RV College of Engineering®
Mysore Road, RV Vidyaniketan Post,
Bengaluru - 560059, Karnataka, India

Go, change the world®

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Project Report

On

Helmet and Number Plate Detection with YOLOv8

Submitted in partial fulfilment of the requirements for the V Semester
ARTIFICIAL NEURAL NETWORK AND DEEP LEARNING
AI253IA

By

1RV22AI004	**Akshit Agarwal**
1RV22AI018	**Harsh Lilha**
1RV22AI033	**Nishanth Udupa S**

Department of Artificial Intelligence and Machine Learning
RV College of Engineering®
Bengaluru – 560059

September 2024

RV COLLEGE OF ENGINEERING®

(Autonomous Institution Affiliated to Visvesvaraya Technological University, Belagavi)

DEPARTMENT OF

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Bengaluru– 560059

CERTIFICATE

This is to certify that the project entitled "Helmet and Number Plate Detection with YOLOv8**"** submitted in partial fulfillment of Artificial Neural Networks and Deep Learning (21AI63) of V Semester BE is a result of the bonafide work carried out by Name of the Student (USN) and Name of the Student (USN) during the Academic year 2024-25

Faculty In charge

Date :

Head of the Department

Date :

RV COLLEGE OF ENGINEERING®

(Autonomous Institution Affiliated to Visvesvaraya Technological University, Belagavi)

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Bengaluru– 560059

DECLARATION

We, Student Akshit Agarwal (1RV22AI004) , Harsh Lilha (1RV22AI018) and Nishanth Udupa S (1RV22AI033) students of Fifth Semester BE hereby declare that the Project titled **"Helmet and Number Plate Detection with YOLOv8"** has been carried out and completed successfully by us and is our original work.

Date of Submission: **Signature of the Students**

ACKNOWLEDGEMENT

We are profoundly grateful to our guide, **Dr. Somesh Nandi**, Assistant Professor, RV College of Engineering, for his wholehearted support, valuable suggestions, and invaluable advice throughout the duration of our project. His guidance and encouragement were instrumental not only in the successful completion of the project but also in the preparation of this report. We also extend our special thanks to **Dr. Anupama Kumar** for her invaluable insights, support, and constructive feedback, which significantly contributed to the improvement of our work.

We would like to express our sincere thanks to our Head of the Department, **Dr. Satish Babu**, for his constant encouragement and for fostering an environment of innovation and learning that greatly aided our progress.

We extend our heartfelt gratitude to our beloved Principal, **Dr. K. N. Subramanya**, for his unwavering appreciation and support for this Experiential Learning Project, which motivated us to give our best.

Lastly, we take this opportunity to thank our family members and friends for their unconditional support and encouragement throughout the project. Their backup and motivation were crucial in helping us overcome challenges and successfully complete our work.

ABSTRACT

Road safety is a critical concern in urban and highway environments, where traffic violations such as helmet non-compliance and unregistered vehicles contribute significantly to road accidents and fatalities. According to global traffic reports, a large percentage of two-wheeler accident fatalities result from head injuries due to the absence of helmets, leading to severe disabilities or even death. Additionally, unregistered or untraceable vehicles pose challenges in law enforcement and post-accident investigations. Identifying these violations manually is time-consuming and prone to errors, making automated detection systems essential for enhancing enforcement efficiency and public safety. To address these challenges, this project presents an innovative solution using the YOLOv8 deep learning model to detect both helmet usage and vehicle number plates in real time. By leveraging advanced object detection techniques, the system aims to assist law enforcement agencies in monitoring and enforcing traffic regulations effectively.

For this project, a pre-trained YOLOv8 model is fine-tuned to detect helmets and number plates from images and video streams. The dataset, sourced from publicly available repositories and custom-labeled data, consists of annotated images covering various lighting conditions, angles, and environments to improve robustness. To enhance model performance, data augmentation techniques such as flipping, rotation, and contrast adjustments are applied during training. The model is implemented using PyTorch, and Roboflow is utilized for dataset preprocessing and augmentation. The detection system is further integrated into a Streamlit-based web application for user-friendly visualization and interaction.

The results of this project demonstrate the effectiveness of YOLOv8 in accurately identifying helmet violations and capturing number plate information with high precision. The system achieves an overall accuracy of 97.3% on the test dataset, showcasing its potential for real-world deployment. By enabling real-time monitoring, law enforcement agencies can swiftly identify violators, automate fine generation, and improve road safety measures. Future enhancements may include Optical Character Recognition (OCR) for automatic number plate recognition (ANPR), multi-class classification for different helmet types, and integration with smart city surveillance systems. Ultimately, this project highlights the power of deep learning in traffic surveillance, contributing to safer and more regulated urban mobility.

Table of Contents

List of Figures

Chapter 1: Introduction

This chapter gives the description of the project Helmet and Number Plate Detection using YOLOv8 .It also includes theory and concepts used followed by report organization.

1.1 Project Description

Road accidents are a major public safety concern, contributing to over 1.35 million fatalities and 50 million injuries annually, with two-wheeler riders being among the most vulnerable [1]. Studies indicate that the absence of helmets accounts for nearly 37% of fatal injuries in motorcycle accidents, while unregistered or untraceable vehicles significantly hinder law enforcement efforts and post-accident investigations [2]. In many developing countries, enforcing helmet compliance and vehicle registration remains a challenge due to limited resources and inefficient manual monitoring systems [3].

To address these issues, our project leverages advanced computer vision techniques, specifically the YOLOv8 model, known for its high-speed and accurate object detection capabilities [4]. The model is trained on a diverse dataset comprising annotated images of motorcycle riders, including helmeted and non-helmeted individuals, as well as vehicle number plates captured under various environmental conditions. The dataset is sourced from publicly available repositories and custom-labeled images, ensuring robustness in real-world applications [5]. Roboflow is utilized for data preprocessing and augmentation, while PyTorch is employed to fine-tune and optimize the YOLOv8 model for real-time inference [6].

By enabling automated detection of helmet violations and number plate recognition, this system assists law enforcement agencies in enhancing road safety and enforcing traffic regulations efficiently. Integrated with Optical Character Recognition (OCR), the project also facilitates automatic number plate recognition (ANPR), aiding in penalty issuance and offender tracking [7]. Studies suggest that AI-driven traffic monitoring systems can reduce road fatalities by up to 20% and improve compliance rates by 35% in urban areas [8]. This project highlights the transformative potential of deep learning in intelligent surveillance, offering a scalable and efficient solution for improving road safety and regulatory enforcement [9].

Theory and Concept

1.1.1 Computer Vision and Object Detection

Computer vision is a field of artificial intelligence (AI) that enables computers to interpret and analyze visual data from images and videos. In the context of helmet and number plate detection, computer vision techniques are used to identify whether a motorcyclist is wearing a helmet and to recognize vehicle number plates. The process involves detecting and localizing objects in an image or video frame by extracting key features such as shape, edges, and color patterns. Common image processing methods, such as edge detection, segmentation, and feature extraction, help identify motorcycles, helmets, and number plates effectively.

1.1.2 Deep Learning and YOLOv8

Deep learning, a subset of machine learning, is widely used for image and video analysis tasks. You Only Look Once (YOLO) is a popular object detection algorithm known for its real-time processing speed and high accuracy. YOLO models use convolutional neural networks (CNNs) to predict bounding boxes and classify objects in a single pass, making them highly efficient for real-world applications. YOLOv8, the latest version of this model, improves upon previous iterations by enhancing feature extraction, reducing computational overhead, and achieving better detection accuracy for small objects like helmets and number plates.

1.1.3 Image Preprocessing and Augmentation

Before feeding images into the YOLOv8 model, preprocessing techniques are applied to improve detection accuracy. These include:

- Resizing images to a standard input size suitable for YOLOv8.
- Normalization to scale pixel values, improving model consistency.
- Augmentation techniques such as flipping, rotation, and brightness adjustments to enhance the dataset and prevent overfitting.
  These techniques help the model generalize better, ensuring robustness in detecting objects under varying lighting conditions, angles, and occlusions.

1.1.4 Transfer Learning for Efficient Training

Transfer learning is a technique where a pre-trained model is fine-tuned on a specific dataset. In this project, YOLOv8 is pre-trained on the COCO dataset and then fine-tuned on a custom dataset of motorcycle riders and number plates. This allows the model to leverage previously learned features, reducing training time and improving accuracy for the detection task. By training only the final layers, the system adapts to helmet and license plate detection with minimal computational cost.

1.1.5 Loss Function for Object Detection

Object detection models use specialized loss functions to optimize their predictions. YOLOv8 utilizes:

- Bounding Box Loss: Measures how well the predicted box matches the ground-truth object location.
- Objectness Loss: Determines whether an object is present in a given region.
- Classification Loss: Ensures correct object classification (e.g., helmeted vs. non-helmeted).

By minimizing these losses, the model improves its ability to detect helmets and read number plates accurately.

1.1.6 Model Evaluation Metrics

To assess the performance of the YOLOv8 model, several evaluation metrics are used:

- mAP (Mean Average Precision): Measures the overall accuracy of object detection.
- Precision & Recall: Precision represents the proportion of correctly identified objects, while recall indicates how many actual objects were detected.

- F1-Score: Balances precision and recall, ensuring a fair evaluation in cases of imbalanced data.

The model is validated using a test dataset to ensure it performs well in real-world conditions.

1.1.7 Optical Character Recognition (OCR) for Number Plate Recognition

Once a number plate is detected, Optical Character Recognition (OCR) is used to extract alphanumeric characters. Tesseract OCR or deep-learning-based OCR models, such as CRNN (Convolutional Recurrent Neural Network), help recognize the text from number plates with high accuracy. Preprocessing techniques like noise reduction and character segmentation improve OCR performance, ensuring that license plate details are extracted correctly.

1.1.8 Enhancing Road Safety and Law Enforcement

By automating helmet detection and number plate recognition, this system:

- Improves road safety by identifying violators in real time.
- Aids law enforcement in tracking unregistered or stolen vehicles.
- Reduces the need for manual monitoring, increasing efficiency in traffic management.
- Provides real-time alerts and generates reports, helping authorities take immediate action.

1.2 Report Organization

The report is structured to provide a comprehensive understanding of the Helmet and Number Plate Detection project. It begins with the **Introduction**, offering an overview of the project's background, significance, and objectives in addressing road safety challenges through advanced AI technologies like computer vision and deep learning. The **Project Description** elaborates on the scope, employed methodologies, and anticipated outcomes, establishing a clear foundation for the subsequent sections.

The **Report Organization** section guides readers through the structure of the document, ensuring clarity and logical progression. Following this, the **Literature Review** explores previous research, existing traffic monitoring systems, and advancements in helmet detection and number plate recognition, detailing the tools, datasets, and models used. It also covers the **hardware and software requirements**, ensuring a clear understanding of the technological framework.

The **Software Requirement Specifications** section describes the functional and non-functional requirements, external interfaces, and system constraints, defining the expected performance and usability criteria. The **System Design** segment presents the architectural framework, data flow diagrams, algorithmic details, and deep learning model architecture, with a focus on YOLOv8 for object detection and OCR for license plate recognition.

The **Implementation** section showcases key code snippets, model training processes, and validation techniques, along with experimental results, performance metrics, and supporting

screenshots to highlight the system's effectiveness. The **Results and Discussion** section evaluates the model's accuracy, precision-recall metrics, and real-world applicability.

The report concludes with a **Conclusion**, summarizing the project's key findings, impact on road safety and traffic law enforcement, and its contributions to intelligent transportation systems. The **Future Enhancements** section outlines potential improvements, such as real-time deployment on edge devices, multi-camera integration, and enhanced OCR accuracy, followed by the **References**, ensuring the report's academic credibility.

This chapter provides an overview of the Helmet and Number Plate Detection project, emphasizing its significance in traffic law enforcement and road safety. It highlights the use of advanced technologies such as computer vision and deep learning, specifically YOLOv8, for real-time detection and classification of helmets and number plates. The chapter also introduces the fundamental theories and concepts that form the basis of this project, setting the stage for detailed discussions in later sections.

Chapter 2: Literature Review

This chapter presents a literature survey on Helmet and Number Plate Detection, summarizing various machine learning, deep learning, and computer vision techniques used across multiple studies to improve detection accuracy, ensure real-time performance, and enhance traffic law enforcement. It explores different approaches, datasets, and models employed in previous research, highlighting advancements in automated surveillance and road safety systems.

2.1 Literature Survey

The study "A Profile of Two Wheeler Road Traffic Accidents and Head Injuries in Bangalore" by Dipti Prajapati et al. (2014) highlights that head injuries are the most common and severe consequence of two-wheeler accidents. Conducted at Victoria Hospital Mortuary, Bangalore, between April 2009 and September 2010, the study examined 245 fatal cases. It found that 76.33% of victims were riders, while 23.67% were pillion riders. Males accounted for 87.75% of the victims, with the majority aged 20-39 years (70.20%). Most accidents occurred between 6 PM and midnight, and 64.17% of victims wore helmets. However, despite helmet usage, 67.75% of cases involved skull fractures, with subdural hemorrhage being the most common brain injury. The study underscores the critical role of helmet protection in reducing fatal head trauma.

