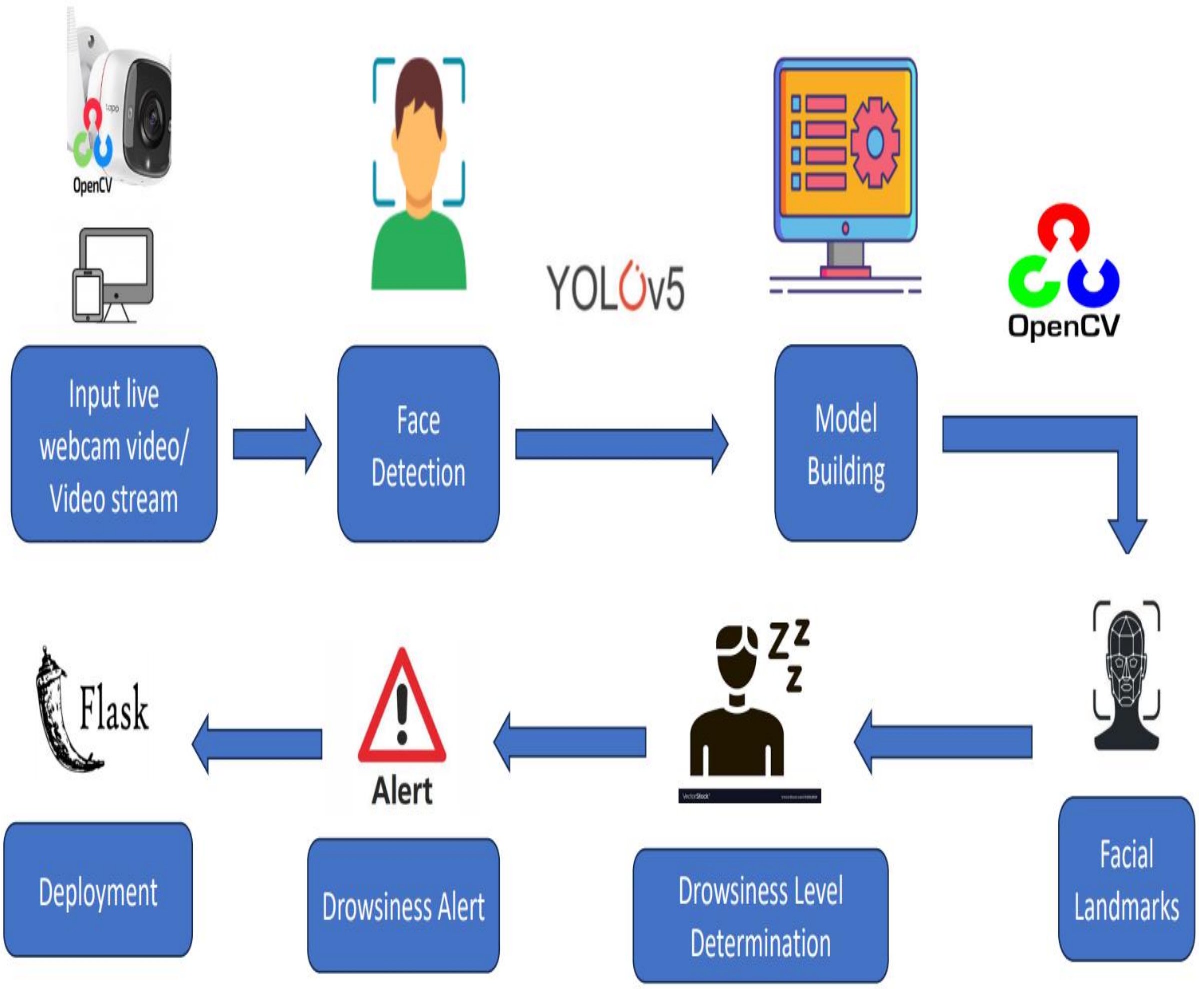
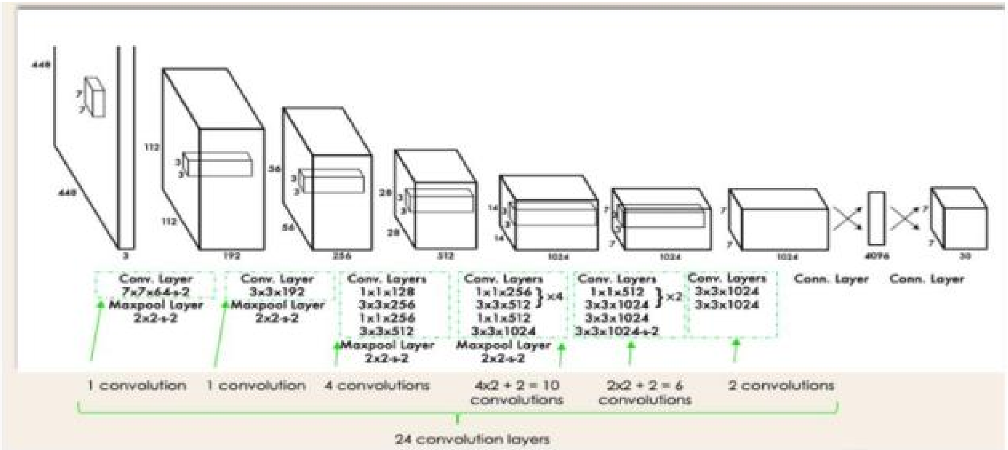