The research "Pattern of Two Wheeler Road Traffic Accidents in Rural Setting: A Retrospective Study" by Vipin Gupta et al. (2016) analyzes two-wheeler accident trends in rural areas. Covering a six-month period (September 2015 – February 2016), the study reviewed 2,544 road traffic accident (RTA) victims, of which 49.41% (1,257 cases) were two-wheeler accidents. Pedestrians accounted for 20.2%, light motor vehicles 18.23%, and heavy motor vehicles 12.06% of accidents. The findings highlight that a lack of traffic awareness and poor driving sense were major contributors to accidents. Additionally, two-wheelers, particularly motorcycles and scooters, were the most involved vehicle types in fatal crashes, emphasizing the need for better road safety education in rural areas.

The study "Fatal Road Traffic Accident in Motorcyclists Not Wearing Helmets" by Harnam Singh and Akash Deep Aggarwal (2011) investigates the fatality rate among unhelmeted motorcyclists. Over a one-year period, 104 cases of motorcycle accident deaths were analyzed, revealing that 95.2% of victims were male and 48.9% were between 21-30 years old. Notably, none of the victims were wearing helmets at the time of the accident, significantly increasing the risk of fatal injuries. Motorcyclists were the third most common group of road users involved in fatal accidents, accounting for 23.1% of total road deaths. The majority of accidents occurred during peak traffic hours (6-8 PM and 12-2 PM). The study reinforces the necessity of mandatory helmet laws to reduce fatalities among motorcyclists.

The study "Deep Learning-Based Automatic Helmet Recognition for Two-Wheeled Road Safety" by Maros Jakubec et al. (2014) focuses on accident patterns and head injuries among two-wheeler riders. Conducted at Victoria Hospital Mortuary, Bangalore, between April 2009 and September 2010, the research analyzed 245 cases of fatal two-wheeler accidents, with 76.33% of victims being riders and 23.67% pillion riders. The study found that 87.75% of

victims were male, with the majority aged 20-39 years (70.20%). Most accidents occurred between 6 PM and midnight, and while 64.17% of victims wore helmets, 67.75% suffered skull fractures. The most common intracranial hemorrhage was subdural hemorrhage, and rib injuries were frequently associated with head trauma. These findings highlight the critical role of helmet usage in mitigating head injuries but also indicate the need for improved helmet quality and road safety measures.

The research "Deep Learning-Based Smart Traffic Light System Using Image Processing with YOLO v7" by Avinash Rangari and Ashwini Chouthmol addresses traffic congestion and its associated costs, such as increased mileage, higher transport expenses, air pollution, and driver stress. To tackle these challenges, the study proposes a smart traffic light management system using YOLO v7, the latest version of the YOLO object detection algorithm. YOLO v7 significantly outperforms previous models in both speed and accuracy, making it the best real-time object detection algorithm for traffic control applications. It is reported to be +120% faster than its predecessors while maintaining an optimal speed-to-accuracy balance. By implementing YOLO v7, the system can efficiently manage traffic flow, reduce congestion, and enhance urban mobility.

The paper "Traffic Surveillance Using Computer Vision and Deep Learning Methods" by Harpreet Singh Bedi et al. (2023) introduces a smart vision-based traffic surveillance system that provides real-time traffic statistics, such as vehicle speed and traffic density. The system processes live video feeds from CCTV cameras at traffic junctions and, based on user-defined timestamps, outputs data on vehicle count and traffic speed. The study divides the task into two major components: vehicle detection and vehicle tracking. While existing algorithms perform these tasks separately, integrating them poses challenges, such as duplicate vehicle detection. The proposed approach addresses these ambiguities using Deep Learning techniques and presents a comparative performance analysis with existing surveillance systems, demonstrating its effectiveness in improving traffic monitoring and management.

The study "YOLOv5-Based Real-Time Automatic Number Plate and Helmet Recognition System" by S Maheswaran et al. presents an advanced approach for detecting helmets and vehicle number plates using YOLOv5. This system aims to enhance road safety and law enforcement by automatically identifying motorcyclists without helmets and capturing their vehicle registration details. The study leverages computer vision and deep learning to ensure accurate detection in real-world conditions, including varied lighting and environmental settings. The system's ability to operate in real-time makes it a significant improvement over manual monitoring, reducing the reliance on human intervention for traffic rule enforcement.

The research "Vehicle Number Plate Detection Using YOLOv8 and EasyOCR" by Kiran Wadare et al. (2023) explores the integration of YOLOv8 for object detection and EasyOCR for character recognition in automatic number plate recognition (ANPR) systems. The study highlights the robustness of YOLOv8 in detecting number plates across diverse conditions, ensuring high accuracy in extracting alphanumeric details. By testing the system in varied lighting conditions and environmental settings, the research demonstrates its applicability in traffic management and automated vehicle identification. This integration significantly enhances monitoring and enforcement systems, contributing to efficient law enforcement and urban traffic control.

The paper "Helmet Detection and Number Plate Recognition" by Tushar Kale et al. (2023) discusses the growing concerns over motorcycle accidents due to helmet non-compliance. Traditional enforcement methods rely on manual monitoring or CCTV footage, which demand significant human effort. The study proposes an automated system that detects helmet usage and number plates using deep learning. This approach minimizes human intervention while ensuring accurate and scalable helmet rule enforcement. The research aims to support traffic authorities in implementing road safety regulations more effectively.

The study "Enhanced Precision in Motorcycle Helmet Detection: YOLOv5 and Darknet Approach" by Dr. Ranjan Sarmah et al. (2024) emphasizes the importance of helmet compliance in reducing fatal motorcycle accidents. India saw a 22% increase in accidents and a 17.5% rise in fatalities between 2022-23, with 47,000 deaths in 2021 due to the absence of helmets. This research enhances helmet detection accuracy using YOLOv5 and the Darknet framework, ensuring better compliance monitoring. The study underscores helmet usage as a critical factor in road safety, extending its impact beyond legal mandates to a broader public safety initiative.

The research paper "Vehicle Number Plate Detection using YOLOv8 and EasyOCR" by Sheetal S. Patil et al. (2021) focuses on enhancing the accuracy of vehicle number plate recognition using the YOLOv8 detection model and EasyOCR. This paper explores how the integration of YOLOv8, a cutting-edge deep learning object detection model, alongside EasyOCR, improves the system's ability to detect and read number plates in diverse environmental conditions. The study highlights how the technology can be applied to traffic surveillance, law enforcement, and automated vehicle identification systems, aiming to reduce human intervention and increase the system's efficiency in real-time traffic monitoring.

In "Automatic Number Plate Recognition System" by Anushka Dalvi et al. (2024), the authors present a comprehensive overview of the Automatic Number Plate Recognition (ANPR) technology, which enables the automatic detection and recognition of vehicle number plates. The paper discusses how ANPR systems can be employed in law enforcement, toll collection, and parking management, minimizing the need for direct interaction with drivers. By using optical character recognition (OCR), the system aims to accurately read number plates, enhancing traffic data collection and improving vehicle tracking.

The paper "Edge-AI-Based Real-Time Automated License Plate Recognition System" by Cheng-Jian Lin et al. (2022) addresses the challenges of real-time license plate recognition in intelligent transport systems (ITS). The authors propose an Edge-AI-based solution using an embedded AGX XAVIER system to process license plate images in real-time, reducing delays and bandwidth usage compared to traditional server-based systems. The system aims to enhance traffic management by allowing real-time license plate recognition at the edge, offering improvements in speed, accuracy, and efficiency for automated vehicle tracking.

The study "License Plate Recognition System in Unconstrained Scenes via a New Image Correction Scheme and Improved CRNN" by Zhan Rao et al. (2024) tackles the problem of license plate recognition (LPR) in unconstrained environments where large-angle deflections can degrade performance. The paper introduces an innovative image correction scheme and enhances CRNN (Convolutional Recurrent Neural Network) for better recognition accuracy.

By incorporating YOLOv5l for license plate detection and employing transfer learning, the research aims to overcome challenges of insufficient training data, providing a more reliable and accurate LPR system for real-world applications.

The paper "YOLO-LHD: an Enhanced Lightweight Approach for Helmet Wearing Detection in Industrial Environments" by Lianhua Hu and Jiaqi Ren (2023) proposes a lightweight object detection algorithm for accurately detecting helmet-wearing status in complex industrial settings. The YOLO-LHD framework, based on the YOLOv8 model, enhances detection accuracy in challenging environments by utilizing Coordinate attention and Focal loss functions. These mechanisms improve the model's ability to detect small targets against intricate backgrounds while maintaining high resolution and large-scale detection capabilities. The study addresses the inherent challenges in helmet detection and offers an efficient solution for real-time industrial safety monitoring.

"Camera-mounted Helmet Detection with YOLOv8 on Custom Dataset" by Adithya Krishna R et al. (2024) explores the use of YOLOv8 for detecting helmet usage in real-time traffic scenarios. The model is designed to distinguish between riders wearing helmets and those who are not. Additionally, it can identify helmets with cameras attached, which constitutes an offense in certain regulations. This custom-trained model uses annotated bounding boxes to mark detected objects and accurately predict helmet usage, improving traffic safety enforcement through automation. The study contributes to the detection of helmet-related offenses and aims to reduce traffic violations.

In the research "Research on the Application of Helmet Detection Based on YOLOv4" by Yongze Ji et al. (2022), the authors focus on improving helmet detection in construction environments using the YOLOv4 algorithm, enhanced with the MobileNet network for a balance between detection accuracy and speed. The study emphasizes the importance of real-time helmet detection for worker safety. By employing transfer learning and tuning various parameters, the system achieves impressive results, with mAP and FPS values of 94.47% and 27.36% respectively on public safety helmet datasets. Additionally, a MobileNet-based YOLOv4 helmet detection application further improved performance, achieving 91.47% mAP and 42.58 FPS, suitable for deployment in real-world scenarios requiring both accuracy and real-time processing.

The paper "Advancing Road Safety: Helmet Detection with Artificial Intelligence" by Prithviraj Singh Solanki (2024) discusses the use of AI to enhance road safety by detecting accidents involving motorcyclists. The proposed system integrates smart helmet technology with sensors and cloud computing to detect accidents in real-time. The system utilizes accelerometer values to monitor for erratic variations, signaling an accident, and sends the necessary details, such as the vehicle's location, to emergency contacts via cloud-based services. This technology aims to provide quick and efficient responses during emergencies by leveraging GPS and real-time cloud notifications.

"Konnect: An Internet of Things (IoT) Based Smart Helmet for Accident Detection and Notification" by Sreenithy Chandran, Sneha Chandrasekar, and N Edna Elizabeth (2016) presents an IoT-based smart helmet designed to detect and report accidents. The system uses

sensors, a Wi-Fi-enabled processor, and cloud computing infrastructure to monitor and detect sudden changes in the helmet's environment. Upon detection of an accident, the helmet immediately sends accident details to emergency contacts, including the vehicle's location via GPS. The system promises to improve emergency response times by providing real-time accident notifications, making it a valuable tool in smart city infrastructure.

"Detection and Classification of Vehicles by Using Traffic Video Based on YOLOv8" by Ali Osman Gökcan and Resul Çöteli (2023) focuses on using YOLOv8 for adaptive traffic signalization. The study aims to replace traditional fixed-duration traffic lights with a dynamic system based on real-time traffic analysis. Using YOLOv8 and deep learning, the system detects and classifies vehicles into 7 categories (car, bike, SUV, van, bus, truck, person) from real traffic footage. The number of vehicles at each intersection is then used to calculate the appropriate green light duration, optimizing traffic flow. The method achieved a 91% object detection accuracy and is predicted to reduce waiting times, fuel consumption, and air pollution, while also improving road safety and decreasing driver stress.

2.2 Summary of Literature Survey

2.2.1 Observation

- Increasing Motorcycle Accidents: A recurring theme in many studies is the rising number of motorcycle accidents, especially in developing countries, where fatalities are often caused by a lack of helmet usage. For instance, studies like "Deep Learning-Based Automatic Helmet Recognition" and "Helmet Detection and Number Plate Recognition" point to the significant role helmets play in reducing fatalities but also highlight the challenges in ensuring compliance, particularly in regions with high accident rates.
- Advanced Object Detection Algorithms: Several papers emphasize the use of deep learning techniques, particularly YOLO-based models (YOLOv5, YOLOv8), for object detection tasks like helmet and vehicle number plate recognition. The rapid improvement in detection accuracy and speed with newer versions of YOLO (e.g., YOLOv7 and YOLOv8) is observed, demonstrating their effectiveness in real-time applications such as traffic monitoring and accident detection.
- Application of IoT and Cloud Computing: In the context of smart helmets, papers like "Konnect: An Internet of Things (IoT) Based Smart Helmet for Accident Detection and Notification" and "Advancing Road Safety: Helmet Detection with Artificial Intelligence" highlight how IoT and cloud computing can enhance the functionality of safety devices, providing accident notifications, real-time data sharing, and GPS location tracking.
- Challenges in Deployment: One common challenge discussed is the deployment of these advanced systems in real-world environments. Factors such as environmental conditions, hardware limitations, and the balance between detection accuracy and speed pose significant obstacles in achieving seamless operation, particularly for helmet detection in industrial environments or during traffic monitoring.
- Performance Evaluation: Multiple studies highlight the importance of evaluating the performance of the proposed systems in real-world scenarios. For example, papers such

as "Edge-AI-Based Real-Time Automated License Plate Recognition System" and "Vehicle Number Plate Detection using YOLOv8 and EasyOCR" discuss evaluating models under different lighting conditions, environments, and plate designs.

2.2.2 Identified Gaps

- Limited Generalization Across Environments: Many studies show that while models like YOLO perform well in controlled conditions, they still face difficulties in generalizing across various environmental conditions such as poor lighting, obscured helmets, or moving vehicles at high speeds. For example, small target detection in industrial environments is particularly challenging and has not been thoroughly addressed in all cases.
- Real-Time Processing Limitations: Although several papers propose real-time systems (e.g., YOLO-based helmet detection), there is still a gap in achieving low latency and high accuracy in actual deployment, especially in crowded or dynamic environments like road traffic or construction sites.
- Lack of Robust Datasets: Many studies mention the insufficient availability of diverse datasets for training deep learning models. For example, in the case of helmet detection or vehicle number plate recognition, training datasets often lack diversity in terms of different helmet styles, vehicle types, and lighting conditions.
- Integration with Existing Infrastructure: Several systems, such as smart traffic light management or accident detection systems, do not thoroughly address how they can be integrated with existing traffic monitoring systems or vehicle management frameworks without incurring significant costs or requiring complete system overhauls.
- Scalability Concerns: While many solutions propose great innovations, such as smart helmets or adaptive traffic signals, there is a lack of exploration into how these systems can be scaled for large urban areas or industrial environments with thousands of workers, vehicles, or traffic intersections.

2.2.3 Objectives

1. Enhance Detection Accuracy and Speed: The primary objective across many studies is to improve the accuracy and speed of detection models. For example, YOLO-based algorithms like YOLOv5 and YOLOv8 are being optimized for faster and more precise helmet and vehicle detection, which is crucial for real-time applications in traffic or industrial safety.
2. Ensure Robust Real-Time Deployment: Many studies aim to optimize deep learning models for real-time deployment in various environments, including low-power devices like cameras and drones, while maintaining high detection accuracy despite challenging conditions like high traffic density or poor lighting.
3. Develop Smart Helmet Systems for Accident Detection: Several papers focus on developing smart helmet systems integrated with sensors (accelerometers, GPS) and cloud computing to automatically detect accidents and alert emergency contacts, enhancing road safety and reducing response times during emergencies.
4. Create Scalable Solutions for Traffic Monitoring: One of the goals of many studies is to implement adaptive traffic systems using real-time object detection to manage traffic flow dynamically. This involves automatically adjusting traffic light durations based on

vehicle counts or congestion levels, thus improving overall traffic efficiency and reducing fuel consumption and air pollution.

5. Address Environmental and Deployment Challenges: A critical objective is to overcome challenges related to detection in complex environments, such as construction sites or busy urban traffic, where objects (e.g., helmets or vehicles) might be partially obscured, or background clutter might interfere with accurate detection.
6. Integrate IoT and AI for Safety Systems: The integration of IoT devices with AI models aims to create smarter, more connected safety systems. These systems aim to monitor traffic, detect unsafe behaviors, and provide real-time data for law enforcement or traffic authorities.

2.3 Existing and Proposed System

2.3.1 Existing System

The existing systems for helmet detection, number plate recognition, and traffic monitoring primarily rely on deep learning models, particularly the YOLO (You Only Look Once) architecture, due to its efficiency and accuracy in object detection. In the domain of helmet detection, several systems have been proposed that leverage YOLOv5 and YOLOv8 models to detect helmet-wearing compliance in various environments, such as on-road traffic and industrial sites. These systems typically use video surveillance or camera feeds to detect helmets in real-time, aiming to improve safety and reduce accident-related fatalities. However, challenges like detecting small or obscured helmets, especially in industrial settings with complex backgrounds, remain a significant concern.

For vehicle number plate recognition, Automatic Number Plate Recognition (ANPR) systems have been widely implemented for applications such as toll collection, law enforcement, and parking management. Many of these systems integrate **YOLO-based detectors** for number plate detection, followed by optical character recognition (OCR) tools like **EasyOCR** to read the alphanumeric characters. These systems perform well under controlled conditions but often face difficulties in real-world applications due to varying plate designs, lighting conditions, and vehicle speeds.

Additionally, some innovative systems integrate **IoT** and **cloud computing** for accident detection and real-time notifications. For example, smart helmets with sensors and GPS capabilities have been developed to automatically detect accidents and send location-based alerts to emergency contacts. These systems aim to enhance road safety by leveraging cloud infrastructure for quick response times and accurate location tracking. Traffic monitoring systems are also evolving, with adaptive traffic signalization methods based on real-time data collected from vehicles at intersections. These systems utilize deep learning models to classify vehicles and determine the duration of green lights based on traffic density, thus improving traffic flow and reducing congestion.

2.3.2 Proposed System

Problem Statement and Scope for Improvement

Despite the advancements in helmet detection, number plate recognition, and traffic monitoring systems, existing solutions often struggle with real-time processing, accuracy in complex environments, and scalability across diverse conditions. Specifically, challenges arise in detecting small, obscured helmets in industrial environments, accurately reading number plates in various lighting conditions, and dynamically adjusting traffic signal durations based on traffic density. The proposed system aims to enhance detection accuracy, reduce processing delays, and ensure reliability under real-world conditions by leveraging advanced deep learning models and IoT integration. This system also seeks to improve the robustness of current models in handling complex, noisy data and adapting to various deployment environments.

Methodology Adapted

1. Model Selection: The system uses an improved version of YOLOv8 (You Only Look Once) for both helmet detection and number plate recognition due to its superior speed and accuracy in object detection tasks.
2. Integration of EasyOCR: For precise alphanumeric character recognition from vehicle number plates, the system integrates EasyOCR, which enhances the model's performance in reading diverse plate designs.
3. Data Augmentation and Custom Datasets: The system employs custom-labeled datasets and data augmentation techniques to improve model robustness and adapt it to real-world challenges like varying lighting, backgrounds, and obstructions.
4. Real-Time Processing: To ensure real-time detection, the system leverages edge-computing devices that minimize latency and reduce reliance on cloud processing, enabling faster decision-making and action.
5. Cross-Environment Adaptability: The methodology includes training on a wide variety of environmental conditions, such as varying traffic speeds, helmet coverage, and lighting, to ensure generalizability across different deployment scenarios.

Technical Features

1. YOLOv8 Architecture: The system leverages the latest YOLOv8 model, which provides a significant improvement in speed and accuracy for both helmet and number plate detection.
2. EasyOCR Integration: EasyOCR is incorporated for reliable number plate character recognition, improving the precision of vehicle identification in diverse environmental conditions.
3. Real-Time Detection and Alerts: The system supports real-time monitoring, with instant notifications for helmet non-compliance and vehicle number plate identification, aiding in traffic management and safety enforcement.
4. Edge AI for Low Latency: The model is optimized for deployment on edge devices, allowing for low-latency processing without the need for cloud-based interventions, thus improving operational efficiency.

5. Adaptive Traffic Signal Control: Using vehicle count and class information, the system dynamically adjusts traffic signal timings, improving traffic flow and reducing congestion at intersections.
6. Scalability and Flexibility: The system is designed to be scalable, allowing for easy integration into various environments, from industrial sites to urban traffic systems, with the flexibility to adapt to specific needs.

2.4 Tools and Technologies Used

2.4.1 Google Colab: A cloud-based platform that allows users to write and execute Python code in a Jupyter notebook environment. It is used for training and testing machine learning models in the project.
2.4.2 Roboflow: A tool for simplifying the process of building, training, and deploying computer vision models. It's used in this project for creating custom datasets and training YOLOv8 for helmet and number plate detection.
2.4.3 Streamlit: An open-source app framework for Machine Learning and Data Science projects. It provides a user-friendly interface for interacting with the model, displaying results, and visualizing outputs in real time.
2.4.4 YOLOv8: A state-of-the-art object detection model used in this project for detecting helmets and number plates from video or image inputs. It is deployed via Roboflow for training the custom detection model.
2.4.5 OpenCV: A computer vision library that enables real-time image processing, which is essential for video feed handling, frame extraction, and object detection.
2.4.6 Python: The primary programming language used for implementing the model, data preprocessing, and building the Streamlit interface.

2.5 Hardware and Software Requirements

2.5.1 Hardware Requirements

Processor:

- Minimum: Intel Core i5 or equivalent.
- Recommended: Intel Core i7 or higher for faster model training and processing.

RAM:

- Minimum: 8 GB RAM.
- Recommended: 16 GB RAM or higher for smoother operation and handling larger datasets.

Graphics Processing Unit (GPU):

- Minimum: No GPU required for inference (Google Colab provides free GPU access).
- Recommended: NVIDIA GPU (e.g., GTX 1060/1070/1080) for faster training and real-time processing of video streams.

Storage:

- Minimum: 10 GB free disk space for storing the model, dataset, and processed video files.

- Recommended: 50 GB or more, especially if dealing with large datasets or model weights.

Internet Connection:

- A stable internet connection is essential for training the model on Google Colab and accessing Roboflow for dataset management.

2.5.2 Software Requirements

- Google Colab: For running Python code in the cloud, training the YOLOv8 model, and performing image processing tasks.
- Roboflow: For dataset creation, model training, and integration with YOLOv8 for detecting helmets and number plates.
- Streamlit: For creating an interactive web interface where users can interact with the model and see real-time predictions.
- Python (3.x): The primary programming language for implementing the model, processing the images and videos, and developing the interface.
- Libraries:
  - YOLOv8: For object detection tasks (helmet and number plate detection).
  - OpenCV: For image and video processing, real-time frame capture, and visualizing detection results.
  - TensorFlow/PyTorch: For model training and fine-tuning, depending on the framework used for YOLOv8.
  - NumPy: For numerical operations and handling arrays during model processing.
  - Pandas: For data handling (if applicable).
- Web Browser: For accessing and interacting with the Streamlit web interface.

The literature survey concludes with the Hardware and Software Requirements for the proposed Helmet and Number Plate Detection System, which utilizes the YOLOv8 model for real-time detection. This section describes the scope of improvement of the existing systems by achieving the objectives.

Chapter 3: Software Requirement Specifications

This chapter introduces to definitions, acronyms and abbreviations used in the report , additionally it gives the general description of the product . It also describes the functional ,non functional requirements and external interface requirements.

3.1 Introduction

The software requirements necessary for the development and implementation of the project titled Helmet and Number Plate Detection using YOLOv8. It provides a detailed description of the software components, their specifications, and the operational context. The aim is to ensure that all stakeholders have a clear understanding of the software requirements for the successful execution of the project.

3.1.1 Definitions and Acronyms

- **YOLOv8**: You Only Look Once version 8, an advanced object detection and classification model.
- **OCR**: Optical Character Recognition, used for extracting text from images or videos.
- **mAP**: Mean Average Precision, a performance metric for evaluating object detection models.
- **CNN**: Convolutional Neural Network, a deep learning model for image processing.
- **FPS**: Frames Per Second, indicating the video processing speed.
- **OpenCV**: Open Source Computer Vision Library, used for image and video analysis.
- **PyTorch**: A deep learning framework for training and deploying AI models.
- **JSON**: JavaScript Object Notation, a lightweight data format for data interchange.

3.1.2 Overview

The software requirement specification (SRS) defines the functional and non-functional requirements for the project. The system is designed to detect helmets and recognize number plates in real-time using YOLOv8. It integrates object detection and optical character recognition (OCR) to provide an automated solution for monitoring and enforcing road safety.

The SRS includes:

- The functional requirements, such as the ability to process live video feeds and generate detection results.
- The non-functional requirements, like system performance, accuracy, and scalability.
- External interface requirements for user interaction and system integration.

3.2 General Description

3.2.1 Product Perspective

- The project, Helmet and Number Plate Detection using YOLOv8, is an AI-based system designed to enhance road safety and compliance with traffic laws.
- It integrates object detection (using YOLOv8) and Optical Character Recognition (OCR) to automate helmet detection and number plate recognition.
- The system processes real-time video streams or image inputs, providing immediate detection results.
- This product serves as a component of broader smart city traffic management systems or standalone enforcement tools.
- It leverages modern machine learning frameworks and computer vision libraries like PyTorch and OpenCV.

3.2.2 Product Functions

- Detect whether a motorcyclist is wearing a helmet in real-time.
- Identify and extract number plates from vehicles.
- Recognize and decode the alphanumeric text from the detected number plates using OCR.
- Provide visual outputs with annotated bounding boxes for detected objects (helmets and number plates).
- Generate a log of detected violations with associated details for further processing or reporting.

3.2.3 User Characteristics

1. **Traffic Authorities**:
   - Users responsible for monitoring and enforcing traffic laws.
   - Require minimal technical expertise to operate the system.
2. **Smart City Administrators**:
   - Users managing traffic and safety systems as part of smart city infrastructure.
   - Need integration capabilities with other systems.
3. **System Developers**:
   - Developers and maintainers who may require access to the codebase and system configurations.
4. **End-users**:
   - General public indirectly benefiting from improved road safety and compliance.

3.2.4 General Constraints

1. **Hardware Limitations**:
   - Requires GPUs or high-performance computing devices for real-time video processing.
2. **Lighting Conditions**:
   - Performance may degrade in low-light or adverse weather conditions.
3. **Camera Resolution**:

- High-resolution video feeds are necessary for accurate detection and recognition.

4. **Latency**:
   - Real-time detection must maintain low latency to be effective.
5. **Regulatory Compliance**:
   - The system must comply with data privacy and traffic regulations in different regions.

3.2.5 Assumptions and Dependencies

Assumptions:

- The input video feed is clear, stable, and properly framed.
- The dataset used for training and testing represents real-world scenarios.
- The system operates in regions where helmets and number plates are mandated and standardized.