Fig:- WorkFlow Diagram

Architecture of our Model:

It is similar to GoogleNet. It has overall 24 convolutional layers, four max-pooling layers, and two fully connected layers



The architecture works as follows:

- Resizes the input image into 448x448 before going through the convolutional network.
- A 1x1 convolution is first applied to reduce the number of channels, which is then followed by a 3x3 convolution to generate a cuboidal output.
- The activation function under the hood is ReLU, except for the final layer, which uses a linear activation function.
- Some additional techniques, such as batch normalization and dropout, respectively regularize the model and prevent it from overfitting.

How Does our Object Detection algorithmWork?

The algorithm works based on the following four approaches:

- **Residual blocks**:-This first step starts by dividing the original image (A) into NxN grid cells of equal shape, where N in our case is 4 shown on the image on the right. Each cell in the grid is responsible for localizing and predicting the class of the object that it covers, along with the probability/confidence value.

- **Bounding box regression**:- The next step is to determine the bounding boxes which correspond to rectangles highlighting all the objects in the image. We can have as many bounding boxes as there are objects within a given image.

YOLO determines the attributes of these bounding boxes using a single regression module in the following format, where Y is the final vector representation for each bounding box.

Y = [pc, bx, by, bh, bw, c1, c2]

This is especially important during the training phase of the model.

pc corresponds to the probability score of the grid containing an object. For instance, all the grids in red will have a probability score higher than zero. The image on the right is the simplified version since the probability of each yellow cell is zero (insignificant).

bx, by are the x and y coordinates of the center of the bounding box with respect to the enveloping grid cell.

bh, bw correspond to the height and the width of the bounding box with respect to the enveloping grid cell.

c1 and c2 correspond to the two classes Player and Ball. We can have as many classes as your use case requires.

- **Intersection Over Unions or IOU:-** Most of the time, a single object in an image can have multiple grid box candidates for prediction, even though not all of them are relevant. The goal of the IOU (a value between 0 and 1) is to discard such grid boxes to only keep those that are relevant. Here is the logic behind it:

The user defines its IOU selection threshold, which can be, for instance, 0.5.

Then YOLO computes the IOU of each grid cell which is the Intersection area divided by the Union Area. Finally, it ignores the prediction of the grid cells having an IOU ≤ threshold and considers those with an IOU > threshold.

- **Non-Maximum Suppression**:- Setting a threshold for the IOU is not always enough because an object can have multiple boxes with IOU beyond the threshold, and leaving all those boxes might include noise. Here is where we can use NMS to keep only the boxes with the highest probability score of detection.

Steps involved in Drowsiness Detection:

- **Data Collection**: This phase involves setting up a camera system to monitor a stream for faces in real-time. Cameras can be positioned strategically within the vehicle to capture the driver's face effectively. The collected data, typically video frames, form the basis for training the drowsiness detection system.

- **Data Preprocessing**: Once the data is collected, it undergoes preprocessing to ensure uniformity and compatibility with the training process. This includes resizing images to a standard size, enhancing image quality, and labeling the images or frames with appropriate annotations indicating drowsy or alert states. Preprocessing is crucial for preparing the data for training and ensuring the effectiveness of the model.
- **Training**: The preprocessed data, along with the corresponding annotations, are used to train the drowsiness detection model. This involves feeding the data into the chosen machine learning or deep learning algorithm, such as convolutional neural networks (CNNs), and optimizing the model parameters through iterative training iterations. The goal is to teach the model to accurately recognize patterns indicative of drowsiness in the driver's facial expressions or behavior.
- **Model Evaluation**: After training, the performance of the trained model is evaluated using a separate validation dataset that the model hasn't seen before. This evaluation assesses the model's ability to accurately detect drowsiness in real-world scenarios. Metrics such as accuracy, precision, recall, and F1 score are commonly used to measure the model's performance. The evaluation results help refine the model and fine-tune parameters to improve its effectiveness in detecting drowsiness accurately and reliably.

Reasons that make our algorithm popular for Object Detection :

- **Speed** :- It is extremely fast because it does not deal with complex pipelines. It can process images at 45 Frames Per Second (FPS). In addition, it reaches more than twice the mean Average Precision (mAP) compared to other real-time systems, which makes it a great candidate for real-time processing.
- **Detection accuracy**:- It is far beyond other state-of-the-art models in accuracy with very few background errors.
- **Good generalization** :-It pushed a little further by providing a better generalization for new domains, which makes it great for applications relying on fast and robust object detection.
- **Open-source:**- Making YOLO open-source led the community to constantly improve the model. This is one of the reasons why YOLO has made so many improvements in such a limited time.