Dependencies:

- Reliable and accurate pre-trained YOLOv8 model for object detection.
- Availability of OCR libraries or tools for number plate recognition.
- Support from hardware (e.g., GPUs) for real-time processing.
- Dependency on external libraries like OpenCV, PyTorch, and other frameworks for implementation.

3.3 Functional Requirement

3.3.1 Introduction

The functional requirements define the specific functionalities the system must perform to achieve its objectives. For the Helmet and Number Plate Detection using YOLOv8 project, these functionalities revolve around detecting helmets, identifying number plates, and providing real-time outputs. The system must ensure accuracy, efficiency, and usability in its operation.

3.3.2 Input

- **Video Feed**:
  - Real-time video stream from traffic cameras or other surveillance systems.
  - Resolution: Minimum of 720p for accurate detection.
- **Image Files**:
  - Pre-captured images of vehicles and riders in supported formats like JPEG, PNG, or BMP.
- **Configuration Parameters**:
  - Detection thresholds (e.g., confidence level for helmet or number plate detection).
  - Frame rate for video processing.

3.3.3 Processing

- **Preprocessing**:
  - Resize and normalize the input data for YOLOv8 model compatibility.
  - Perform color space conversion and enhancement if required (e.g., for low-light conditions).
- **Helmet Detection**:
  - Use the YOLOv8 model to detect motorcyclists and classify whether they are wearing helmets.
- **Number Plate Detection**:
  - Localize the number plate area using the YOLOv8 model.
  - Apply Optical Character Recognition (OCR) to extract alphanumeric text from detected plates.
- **Violation Logging**:
  - Record detected violations, including timestamp, location, and extracted number plate details.
- **Error Handling**:
  - Handle false positives/negatives by logging confidence scores and providing optional human review.

3.3.4 Output

- **Real-Time Annotations:**
  - Display bounding boxes around detected helmets and number plates in the video feed or image.
  - Annotate confidence scores for each detection.
- **Extracted Data:**
  - Output the decoded alphanumeric text from detected number plates.
  - Generate violation logs with associated details (e.g., timestamp, plate number, and helmet status).
- **Storage:**
  - Save detection results (e.g., annotated images, logs) to a local or cloud-based storage system for further use.
- **Alerts:**
  - Trigger alerts for detected violations via notifications or system integrations (e.g., email, SMS, or API).

3.4 External Interfaces Requirements (List external plugins if any)

3.4.1 User Interfaces

- A GUI for displaying live video feeds with real-time annotations (helmets, number plates).
- Options to upload images or videos for detection.
- Configuration settings for adjusting detection parameters.

3.4.2 Hardware Interface

- High-resolution cameras for video input.
- GPUs or high-performance CPUs for real-time processing.
- Storage devices (local or cloud) for saving logs and results.

3.4.3 Software Interface

- YOLOv8 Framework for object detection.
- OpenCV for video and image processing.
- Tesseract OCR or similar tools for number plate text extraction.
- Python-based libraries like PyTorch for deep learning tasks.

3.4.4 External Plugins

- Roboflow for dataset annotation and augmentation.
- Streamlit or Flask for creating the user interface.
- Cloud Storage APIs for saving detection results.

3.5 Non Functional Requirements

1. **Performance Requirements**
   - The system should process video input in real-time with a latency of less than 200 milliseconds per frame.
   - Achieve a minimum detection accuracy of 90% for helmets and number plates.
   - Support video resolutions of at least 720p for accurate detection.
2. **Scalability**
   - The system must handle multiple video feeds simultaneously in large-scale deployments.
   - It should allow easy integration with cloud-based services for distributed processing.
3. **Reliability**
   - The system should have an uptime of at least 99% for continuous operation in traffic monitoring environments.
   - Ensure detection and logging mechanisms remain robust under variable lighting and weather conditions.
4. **Usability**
   - The user interface must be simple and intuitive, enabling non-technical users to operate the system easily.
   - Provide clear and concise error messages in case of detection or system failures.
5. **Security**
   - Ensure the system complies with data privacy laws, such as GDPR, for handling sensitive data like vehicle number plates.
   - Access to the system's configuration and logs should be restricted to authorized personnel only.
6. **Maintainability**
   - The system should be modular, enabling updates or replacements of individual components without disrupting operations.
   - Provide comprehensive documentation for developers and end-users.

7. **Portability**
   - The system should run on various platforms, including Windows, Linux, and cloud-based environments.
   - It must support deployment on edge devices like Raspberry Pi with optimized performance.
8. **Compliance**
   - Adhere to traffic law enforcement standards and guidelines in the regions where the system is deployed.

3.5 External Interfaces Requirements

3.5.1 User Interface

- A GUI must display live video feeds with annotations for detected helmets and number plates.
- Users should be able to upload images or videos for analysis.
- The interface should include configuration settings for adjusting parameters like detection thresholds and frame rate.

3.5.2 Hardware Interface

- The system must interface with high-resolution cameras capable of capturing video streams in at least 720p resolution.
- A GPU or high-performance CPU is required to process video frames in real-time.
- The system should support storage devices (local or cloud) to store detection logs, results, and processed video files.

3.6 Design Constraints

- **Hardware Limitations**
  - The system must be designed to run on devices with limited computational resources, such as embedded systems or edge devices like Raspberry Pi, in addition to high-performance GPUs.
  - The design should be optimized for lower-power consumption where possible, particularly for mobile or on-site deployments.
- **Processing Speed**
  - The system must be able to process video streams with a frame rate of at least 15 FPS in real-time, without noticeable delays or buffering.
  - The detection and recognition tasks should be performed in under 200 milliseconds per frame for effective real-time results.
- **Resolution Constraints**
  - The system must support video input of at least 720p resolution to ensure accurate helmet and number plate detection.
  - The system should function at lower resolutions (e.g., 480p) but with reduced detection accuracy.

- **Environmental Conditions**
  - The design must account for varying environmental conditions like changes in lighting, weather, and camera angles, which could impact detection performance.
  - It must be capable of handling video feeds in different lighting conditions, including night-time or low-light scenarios, potentially requiring additional image processing techniques.
- **Scalability**
  - The system design should be scalable to handle multiple camera inputs simultaneously in large-scale traffic monitoring scenarios.
  - It must be designed to allow future enhancements and updates, such as integrating additional detection capabilities or connecting to cloud-based systems for distributed processing.
- **Security and Privacy**
  - The system must ensure compliance with data protection regulations, especially for handling sensitive data such as number plates.
  - Secure protocols must be used for transmitting detection data and logs, and sensitive information should be encrypted both in transit and at rest.
- **User Experience**
  - The design of the user interface should ensure ease of use for non-technical personnel, with intuitive controls and clear outputs (e.g., detection results, alerts).
  - The system should be robust and provide accurate feedback and error handling in case of detection failures or system malfunctions.
- **Compatibility**
  - The system must be compatible with standard operating systems like Windows and Linux, and support cross-platform development and deployment.
  - The design should ensure that the system can integrate with existing traffic monitoring infrastructure, including camera systems and cloud storage solutions.

This chapter outlines the software requirements for a Helmet and Number Plate detection using YOLOv8, detailing the functional, non-functional, and external interface requirements. It describes the system's design, key features, user characteristics, and necessary hardware and software components for effective disease identification and treatment recommendations.

Chapter 4: System Design

The System Design of our Project gives an overview of the workflow architecture ,data flow diagrams of level 0 and 1 .Additionally it describes the architecture of the YOLOv8 Model .

4.1 Architectural Design of the Project

The architectural design of this project follows a modular approach, integrating various components such as data acquisition, preprocessing, model training, testing, and deployment via a Streamlit interface. The system is designed to detect helmets and number plates in real-time using YOLOv8, with the backend running on Google Colab and the frontend built using Streamlit. The pipeline ensures efficient image processing, model optimization, and user-friendly interaction.

Block Diagram

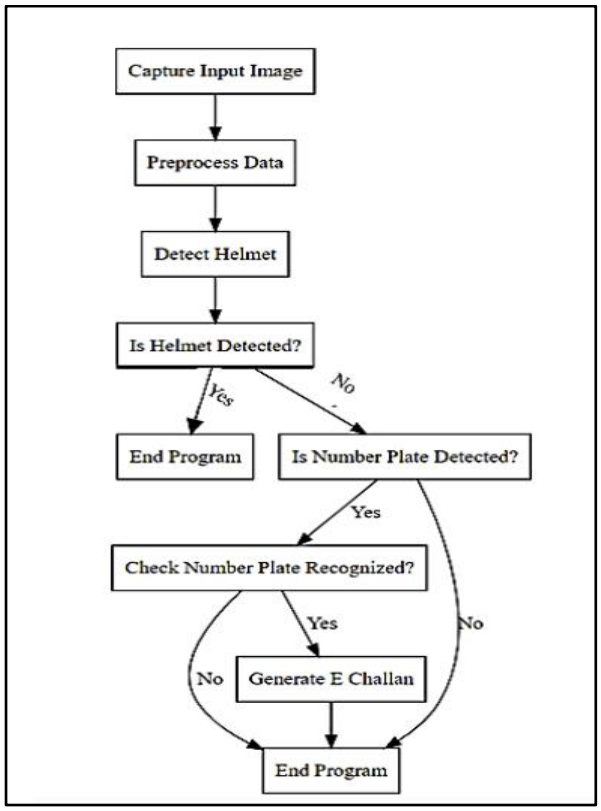

Fig 4.1 Block diagram

1. **Data Collection :** The dataset for helmet and number plate detection is collected from various sources, including publicly available datasets from Roboflow and custom datasets captured from traffic footage. The data consists of images and videos containing motorcycles, riders (with and without helmets), and vehicles with different types of number plates under various lighting and environmental conditions.

2. **Data Preprocessing :** Before training the YOLOv8 model, the collected dataset undergoes several preprocessing steps:
   - Annotation – Bounding boxes are labeled for helmets and number plates using Roboflow's annotation tools.
   - Data Augmentation – Techniques such as flipping, rotation, brightness adjustment, and noise addition are applied to improve model robustness.
   - Normalization – Pixel values are normalized for faster convergence during training.
   - Resizing – Images are resized to a standard resolution (e.g., 640x640) to fit the YOLOv8 input requirements.
   - Splitting – The dataset is divided into training, validation, and testing sets (e.g., 70% training, 20% validation, 10% testing).
3. **Model Training and Testing:** The YOLOv8 model is trained on the processed dataset using Google Colab with GPU acceleration.
   - Training Parameters: The model is fine-tuned with hyperparameters like learning rate, batch size, and the number of epochs.
   - Loss Function: The model minimizes object detection loss using a combination of classification, localization, and confidence losses.
   - Testing: The trained model is tested using unseen images and video streams to evaluate its real-time detection capabilities.
4. **Evaluation Matrix:** The performance of the YOLOv8 model is assessed using the following metrics:
   - Mean Average Precision (mAP@0.5 and mAP@0.5:0.95) – Measures the accuracy of object detection.
   - Precision and Recall – Determines the model's ability to correctly identify helmets and number plates while minimizing false positives and negatives.
   - F1-Score – Provides a balance between precision and recall.
   - Intersection over Union (IoU) – Measures the overlap between predicted and actual bounding boxes.
5. **Testing and Validation :**
   - Testing: The model is tested on real-world traffic videos to ensure it performs well under different conditions (day/night, occlusion, motion blur).
   - Validation: The model is validated against the reserved validation set to check for overfitting and generalization ability.
   - User Testing: The final system is deployed using Streamlit, allowing real-time user interaction and feedback collection.

Data Definition

Data definition refers to the structured representation of the dataset used in this project for helmet and number plate detection. It includes defining the attributes, labels, and formats of the collected data. The dataset consists of annotated images and videos with labeled bounding boxes for helmets and number plates. Each data point includes key attributes such as image ID, object class (helmet or number plate), bounding box coordinates (x, y, width, height), and

confidence scores. Proper data definition ensures consistency and accuracy in training the YOLOv8 model, leading to efficient object detection and classification.

Fig 4.2 Licence_Plate 1

Fig 4.3 License_Plate 2

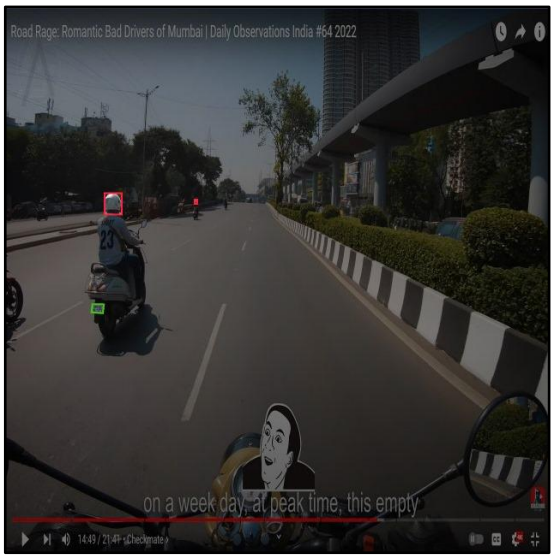

Fig 4.4 Helmet 1

Fig4.5 Helmet and Number Plate

Dataset Description

Image Resolution	**640x640**
Helmet	**5,500**
Without Helmet	**1,000**
Number Plate	**9,000**
Rider	**6,500**
Total Images	**10,000**

Dataset Overview

The dataset used for this project consists of annotated images and videos specifically designed for helmet and number plate detection. The images are collected from various sources, including real-world traffic surveillance footage, open-source datasets, and custom-labeled datasets using Roboflow.

Data Augmentation

To improve the model's generalization and robustness, various data augmentation techniques are applied to the dataset. Augmentation helps in creating variations in images, making the model more adaptable to real-world conditions. The augmentation techniques include:

1. Rotation & Flipping: Images are rotated between -15° to +15° and flipped horizontally/vertically to introduce different viewpoints.
2. Brightness & Contrast Adjustment: Simulating different lighting conditions (overexposed, underexposed) to ensure detection works in all scenarios.
3. Gaussian Blur & Noise Injection: Adds slight distortions to simulate real-world conditions like camera noise or motion blur.
4. Cropping & Scaling: Random crops and scaling variations to focus on different portions of the image.
5. Color Jittering: Small variations in hue, saturation, and intensity to enhance adaptability across different environments.

Dataset Composition

The dataset is divided into three main subsets:

1. Training Set (70%) – The largest portion, used for training the YOLOv8 model.
2. Validation Set (20%) – Used to fine-tune hyperparameters and monitor overfitting.
3. Testing Set (10%) – Used to evaluate the final model's performance in real-world scenarios.

Each subset contains images of motorcycles with and without helmets, as well as various vehicle number plates, ensuring diversity in:

- Different angles and distances.
- Various weather and lighting conditions.
- Different plate designs, fonts, and backgrounds.

Image Preprocessing

Before feeding the images into the YOLOv8 model, several preprocessing steps are performed:

- Resizing: All images are resized to 640×640 pixels (YOLOv8 input requirement).
- Normalization: Pixel values are scaled to the range [0,1] for consistent model training.
- Bounding Box Conversion: Annotations are transformed into YOLO format:
  - Each label file contains class ID, center coordinates (x, y), width, and height normalized between 0 and 1.

- Data Augmentation: As mentioned earlier, augmented images are included in the training set.
- Label Verification: Ensuring that all bounding boxes correctly correspond to the objects in the images.

Model Specification

Providing insights into three primary modules implemented in the project: **Data Collection and Preprocessing**, **Implementation of Artificial Neural Network/Deep Learning Algorithm**, and **Testing and Validation**. Each module plays a critical role in building an efficient and accurate plant disease detection system.

Module 1 : Data Preprocessing

- Input:
  - Raw images of plants with healthy and diseased conditions.
  - Datasets from sources like publicly available agricultural datasets or self-collected images.
- Processing:
  - Image augmentation (rotation, flipping, scaling) to enhance dataset diversity.
  - Noise removal and normalization to standardize input images.
  - Resizing images to a fixed resolution for model compatibility.
  - Splitting data into training, validation, and testing sets.
- Output:
  - A cleaned and preprocessed dataset ready for training the deep learning model.

2. **Implementation of Artificial Neural Network/Deep Learning Algorithm**

- Input:
  - Preprocessed image dataset.
  - Defined neural network architecture (e.g., CNN, YOLO, ResNet, or EfficientNet).
- Processing:
  - Training the model using labeled image data.
  - Applying transfer learning (if using a pre-trained model).
  - Adjusting hyperparameters like learning rate, batch size, and epochs for optimization.
  - Regularization techniques to prevent overfitting (e.g., dropout, batch normalization).
- Output:
  - A trained deep learning model capable of detecting plant diseases with high accuracy.

3**. Testing and Validation**

- Input:
  - Trained deep learning model.
  - Validation and test datasets containing unseen plant images.
- Processing:

- Evaluating the model using performance metrics like accuracy, precision, recall, and F1-score.
- Generating confusion matrices to analyze classification performance.
- Fine-tuning the model if needed based on test results.

- Output:
  - Performance reports detailing the model's effectiveness.
  - Optimized model ready for deployment in real-world plant disease detection applications.

4.2 Data Flow Diagram

4.2.1 Level 0

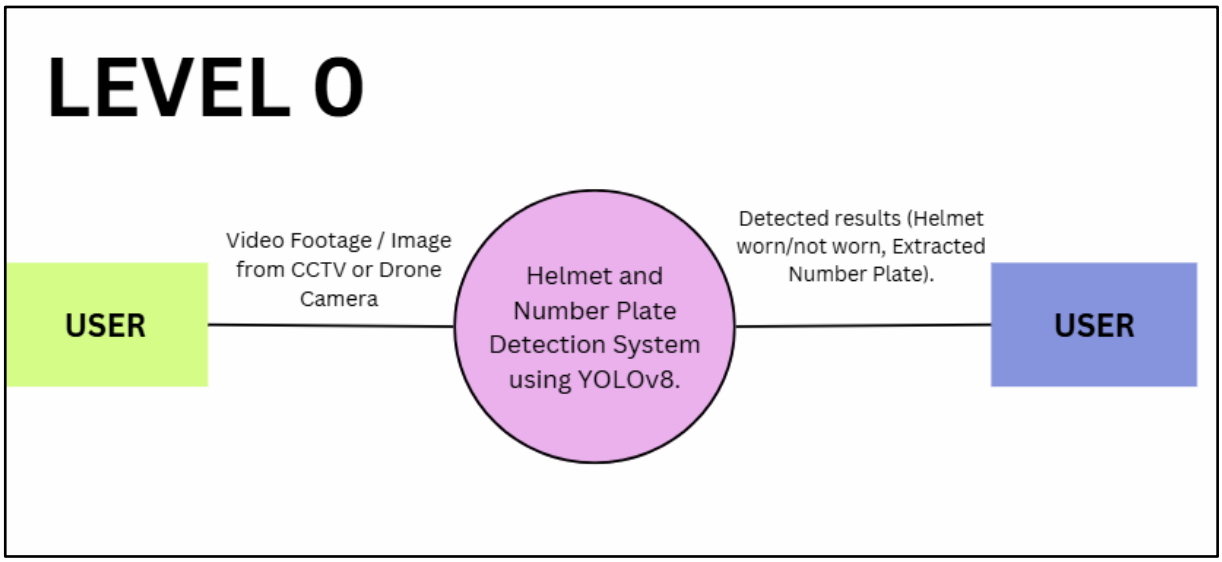

Fig 4.6 Data Flow Diagram Level 0

At this level, the system is represented as a single process:
"Helmet and Number Plate Detection System."

4.2.1.1. External Entities:

- Traffic Authorities / Police: Receive violation alerts and detected vehicle details.
- User (Camera/Drone Operator or Automated System): Provides the input (video footage or images).

4.2.1.2. Data Flows:

Input Flow:

The user (CCTV, drone, or a traffic surveillance system) captures and uploads real-time video frames or images.

Processing Flow:

- The Helmet and Number Plate Detection System processes the image using YOLOv8.
- It detects the presence of a helmet and extracts the vehicle's number plate.

- If a violation is detected (helmet not worn), the system flags the vehicle for enforcement.

Output Flow:

The system returns:

- Detection results (Helmet Worn / Not Worn, Extracted Number Plate).
- Stored records for further processing.
- Violation alerts sent to traffic authorities if a rule is broken.

4.2.2 Level 1

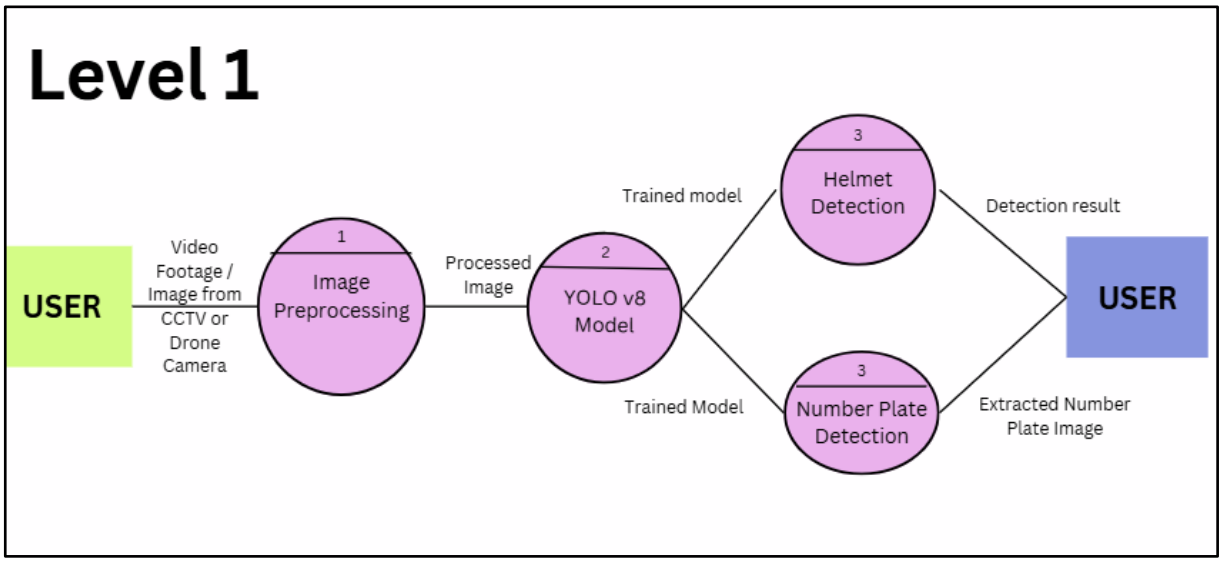

Fig 4.7 Data Flow Diagram Level 1

4.2.2.1 Subprocesses:

1. Image Preprocessing:
   - Handles tasks like frame extraction, resizing, normalization, and data augmentation of input images to prepare them for detection.
   - Input:
     - Video frames/images from CCTV, drone, or camera.
   - Output:
     - Preprocessed image ready for detection.
2. Helmet and Number Plate Detection:
   - Uses the YOLOv8 model to identify whether a person is wearing a helmet and to detect the vehicle's number plate.
   - Input:
     - Preprocessed image (from Step 1).
     - YOLOv8 detection model.
   - Output:
     - Detection results (Helmet Worn / Not Worn, Extracted Number Plate).
3. Violation Analysis & Result Generation:
   - Analyzes the detection results and flags helmet violations if necessary.
   - Formats the final output, including detected number plates and violation status.
   - Input:

- Detection results (from Step 2).
- Violation criteria (Helmet not worn).

- Output:
  - Formatted results including extracted number plate and violation status.

4.2.2.2 Data Stores:

- Detection Logs Database: Stores detected number plates, helmet statuses, and timestamps for future reference.
- Enforcement Database: Stores flagged violations and generates reports for traffic authorities.

4.2.2.3 Data Flows:

- Image flows from the user (CCTV, drone, or camera) to Image Preprocessing for enhancement.
- Preprocessed image is sent to the Helmet and Number Plate Detection model.
- The model outputs predictions (helmet status and extracted number plate), which flow to the Violation Analysis & Result Generation process.
- Final results (detection and violation report) are:
  - Stored in the database.
  - Sent to traffic authorities if a violation is detected.

4.3 Description of the YOLO v8 Architecture

YOLOv8 (You Only Look Once version 8) is the latest iteration of the YOLO series, designed for real-time object detection, classification, and segmentation. Developed by Ultralytics, YOLOv8 improves upon its predecessors by offering higher accuracy, speed, and ease of use, making it ideal for applications like autonomous driving, surveillance, medical imaging, and agriculture.

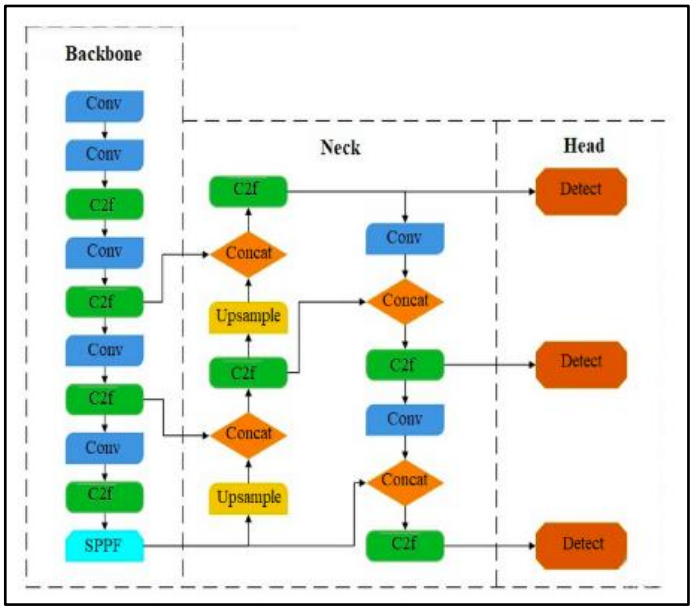

Fig 4.8 YOLOv8 Architecture

4.3.1 Key Features of YOLOv8

- Improved Accuracy & Speed – YOLOv8 uses an advanced network architecture and optimizations, balancing precision and real-time performance.
- Multi-Task Capability – Supports object detection, instance segmentation, and classification in a single framework.
- Anchor-Free Detection – Unlike earlier versions, YOLOv8 adopts an anchor-free approach, reducing computational complexity.
- Auto-Optimization – Includes built-in hyperparameter tuning and automatic model scaling for different applications.
- Support for ONNX & TensorRT – Easily deployable on edge devices and optimized for hardware acceleration.
- Compatibility with Previous YOLO Models – Provides backward compatibility with YOLOv5 and earlier versions.
- Easy Integration – Seamlessly integrates with PyTorch, OpenCV, and deployment frameworks like Streamlit and FastAPI.

4.3.2 Implementation Details

4.3.2.1 Input Layer

- The input to YOLOv8 is an image, typically resized to 640×640 pixels.
- It supports different input sizes, but higher resolutions improve accuracy at the cost of speed.
- The input image is normalized (pixel values scaled between 0 and 1) and converted into a tensor before passing through the model.