Chapter 6

Code Snippets of Project

The end-to-end process of creating a drowsiness detection system using computer vision and deep learning techniques. It includes the steps of data collection, model training, and real-time detection, showcasing the integration of various libraries and tools to build a functional drowsiness monitoring application.

The steps involved are:

1. **Setting up Paths and Labels:-**

IMAGES_PATH: Defines the path where the collected images will be stored (data/images).

number_imgs:- Specifies the number of images to be collected for each drowsiness level.

Install and Import Dependencies

```
In [6]: !pip install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu118

Looking in indexes: https://download.pytorch.org/whl/cu118
Requirement already satisfied: torch in c:\users\hp\anaconda3\lib\site-packages (2.2.2+cu118)
Requirement already satisfied: torchvision in c:\users\hp\anaconda3\lib\site-packages (0.17.2+cu118)
Requirement already satisfied: torchaudio in c:\users\hp\anaconda3\lib\site-packages (2.2.2+cu118)
Requirement already satisfied: filelock in c:\users\hp\anaconda3\lib\site-packages (from torch) (3.6.0)
Requirement already satisfied: sympy in c:\users\hp\anaconda3\lib\site-packages (from torch) (1.10.1)
Requirement already satisfied: networkx in c:\users\hp\anaconda3\lib\site-packages (from torch) (2.8.4)
Requirement already satisfied: jinja2 in c:\users\hp\anaconda3\lib\site-packages (from torch) (2.11.3)
Requirement already satisfied: typing-extensions>=4.8.0 in c:\users\hp\anaconda3\lib\site-packages (from torch) (4.8.0)
Requirement already satisfied: fsspec in c:\users\hp\anaconda3\lib\site-packages (from torch) (2022.7.1)
Requirement already satisfied: numpy in c:\users\hp\anaconda3\lib\site-packages (from torchvision) (1.24.4)
Requirement already satisfied: pillow!=8.3.*,>=5.3.0 in c:\users\hp\anaconda3\lib\site-packages (from torchvision) (10.2.0)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\hp\anaconda3\lib\site-packages (from jinja2->torch) (2.0.1)
Requirement already satisfied: mpmath>=0.19 in c:\users\hp\anaconda3\lib\site-packages (from sympy->torch) (1.2.1)

In [7]: !git clone https://github.com/ultralytics/yolov5

fatal: destination path 'yolov5' already exists and is not an empty directory.

In [8]: !cd yolov5 & pip install -r requirements.txt

Requirement already satisfied: gitpython>=3.1.30 in c:\users\hp\anaconda3\lib\site-packages (from -r requirements.txt (line
5)) (3.1.42)
Requirement already satisfied: matplotlib>=3.3 in c:\users\hp\anaconda3\lib\site-packages (from -r requirements.txt (line
```

Load the Model:-

```
In [9]:
import torch
from matplotlib import pyplot as plt
import numpy as np
import cv2
```

Load Model

```
In [4]:
model=torch.hub.load("ultralytics/yolov5",'yolov5s')
```

```
Using cache found in C:\Users\hp/.cache\torch\hub\ultralytics_yolov5_master
YOLOv5  2024-3-30 Python-3.9.13 torch-2.2.2+cu118 CPU

Fusing layers...
YOLOv5s summary: 213 layers, 7225885 parameters, 0 gradients, 16.4 GFLOPs
Adding AutoShape...
```

```
In [5]:
model
```

```
    (conv): Conv2d(512, 256, kernel_size=(1, 1), stride=(1, 1))
    (act): SiLU(inplace=True)
  )
  (cv3): Conv(
    (conv): Conv2d(512, 512, kernel_size=(1, 1), stride=(1, 1))
    (act): SiLU(inplace=True)
  )
  (m): Sequential(
    (0): Bottleneck(
```

Training the Model:- Specifies the training parameters, such as image size, batch size, number of epochs, and the dataset configuration file.

Train from Scratch

```
In [16]:
import uuid #unique identifier
import os
import time
```

```
In [17]:
IMAGES_PATH=os.path.join('data','images') #/data/images
labels=["fully_awake","slightly_awake","moderately_awake","mildly_drowsy","partially_drowsy","heavily_drowsy","fully_drow
number_imgs=5
```

```
In [18]:
cap=cv2.VideoCapture(0)
#Loop through labels
for label in labels:
    print('Collecting images for {}'.format(label))
    time.sleep(5)

    #Loop through image range
    for img_num in range(number_imgs):
        print('Collecting images for {}, image number {}'.format(label,img_num))

        #Webcam feed
        ret, frame = cap.read()

        #Naming out image path
        imgname=os.path.join(IMAGES_PATH, label+'.'+str(uuid.uuid1())+'.jpg')
```