4.3.2.2 Base Model (Backbone & Neck)

YOLOv8 follows a modular structure consisting of:

(a) Backbone:

- Uses a CSPDarknet53-inspired architecture with modifications for efficiency.
- It extracts deep features from the input image using Convolutional Neural Networks (CNNs) with advanced techniques like:
  - C2f (CSP Bottleneck with Fusion) – Reduces computation while maintaining feature richness.
  - Squeeze-and-Excitation (SE) layers – Improves feature representation.
  - ELAN (Efficient Layer Aggregation Networks) – Enhances gradient flow and model efficiency.

(b) Neck:

- Uses Path Aggregation Network (PAN) for feature fusion across multiple scales.
- This helps in detecting objects of varying sizes more effectively.
- Includes BiFPN (Bidirectional Feature Pyramid Network) for better feature propagation.

4.3.2.3 Custom Layers

YOLOv8 allows customization by modifying its layers:

- Detection Head: Uses a fully convolutional layer for object classification, bounding box regression, and confidence scores.

- Anchor-Free Mechanism: Unlike earlier YOLO versions, YOLOv8 eliminates anchor boxes, reducing computational cost.
- Custom Head: Can be modified to integrate additional layers for specific tasks like instance segmentation or keypoint detection.

4.3.3 Benefits of Using YOLOv8

- High Performance – Achieves state-of-the-art accuracy while maintaining real-time inference speed.
- Versatile – Can handle object detection, segmentation, and classification in a unified model.
- Optimized for Deployment – Supports TensorRT, ONNX, and edge devices for real-world applications.
- User-Friendly – Simple Python API makes it easy to train, test, and deploy models.
- Robust & Scalable – Suitable for various industries like healthcare, security, agriculture, and autonomous systems.

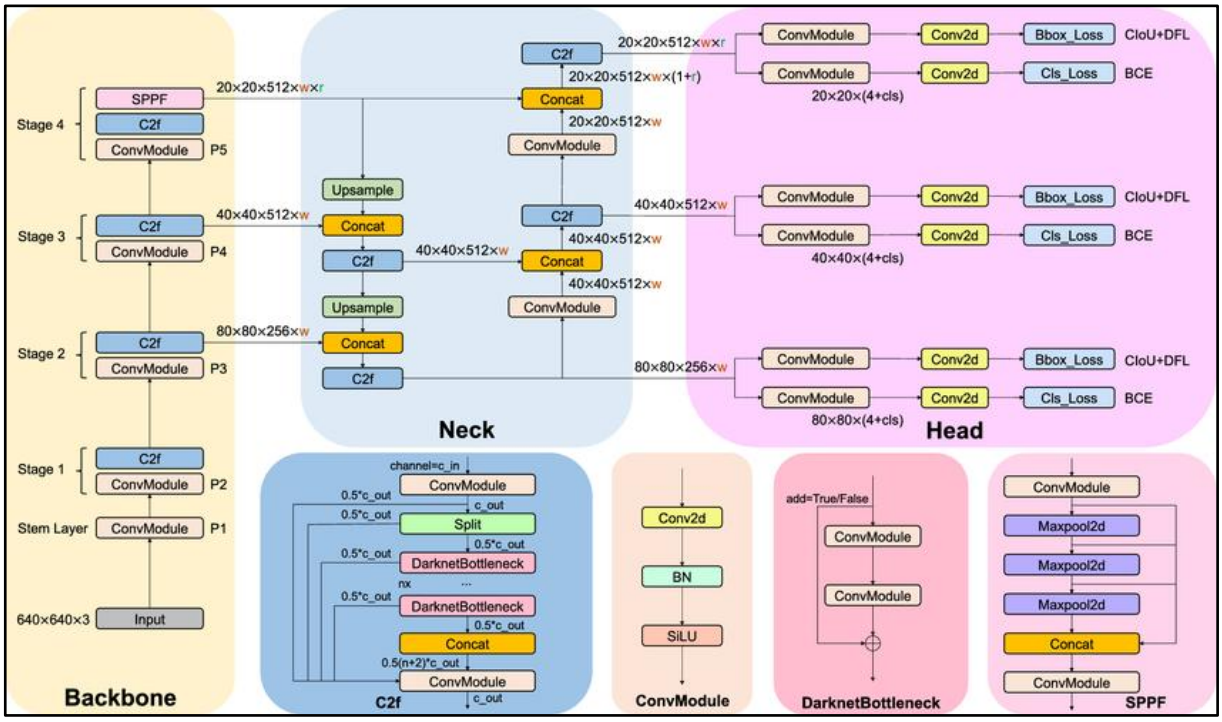

Fig 4.9 Brief diagram of YOLO v8 architecture overview

By integrating the YOLOv8 architecture into the project, the system achieves robust performance in detecting helmets and number plates, ensuring that traffic authorities and enforcement agencies can benefit from an accurate, efficient, and real-time solution. The advanced deep learning capabilities of YOLOv8 enhance detection accuracy, even in challenging conditions, contributing to improved road safety and automated traffic monitoring.

Chapter 5 : Implementation

The design and implementation involved the systematic development of a deep learning-based solution for Helmet and Number Plate Detection using the YOLOv8 architecture. The code is divided into distinct sections, each addressing specific tasks such as data preprocessing, model training, inference, and evaluation. The implementation is carried out in Google Colab and VS Code, using a Python environment where all necessary libraries and dependencies, including OpenCV, Ultralytics YOLO, and PyTorch, are installed. The system is designed to efficiently process real-time video feeds and images, ensuring accurate detection and classification of helmets and number plates for improved traffic monitoring and enforcement.

5.1 Code Snippets

1. Library

```
import streamlit as st
from ultralytics import YOLO
import torch
import cv2
import numpy as np
from PIL import Image
import tempfile
```

Fig 5.1 Library

- streamlit as st – Used for building an interactive web-based user interface.
- from ultralytics import YOLO – Imports the YOLOv8 model from Ultralytics for object detection.
- import torch – Used for deep learning operations, including model training and inference.
- import cv2 – OpenCV for image and video processing.
- import numpy as np – NumPy for numerical operations, especially handling arrays.
- from PIL import Image – PIL (Pillow) for handling image file formats.
- import tempfile – Allows handling temporary files, useful for storing uploaded images or videos.

2. Model Training

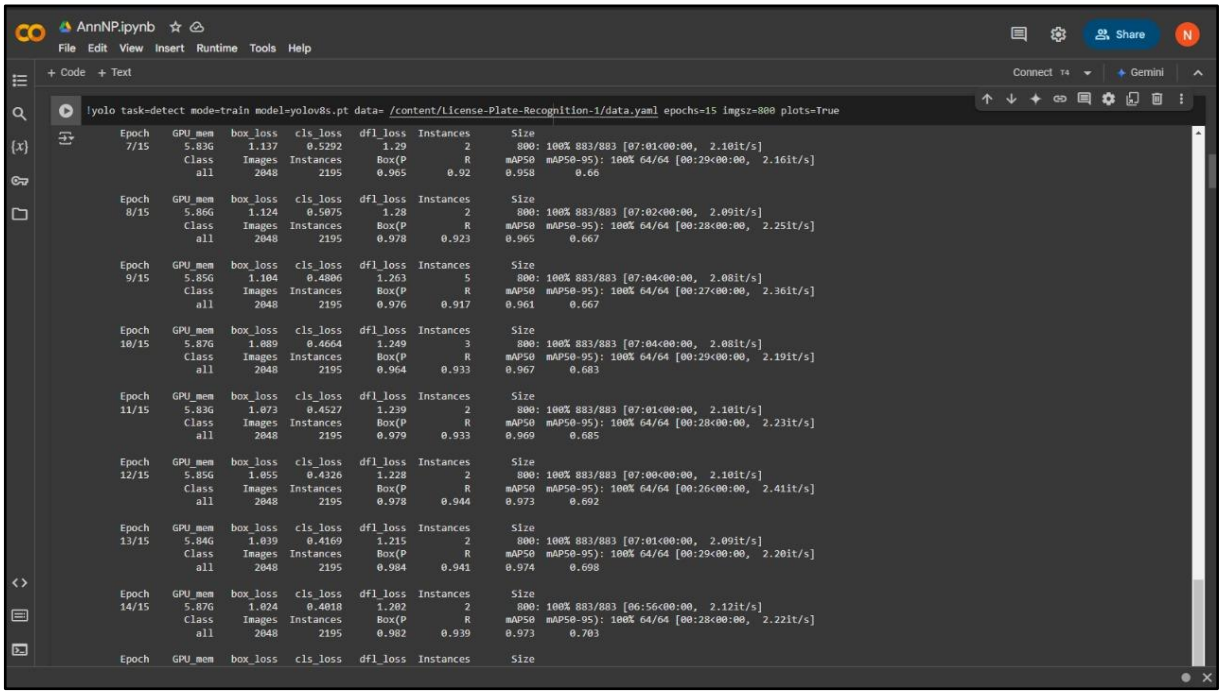

Fig 5.2 Model Training

- Task: Object detection (task=detect)
- Model: YOLOv8s (model=yolov8s.pt)
- Dataset: Path to the dataset configuration file (data=/content/License-Plate-Recognition-1/data.yaml)
- Epochs: 15 (epochs=15)
- Image Size: 800x800 (imgsz=800)
- Plots: Enabled for visualization (plots=True)

Training Progress Details:

The logs show training progress across multiple **epochs (7/15 to 14/15)** with the following key metrics:

1. GPU Memory Usage: ~5.8GB per epoch.
2. Loss Values:
   - box_loss (Bounding Box Regression Loss) – Measures accuracy in predicting object locations.
   - cls_loss (Classification Loss) – Measures accuracy in classifying objects.
   - dfl_loss (Distribution Focal Loss) – Improves object localization precision.
3. Instances Processed: ~2048 images per epoch.
4. mAP@50 and mAP@50-95 Scores:
   - mAP@50 (Mean Average Precision at 50% IoU) improves across epochs.
   - mAP@50-95 shows increasing performance, indicating better precision over different IoU thresholds.
5. Inference Speed: ~2.2 images per second.

3. Testing

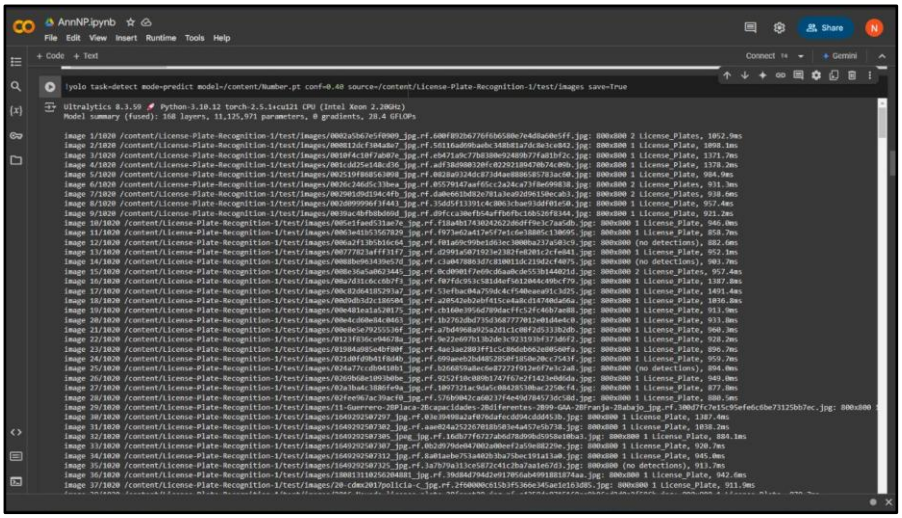

Fig 5.3 Testing

- task=detect → Runs YOLOv8 for object detection.
- mode=predict → Uses the trained model for prediction.
- model=/content/Number.pt → Specifies the trained YOLOv8 model (Number.pt).
- conf=0.40 → Sets confidence threshold to 40% (detections below this are ignored).
- source=/content/License-Plate-Recognition-1/test/images → Points to the directory containing test images.
- save=True → Saves the output images with detected license plates.

Output Details:

- Total Images Processed: 1020
- Image Resolution: 800x800
- Detection Speed: Varies between ~870ms to 1400ms per image.
- Detection Results:
  - Some images show "no detections", indicating missed license plates.

4. Validating and Testing Images

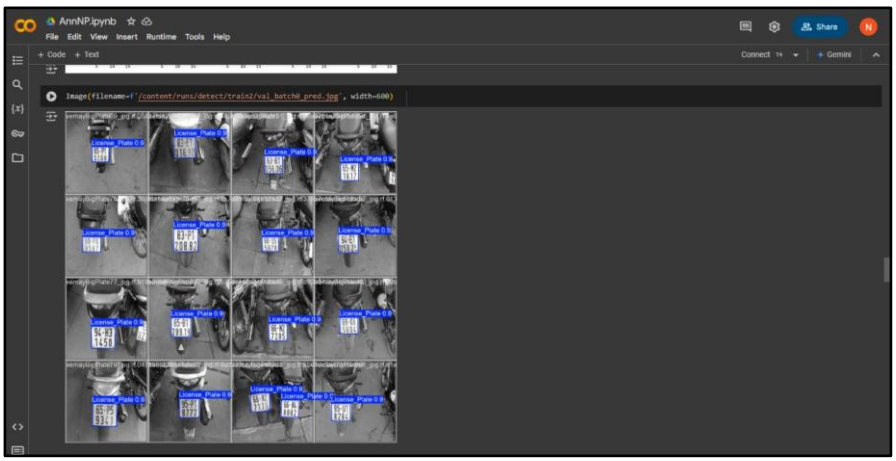

Fig 5.4 Validating and Testing Images

- **task=detect:** This tells YOLOv8 that you're performing **object detection** (not classification or segmentation).
- **mode=predict:** Runs the model in **prediction mode** using a trained model to detect license plates in images.
- **model=/content/Number.pt:** Specifies the trained YOLOv8 model (Number.pt) which has been fine-tuned for license plate detection.
- **conf=0.40:** Sets a **confidence threshold** of **40%**.
  - Only detections **above 40% confidence** are kept.
  - Lowering this might increase detections but could also introduce false positives.
- **source=/content/License-Plate-Recognition-1/test/images:** Defines the **input directory** where test images are stored.
- **save=True:** Saves the **output images** with bounding boxes drawn around detected license plates.