Collecting Images:- For each label, it collects the specified number of images, capturing frames from the webcam and saving them with unique filenames in the IMAGES_PATH directory.Displays the captured frames in a window and adds a 2-second delay between captures.

```
        #Webcam feed
        ret, frame = cap.read()

        #Naming out image path
        imgname=os.path.join(IMAGES_PATH, label+'.'+str(uuid.uuid1())+'.jpg')

        #Writes out image to file
        cv2.imwrite(imgname,frame)

        #Render to the screen
        cv2.imshow('Image Collection',frame)

        #2 second delay between captures
        time.sleep(2)

        if cv2.waitKey(10) & 0xFF == ord('q'):
            break
cap.release()
cv2.destroyAllWindows()
```

```
Collecting images for mildly_drowsy, image number 4
Collecting images for partially_drowsy
Collecting images for partially_drowsy, image number 0
Collecting images for partially_drowsy, image number 1
Collecting images for partially_drowsy, image number 2
Collecting images for partially_drowsy, image number 3
Collecting images for partially_drowsy, image number 4
Collecting images for heavily_drowsy
Collecting images for heavily_drowsy, image number 0
Collecting images for heavily_drowsy, image number 1
```

Setting up LabelImg:- Clones the LabelImg repository, which is a graphical image annotation tool.Installs the required dependencies (pyqt5 and lxml).

In [1]:

```
!git clone https://github.com/HumanSignal/labelImg
```

```
fatal: destination path 'labelImg' already exists and is not an empty directory.
```

In [2]:

```
!pip install pyqt5 lxml --upgrade
!cd labelImg && pyrcc5 -o libs/resources.py resources.qrc
```

```
Requirement already satisfied: pyqt5 in c:\users\hp\anaconda3\lib\site-packages (5.15.10)
Requirement already satisfied: lxml in c:\users\hp\anaconda3\lib\site-packages (5.2.1)
Requirement already satisfied: PyQt5-sip<13,>=12.13 in c:\users\hp\anaconda3\lib\site-packages (from pyqt5) (12.13.0)
Requirement already satisfied: PyQt5-Qt5>=5.15.2 in c:\users\hp\anaconda3\lib\site-packages (from pyqt5) (5.15.2)
```

In [*]:

```
#training command
!cd yolov5 && python train.py --img 320 --batch 16 --epochs 100 --data dataset.yml --weights yolov5s.pt
```

Training our model on labelled dataset using yolo algorithm:-

Load Custom Model

```
model= torch.hub.load('ultralytics/yolov5','custom',path='yolov5/runs/train/exp4/weights/last.pt')
```

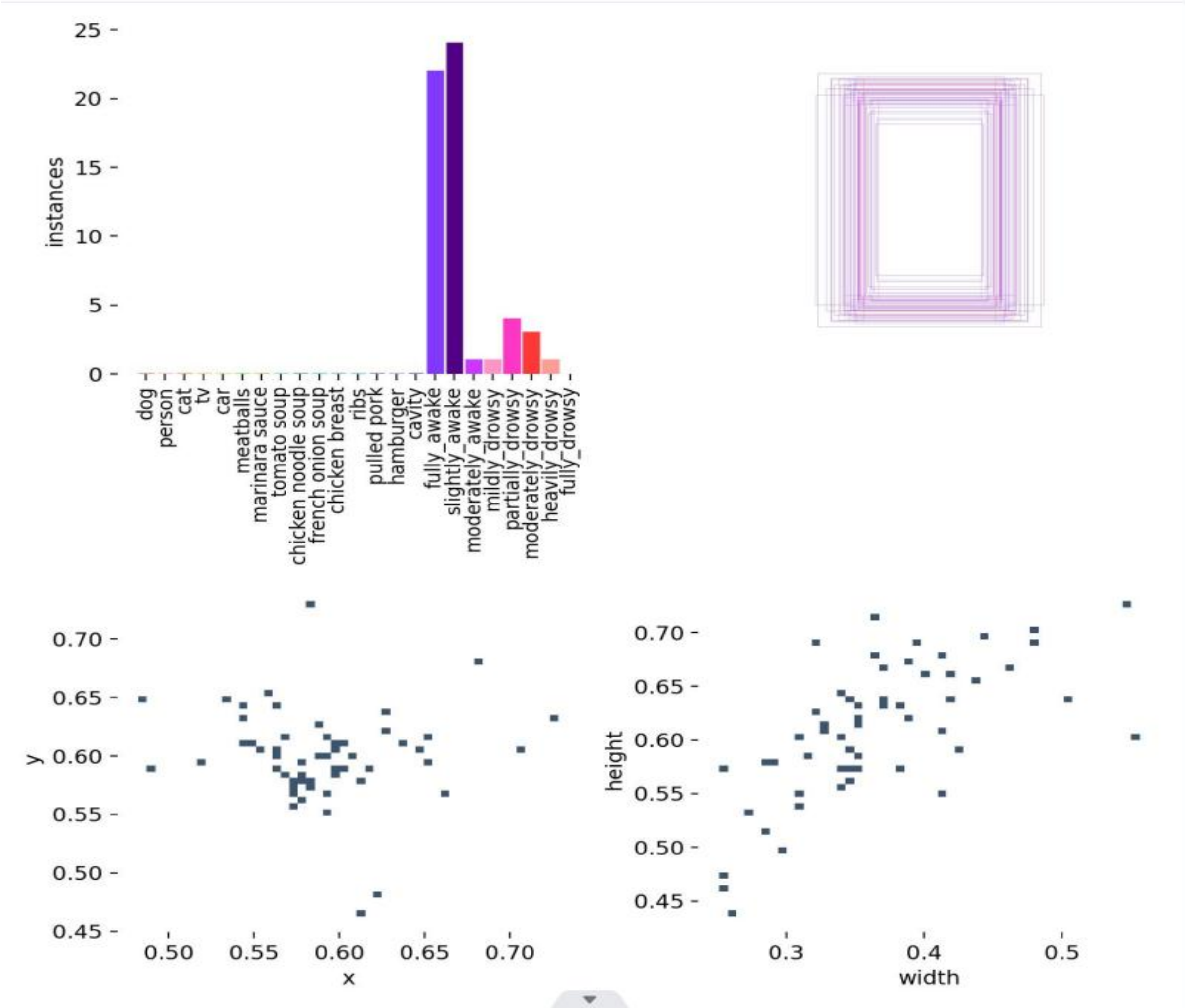
```
cap= cv2.VideoCapture(0)
while cap.isOpened():
    ret,frame=cap.read()

    #Make Detections
    results=model(frame)
    cv2.imshow('YOLO Drowsy Detection',np.squeeze(results.render()))

    if cv2.waitKey(10) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

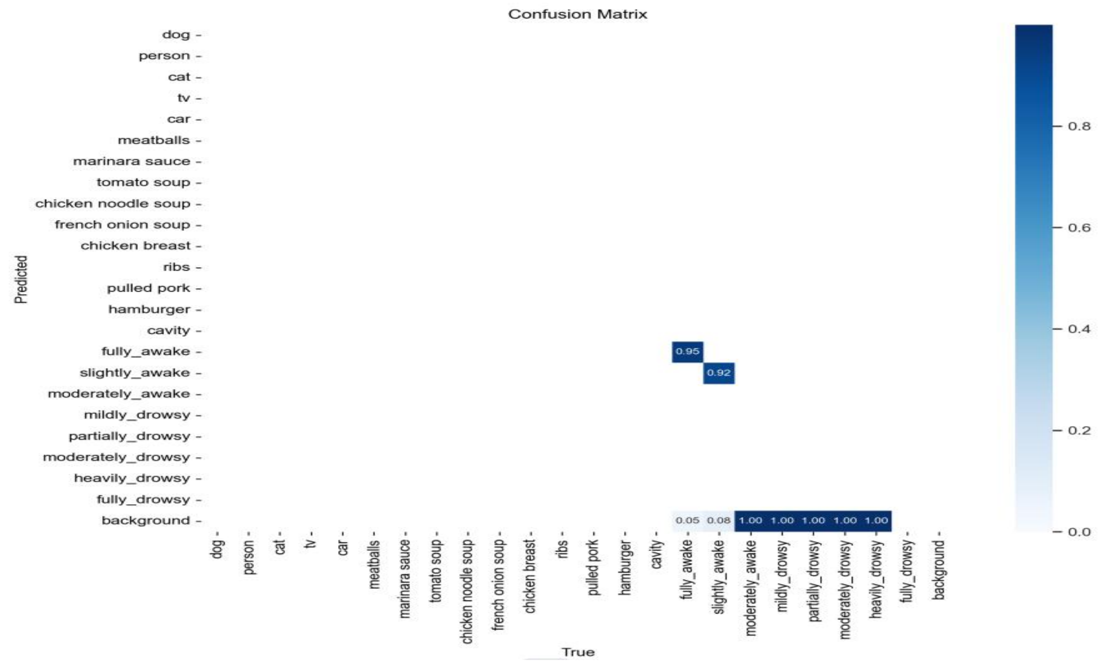
Chapter 7

Screenshots of Project

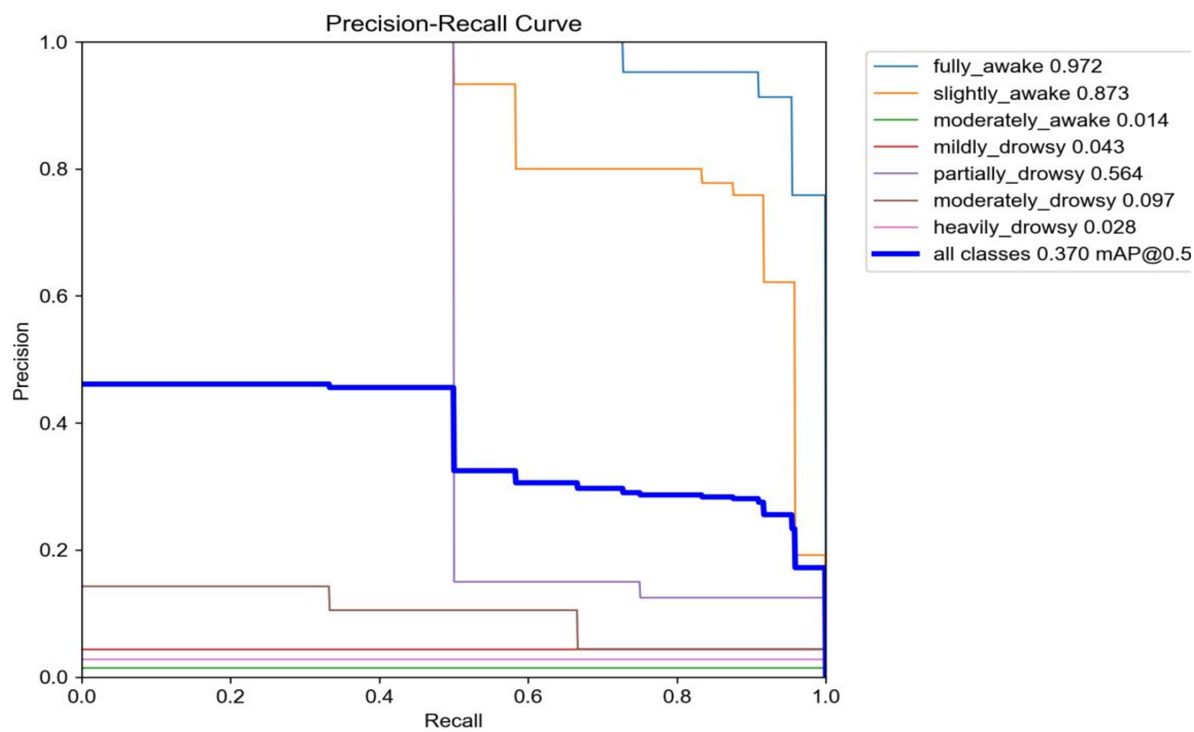


labels: Defines the different drowsiness levels to be detected ("fully_awake", "slightly_awake", "moderately_awake", "mildly_drowsy", "partially_drowsy", "heavily_drowsy", "fully_drowsy").

The below confusion matrix compares the predicted vs actual values for taken eight classes.



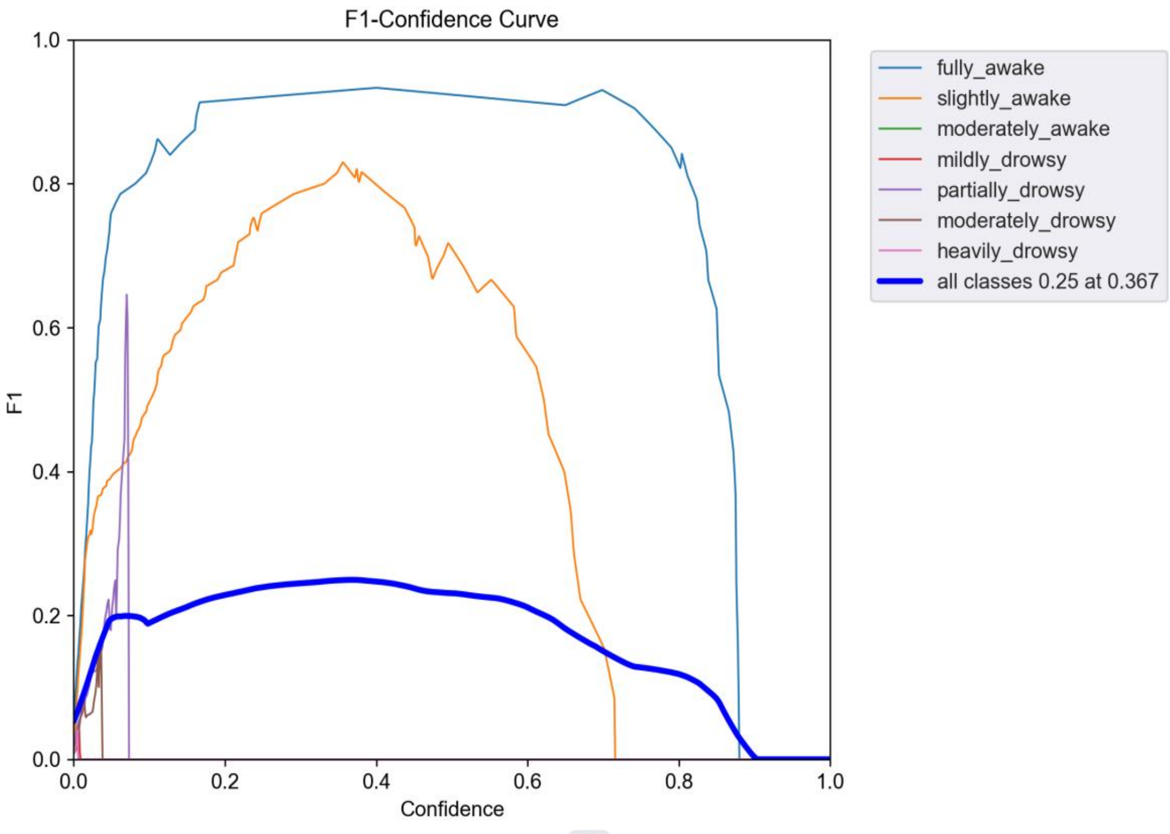
The precision-recall curve shows the trade-off between precision (positive predictive value) and recall (sensitivity) for different thresholds of a classification model.



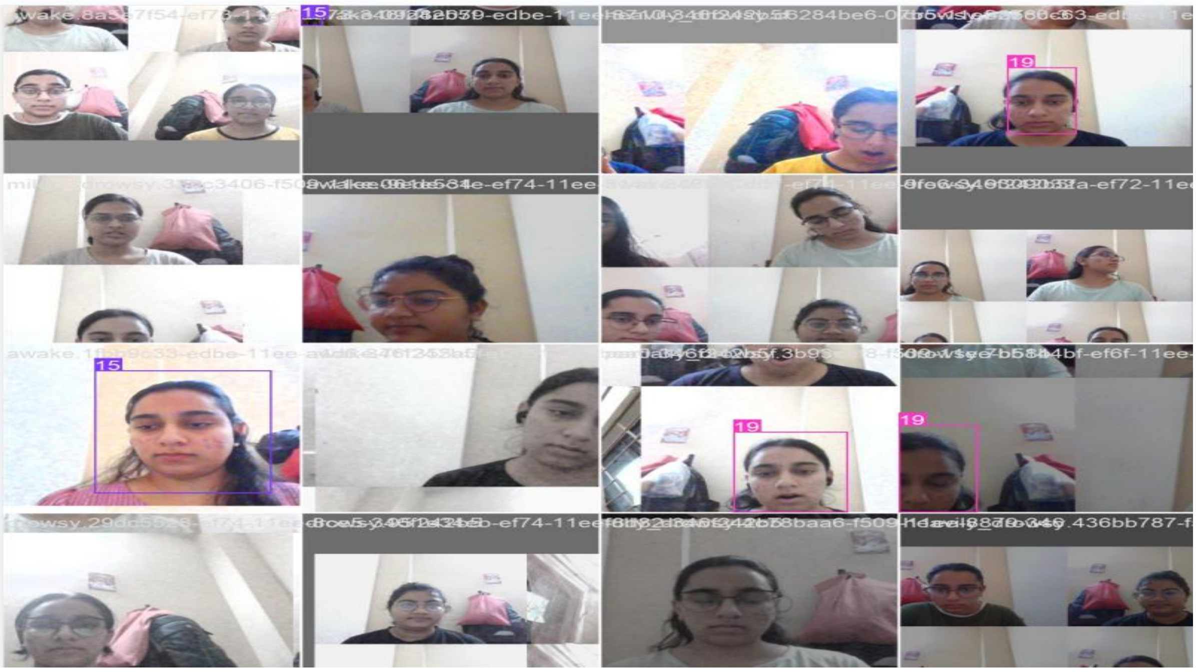
F1-Confidence curve

The curve between the F1 score and confidence level typically indicates the relationship between the model's performance, as measured by the F1 score, and the confidence threshold used for predictions. This curve shows how the F1 score changes as the confidence threshold for accepting predictions varies. It helps in understanding the trade-off between precision and recall at different confidence levels.

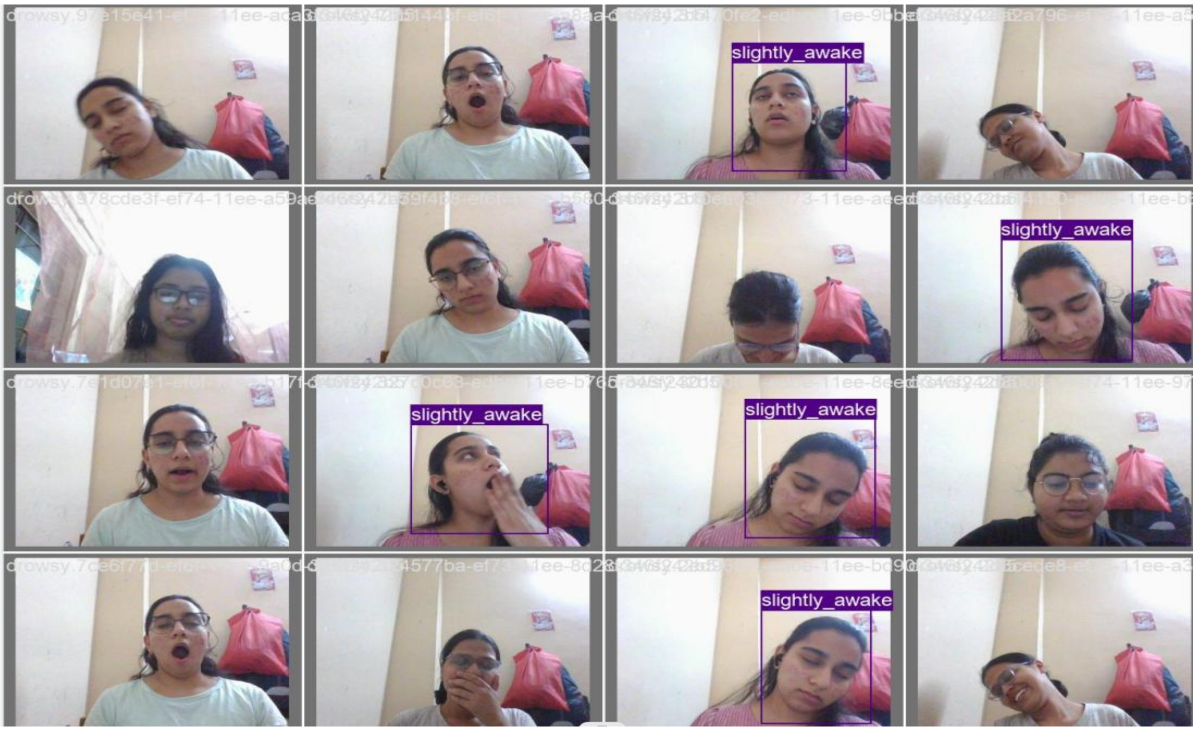
- An increasing slope means the model performance, as measured by F1 score, is improving as the confidence threshold increases.