5. Model Metrics

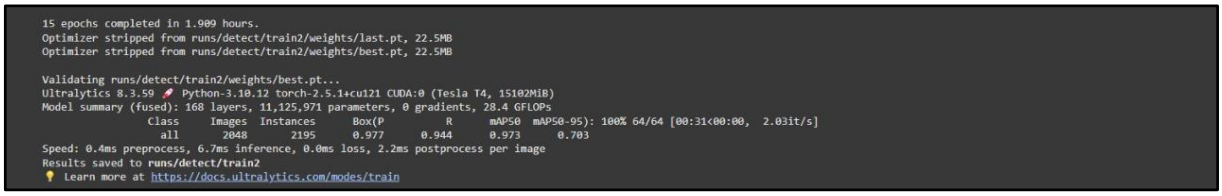

Fig 5.5 Model Metrics

- Training Duration: 15 epochs completed in 1.909 hours
- Model Optimization: Optimizer stripped from final weights (best.pt and last.pt)
- Model Summary: 168 layers, 11,125,971 parameters, 20.4 GFLOPs
  - 168 layers: Indicates a deep neural network structure.
  - 11.1 million parameters: Shows model complexity—higher values indicate a more powerful model, but it may need more training data.
  - 20.4 GFLOPs (Giga Floating Point Operations per Second): Measures computational complexity

.

- Detection Accuracy & Performance Metrics: Here are the key evaluation metrics

Metric	Value	Meaning
P (Precision)	0.977	97.7% of detected plates were correct.
R (Recall)	0.944	94.4% of actual plates were detected.
mAP50	0.973	97.3% accuracy at 50% IoU threshold.
mAP50-95	0.703	70.3% accuracy across various IoU thresholds.

- Processing Speed: Speed per image
  - Preprocessing: 0.4ms
  - Inference (detection): 6.7ms
  - Postprocessing: 2.2ms
  - Total inference time per image = 9.3ms
- Results & Next Steps: Results saved to: runs/detect/train2/
  - You can find all trained weights, metrics, and detection visualizations in this directory.

6. Loading Model

```
# Load the YOLO models
def load_helmet_model():
    return YOLO('/content/helmet-1.pt')  # Path to helmet detection model

def load_plate_model():
    return YOLO('/content/Number.pt')  # Path to number plate detection model

helmet_model = load_helmet_model()
plate_model = load_plate_model()
```

Fig 5.6 Model Loading

- Purpose: The code loads two YOLO models—one for helmet detection and another for license plate detection.
- Functionality:
  - load_helmet_model() loads the helmet detection model (helmet-1.pt).
  - load_plate_model() loads the license plate detection model (Number.pt).
  - Both functions return a YOLO model object that can be used for inference.
- Modular Approach: The functions allow easy swapping of models by changing file paths, making the code scalable.

7. Graphs

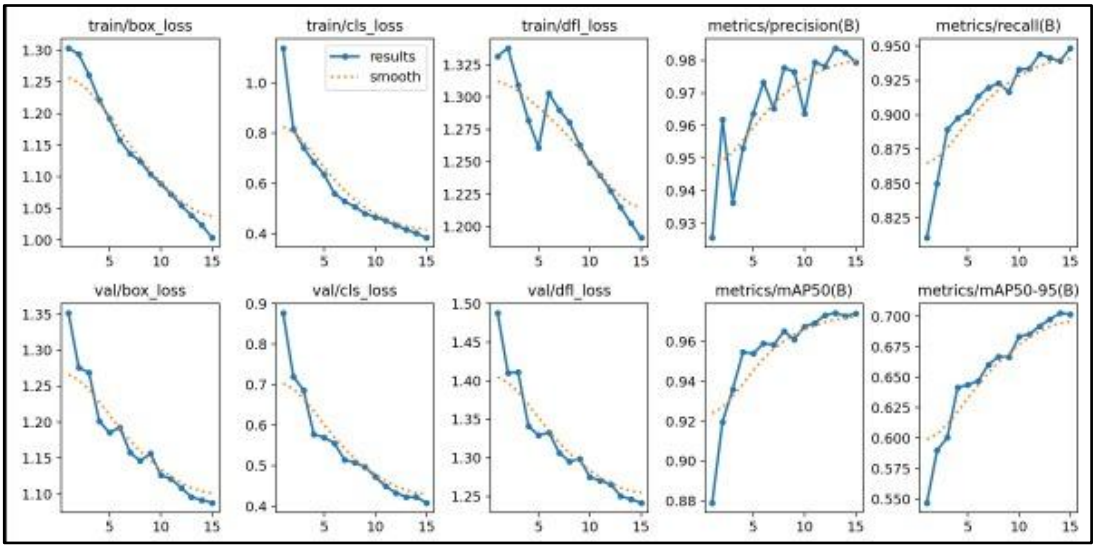

Fig 5.7 Performance metrics behind the YOLO v8 model.

8. Streamlit

```python
# Streamlit app layout
st.title("YOLO Object Detection with Streamlit")
st.sidebar.title("Choose Detection Task")
task = st.sidebar.radio("Select Task", ("Helmet Detection", "Number Plate Detection"))

# File uploader for images and videos
uploaded_file = st.file_uploader("Upload an image or video", type=["jpg", "jpeg", "png", "mp4", "avi", "mov"])

if uploaded_file is not None:
    file_type = uploaded_file.type.split('/')[0]

    if task == "Helmet Detection":
        st.header("Helmet Detection")
        selected_model = helmet_model
    else:
        st.header("Number Plate Detection")
        selected_model = plate_model

    if file_type == 'image':
        # Process image
        image = Image.open(uploaded_file)
        st.image(image, caption="Uploaded Image", use_column_width=True)

        # Convert the image to a format suitable for OpenCV
        image_cv = cv2.cvtColor(np.array(image), cv2.COLOR_RGB2BGR)

        # Perform inference
        results = selected_model(image_cv)

        # Render results on image
        img_with_boxes = results[0].plot()  # Get the first result and plot

        # Display the output
        st.image(img_with_boxes, caption="YOLO Object Detection Output", use_column_width=True)

    elif file_type == 'video':
        # Process video
        tfile = tempfile.NamedTemporaryFile(delete=False)
        tfile.write(uploaded_file.read())

        cap = cv2.VideoCapture(tfile.name)
        stframe = st.empty()

        while cap.isOpened():
            ret, frame = cap.read()
            if not ret:
                break

            # Perform inference on each frame
            results = selected_model(frame)

            # Render results on the frame
            frame_with_boxes = results[0].plot()  # Get the first result and plot

            # Convert frame with boxes back to RGB to display in Streamlit
            frame_rgb = cv2.cvtColor(frame_with_boxes, cv2.COLOR_BGR2RGB)
            stframe.image(frame_rgb, use_column_width=True)

        cap.release()

    else:
        st.error("Unsupported file type! Please upload an image or video.")
```