- A decreasing slope means the model performance is degrading as the confidence threshold increases.



Train Batch



Validation Batch



Batch size of 16 is taken which is show in above figure(separately for train and validation)

Chapter 8

Conclusion and Future scope

8.1 Conclusion

In conclusion, the development of driver drowsiness detection systems holds significant promise for enhancing road safety and mitigating the risks associated with driver fatigue. Through advancements in technology, such as advanced sensing technologies, machine learning algorithms, and intuitive interfaces, these systems have the potential to accurately detect drowsiness in real-time and provide timely alerts to drivers, thus preventing accidents and saving lives. Furthermore, ongoing research and innovation in areas like user experience optimization, data privacy, and personalization will continue to refine and improve the effectiveness of these systems, ensuring a safer and more secure driving experience for all road users. As we look towards the future, it is evident that driver drowsiness detection systems will play a crucial role in promoting safety and awareness on the roads, ultimately contributing to the goal of zero fatalities due to drowsy driving.

8.2 Future Scope

The future scope of driver drowsiness detection systems is promising, with opportunities for advancements in technology, research, and implementation. Here are some potential areas of future development:

1. **Advanced Sensing Technologies:** Integration of infrared cameras, 3D depth sensors, and wearables can enhance data capture accuracy. Infrared cameras can detect subtle changes in facial temperature, while 3D depth sensors provide more detailed information about facial expressions and movements. Wearable devices, such as smartwatches or headsets, can track physiological signals like heart rate and EEG patterns, offering additional insights into the driver's alertness levels.

2. **Machine Learning and AI**: Continued development of deep learning and reinforcement learning techniques can lead to more sophisticated detection algorithms. These algorithms can learn from vast amounts of data to accurately identify drowsiness patterns and adapt in real-time to changing driving conditions, improving overall system performance.

3. **User Experience Optimization**: Designing interfaces and feedback mechanisms for improved usability and effectiveness is crucial. Clear and intuitive interfaces, combined with timely and actionable feedback to the driver (e.g., visual, auditory, or haptic alerts), can enhance the system's usability and acceptance among users.

4. **Data Privacy and Security**: Implementing encryption and privacy-preserving techniques is essential to protect sensitive driver information. As these systems collect and process personal data, ensuring data privacy and security is paramount to maintain trust and compliance with regulations such as GDPR and CCPA.

5. **Intuitive Interfaces**: Designing user-friendly interfaces that provide clear and actionable feedback to the driver can significantly improve the overall user experience. Visual indicators, such as dashboard alerts or heads-up displays, auditory warnings, or haptic feedback through steering wheels or seat vibrations, can effectively communicate drowsiness alerts without distracting the driver.

6. **Personalization**: Allowing for customization and personalization of the system's settings, thresholds, and feedback mechanisms can cater to individual preferences and driving styles. By adapting to each driver's unique characteristics and preferences, the system can provide a more seamless and effective user experience, ultimately enhancing safety on the roads.

References

[1]Yogesh, R., et al. "Driver Drowsiness Detection and Alert System using YOLO." 2022 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES). IEEE, 2022.

[2] OpenCV. (2024). OpenCV: Open Source Computer Vision Library. Retrieved from https://opencv.org

[3] GitHub - ultralytics/yolov5. (2024, April 29). YOLOv5 in PyTorch > ONNX > CoreML > TFLite. Retrieved from https://github.com/ultralytics/yolov5

[4]DataCamp. (2024, May 2). YOLO Object Detection Explained: A Beginner's Guide. Retrieved from https://www.datacamp.com/blog/yolo-object-detection-explained

[5] LabelImg is now part of the Label Studio community. The popular image annotation tool created by Tzutalin , Label Studio the open source data labeling tool for images, text, hypertext. Retrieved from https://github.com/HumanSignal/labelImg

[6] PyTorch. (2024). PyTorch. Retrieved from https://pytorch.org.

[7] Jupyter Notebook. (2024). Jupyter Notebook. Retrieved from https://jupyter.org