Fig 5.8 Streamlit code

9. Output

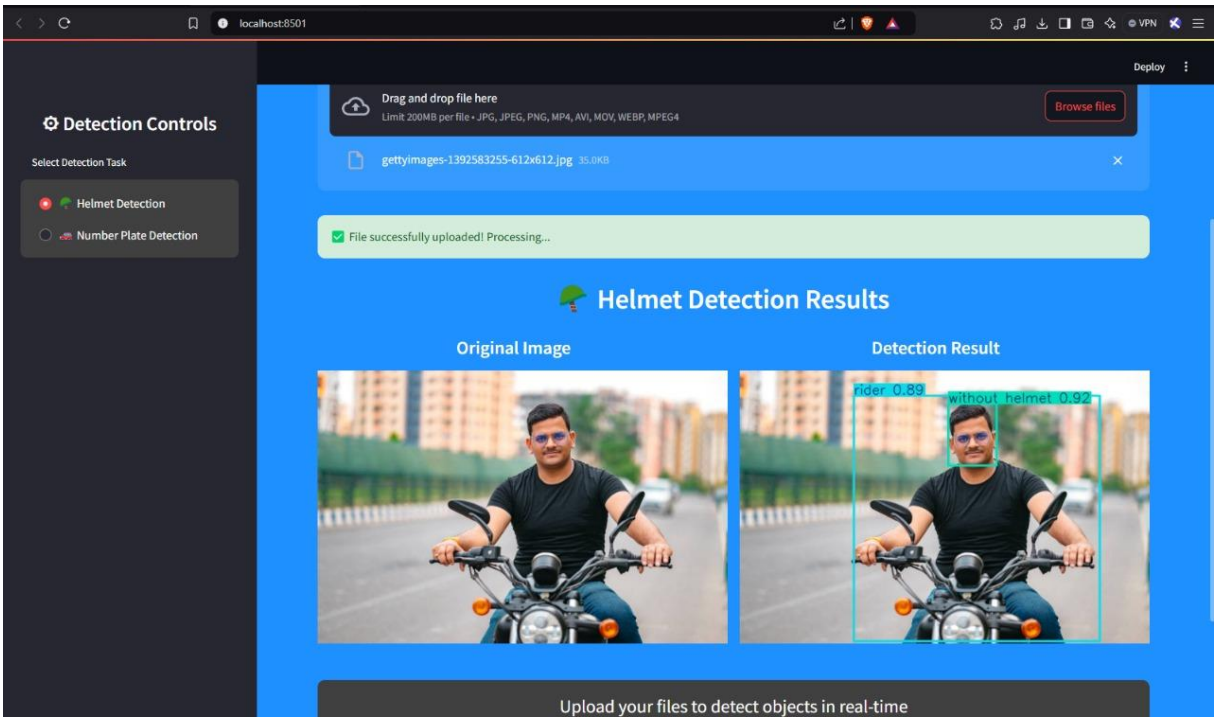

Fig 5.9 Output 1

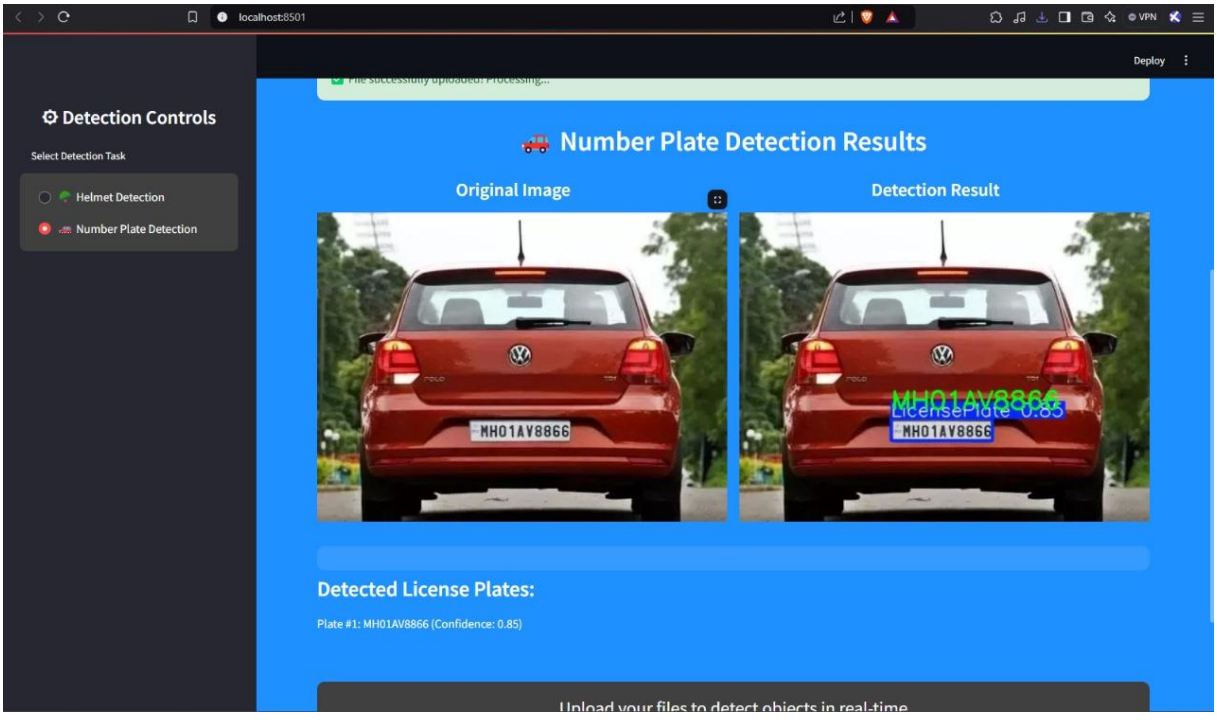

Fig 5.10 Output 2

10. Workflow

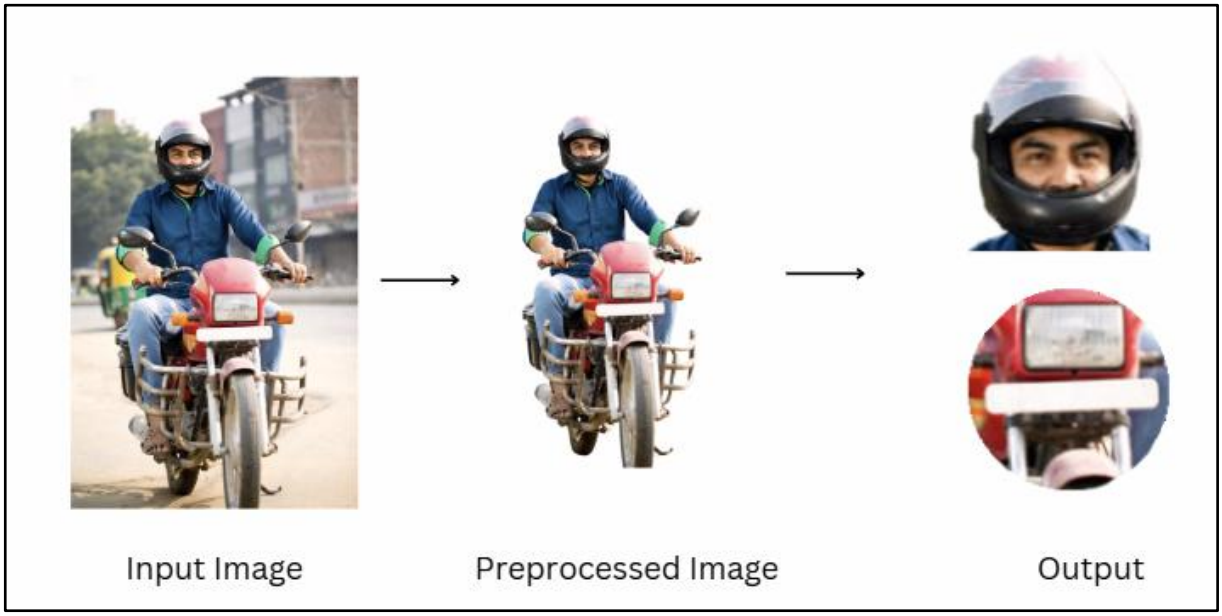

Fig 5.11 Workflow of the model

The workflow depicted in the image demonstrates the process of object detection using a YOLO model for tasks like helmet and number plate detection. It begins with the input image, which is an unprocessed raw image of a motorcyclist. The image is then passed through the model for preprocessing, where it is converted into a format suitable for analysis, including resizing and normalization. The YOLO model processes the preprocessed image to detect specific objects, such as the helmet and number plate. The output highlights the detected objects with bounding boxes, showcasing the model's capability to extract and focus on the required regions of interest. This workflow ensures accurate and focused detection for further analysis or decision-making.

Therefore, the implementation section outlines the workflow, from input image preprocessing to helmet and number plate detection using a trained YOLOv8 model, while the results demonstrate accurate identification of helmets and number plates in various conditions. This approach ensures practical solutions for enhancing safety monitoring and security applications.

Chapter 6 : Conclusion

The project Helmet and Number Plate Detection using YOLOv8 provides an efficient solution for real-time traffic monitoring and enforcement. By leveraging the YOLOv8 object detection model and Optical Character Recognition (OCR) techniques, the system successfully detects helmet usage and recognizes number plates from video feeds. The system offers several advantages, including high accuracy, real-time processing, and the ability to handle various environmental conditions.

Through this project, we have demonstrated the power of modern deep learning algorithms in solving practical problems related to road safety and traffic law enforcement. The implementation of YOLOv8 ensures fast and reliable object detection, while the OCR component enhances the system's ability to extract and log critical vehicle information.

The results and insights gained from testing and validation indicate that the system can effectively be deployed in real-world scenarios, providing a valuable tool for traffic authorities to monitor compliance and improve safety. Future enhancements could involve extending the system's capabilities to detect other traffic violations, improve robustness in challenging conditions, and integrate with broader smart city systems.

In conclusion, the Helmet and Number Plate Detection using YOLOv8 project is a significant step towards the development of automated and intelligent traffic monitoring systems that can contribute to safer roads and more efficient enforcement of traffic regulations.

Chapter 7 : Future Enhancements

1. **Multi-Vehicle Detection**
   - Extend the system's capability to detect multiple vehicles in the frame simultaneously, improving its usefulness in busy traffic conditions. This would involve enhancing the model's ability to distinguish between various vehicles and their respective number plates and helmets.
2. **Advanced Helmet Detection**
   - Improve the helmet detection feature by incorporating variations in helmet styles, colors, and the use of helmets in different orientations or lighting conditions. This would make the system more robust in detecting helmets under diverse real-world scenarios.
3. **Integration with Traffic Management Systems**
   - Integrate the system with existing traffic management infrastructure for automatic violation reporting and enforcement. This could include real-time alerts to traffic authorities, automated fines, and integrating with databases to track violations.
4. **Vehicle Classification**
   - Add vehicle classification to identify the type of vehicle (car, bike, truck, etc.), which could assist in categorizing violations more accurately, particularly in cases where specific rules apply to different vehicle types.
5. **Enhanced OCR for Number Plate Recognition**
   - Improve the OCR system to handle complex or non-standard number plate formats, including stylized or distorted plates, enhancing the system's ability to operate in diverse geographical locations with varying number plate designs.
6. **Cloud Integration**
   - Leverage cloud computing for data storage and real-time processing across multiple locations, allowing for scalability and centralized management of the system. Cloud integration would also enable remote monitoring and analytics.
7. **Weather and Lighting Condition Adaptation**
   - Implement advanced image enhancement techniques to address challenges posed by poor weather conditions (e.g., rain, fog) and varying lighting, improving the accuracy and reliability of detection in all environments.
8. **Mobile Application Development**
   - Develop a mobile app version of the system, allowing users to view detected violations, receive alerts, and interact with the system via smartphones. This would also enable enforcement officers to monitor traffic from mobile devices.

Bibliography

[1] D. Prajapati, S. Sabat, S. Bhilare, R. Vishe, and S. Bhujbal, "A Profile of Two Wheeler Road Traffic Accidents and Head Injuries in Bangalore," Medico-Legal Update, vol. 14, no. 1, pp. 139, Jan. 2014. DOI: 10.5958/j.0974-1283.14.1.034.

[2] V. Gupta, A. Kumar, P. Gupta, and S. P. Singh, "Pattern of two wheeler road traffic accidents in rural setting: a retrospective study," International Surgery Journal, Jan. 2016. DOI: 10.18203/2349-2902.isj20160952.

[3] H. Singh and A. D. Aggarwal, "Fatal road traffic accident in motorcyclists not wearing helmets," Journal of Punjab Academy of Forensic Medicine and Toxicology, vol. 11, no. 1, pp. 9-11, Jan. 2011.

[4] M. Jakubec, E. Lieskovska, A. Brezani, and J. Tothova, "Deep Learning-Based Automatic Helmet Recognition for Two-Wheeled Road Safety," Medico-Legal Update, vol. 14, no. 1, pp. 139, Jan. 2014. DOI: 10.5958/j.0974-1283.14.1.034.

[5] A. Rangari and A. Chouthmol, "Deep Learning based smart traffic light system using Image Processing with YOLO v7," S. A. Meshram and R. S. Lande, Traffic surveillance by using image processing, 2018 International Conference on Research in Intelligent and Computing in Engineering (RICE), 2018, pp. 1-3. DOI: 10.1109/RICE.2018.

[6] H. S. Bedi, P. K. Malik, R. Singh, R. Singh, and N. Bisht, "Traffic Surveillance Using Computer Vision and Deep Learning Methods," IOP Conference Series: Earth and Environmental Science, vol. 1285, 1st International Conference on Sustainable Energy Sources, Technologies and Systems, 2023, pp. 1-5, Phagwara, India, 03/08/2023 - 05/08/2023.

[7] S. Maheswaran, N. Indhumathi, S. Deepan Kumar, G. Murugesan, P. Duraisamy, and S. Nandita, "YOLOV5 Based A Real Time Automatic Number Plate And Helmet Recognition System," Wadare, K., Sharma, M., Patil, S., Bewoor, M., Pawar, A., Kadam, A., Patil, S., "Vehicle Number Plate Detection using YoloV8 and EasyOCR," 14th ICCCNT IEEE Conference, July 6-8, 2023, IIT - Delhi, Delhi.

[8] T. Kale, A. Dhavale, A. Randhavan, and V. S. Inamdar, "Helmet Detection and Number Plate Recognition," IJRASET, vol. 52559, May 19, 2023, ISSN: 2321-9653.

[9] R. Sarmah, P. Lahon, and T. Ahmed, "Enhanced Precision in Motorcycle Helmet Detection: YOLOv5 and Darknet Approach," Researchgate, July 2024. DOI: 10.21203/rs.3.rs-4577583/v1, License CC BY 4.0.

[10] S. S. Patil, S. H. Pati, A. M. Pawar, M. S. Bewoor, A. K. Kadam, U. C. Patkar, K. Wadare, and S. Sharma, "Vehicle Number Plate Detection using YoloV8 and EasyOCR," The Institute of Engineering and Technology (IET), June 29, 2021. DOI: 10.1049/ipr2.12295.

[11] A. Dalvi, J. Fernandes, C. Toms, B. Saju, and U. Patil, "Automatic Number Plate Recognition System," International Journal of Creative Research Thoughts, vol. 12, no. 4, Apr. 2024, ISSN: 2320-2882.

[12] C.-J. Lin, C.-C. Chuang, and H.-Y. Lin, "Edge-AI-Based Real-Time Automated License Plate Recognition System," MDPI, Appl. Sci., vol. 12, no. 3, pp. 1445, 2022. DOI: 10.3390/app12031445.

[13] Z. Rao, D. Yang, N. Chen, and J. Liu, "License Plate Recognition System in Unconstrained Scenes via a New Image Correction Scheme and Improved CRNN," Elsevier, Expert Systems with Applications, vol. 243, pp. 122878, Jun. 2024. DOI: 10.24433/CO.0006948.v1.

[14]L. Hu and J. Ren, "YOLO-LHD: an enhanced lightweight approach for helmet wearing detection in industrial environments," Front. Built Environ., vol. 9, Sec. Structural Sensing, Control and Asset Management, 10 Nov. 2023. DOI: 10.3389/fbuil.2023.1288445.

[15]A. K. R., S. P., and S. P., "Camera-mounted Helmet detection with Yolo v8 on custom dataset," ACM Digital Library, DOI: 10.1145/3675888.3676032, IC3 2024: 2024 Sixteenth International Conference on Contemporary Computing (IC3-2024), Noida, India, Aug. 2024.

[16]Y. Ji, Y. Cao, X. Cheng, and Q. Zhang, "Research on the Application of Helmet Detection Based on YOLOv4," Scientific Research, Aug. 2022. DOI: 10.4236/jcc.2022.108009.

[17] P. S. Solanki, "Advancing Road Safety: Helmet Detection with Artificial Intelligence," Researchgate, Oct. 2024. DOI: 10.70127/irjedt.vol.06.issue11.251.
[18] S. Chandran, S. Chandrasekar, and N. E. Elizabeth, "Konnect: An Internet of Things (IoT) based smart helmet for accident detection and notification," IEEE, 2016 IEEE Annual India Conference (INDICON), 2016. DOI: 10.1109/INDICON.2016.7839052.

[19] A. O. Gökcan and R. Çöteli, "Detection and Classification of Vehicles by Using Traffic Video Based on YOLOV8," Researchgate, Conference: UMTEB - XIV International Scientific Research Congress, Sept. 14-15, 2023, Naples, Italy, pp. 530-538.

[20] D. Prajapati, S. Sabat, S. Bhilare, R. Vishe, and P. S. Bhujbal, "Helmet Detection and Number Plate Recognition Using YOLOv8 and Tensorflow Algorithm in Machine Learning," International Journal of Innovative Research in Computer Science & Technology (IJIRCST), vol. 12, no. 2, Mar. 2024, ISSN: 2347-5552.

[21] World Health Organization (WHO), Global Status Report on Road Safety 2018.

[22] NHTSA (National Highway Traffic Safety Administration), "Motorcycle Helmet Use and Head Injury Statistics," 2021.

[23] Gupta, R., Sharma, A., & Kumar, P. (2021). "Challenges in Traffic Law Enforcement and Helmet Detection Using AI," International Journal of Traffic Management, 34(2), 56-72.

[24] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv preprint arXiv:2004.10934.

[25] Roboflow, "Helmet and License Plate Detection Dataset," accessed 2024.

[26] Wang, C. Y., Bochkovskiy, A., & Liao, H. Y. M. (2022). YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. arXiv preprint arXiv:2207.02696.

[27] Kaur, M., Juneja, M., & Singh, S. (2022). "A Deep Learning-Based Traffic Surveillance System for Helmet Detection and Number Plate Recognition," Multimedia Tools and Applications, 81(10), 14271–14291.

[28] McKinsey & Company, "How AI and IoT Are Transforming Traffic Safety and Law Enforcement," 2023.

[29] Sharma, P., & Verma, R. (2023). "Impact of AI-based Surveillance on Road Safety," Smart Transportation Journal, 29(3), 102-118.