

# VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belgavi-590018



**“A PROJECT REPORT”**  
**[Subject Code: 21CSP76]**  
**ON**

## **“Intelligent Traffic Violation Detection and Notification System Using Deep Learning”**

*Submitted in partial fulfillment of the requirement for award of degree*

*of*

**BACHELOR OF ENGINEERING**

*in*

**COMPUTER SCIENCE AND ENGINEERING**

**By**

<b>PRITISH ALI</b>	<b>1EP21CS076</b>
<b>RUPAM BHATTACHARYYA</b>	<b>1EP21CS089</b>
<b>H D KISHORE</b>	<b>1EP22CS405</b>
<b>VINAY KUMAR</b>	<b>1EP21CS122</b>

**Under the guidance of**

**Mrs. Manimegalai A**  
**Asst. Professor**  
**Dept. of CSE, EPCET**



**EAST  
POINT**

**COLLEGE OF ENGINEERING &  
TECHNOLOGY**

**Department of Computer Science and Engineering**

Approved by AICTE New Delhi | Affiliated to VTU, Belgavi,  
Virgo Nagar, Bengaluru-560049



**2024-2025**



**EAST  
POINT**

**COLLEGE OF ENGINEERING &  
TECHNOLOGY**

**Department of Computer Science and Engineering**

Approved by AICTE New Delhi | Affiliated to VTU, Belagavi,  
Virgo Nagar, Bengaluru-560049



## CERTIFICATE

This is to certify that the Project work entitled **“Traffic Violation Detection and notification System Using Deep Learning”** is a bonafide work carried out by **PRITISH ALI [1EP21CS076]**, **RUPAM BHATTACHARYYA [1EP21CS089]**, **H D KISHORE [1EP22CS405]**, and **VINAY KUMAR [1EP21CS122]**, in the partial fulfillment for the award of Bachelor of Engineering in Computer Science and Engineering of Visvesvaraya Technological University, Belagavi, during the year **2024-2025**. It is certified that corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The mini project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

(Signature of the Guide)

Mrs. Manimegalai A  
Asst. Prof., Dept. of CSE,  
EPCET, Bengaluru

(Signature of the HOD)

Dr. L. Mahimozhi  
HOD, Dept. of CSE,  
EPCET, Bengaluru

(Signature of the Principal)

Dr. Mrityunjaya V. Latte  
Professor & Principal  
EPCET, Bengaluru

**External Viva-Voce:**

**Name of the Examiners**

- 1.
- 2.

**Signature with date**

**PRINCIPAL**  
EAST POINT COLLEGE OF  
ENGINEERING & TECHNOLOGY  
BANGALORE- 560049.

  
29/5/25

## ACKNOWLEDGEMENT

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. We would like to take this opportunity to thank them all.

First and fore most we would like to express our sincere regards and thanks to **Mr. Promod Gowda** and **Mr. Rajiv Gowda**, CEOs East Point Group of Institutions, Bangalore, for providing necessary infrastructure and creating good environment.

We express our gratitude to **Dr. T.K. Sateesh**, Principal, EPCET who has always been a great source of inspiration.

We express our sincere regards and thanks to **Dr. I. Manimozhi**, Professor and Head, Department of Computer Science and Engineering, EPCET, Bangalore, for his encouragement and support.

We are grateful to acknowledge the guidance and encouragement given to us by **Mrs. Manimegalai A**, Assistant Professor, Department of Computer Science and Engineering, EPCET, Bangalore, who has rendered a valuable assistance.

We also extend our thanks to all the faculties of the **Department of Computer Science and Engineering, EPCET**, Bangalore, who have encouraged us throughout the course of the Mini Project Work.

Last, but not the least, we would like to thank our family and friends for their inputs to improve the Mini Project works.

**PRITISH ALI [1EP21CS076]**  
**RUPAM BHATTACHARYYA [1EP21CS089]**  
**HD KISHOR [1EP22CS405]**  
**VINAY KUMAR C [1EP21CS122]**

## **ABSTRACT**

Intelligent vehicle detection and counting are becoming increasingly important in the field of highway management. However, due to the different sizes of vehicles, their detection remains a challenge that directly affects the accuracy of vehicle counts. To address this issue, this paper proposes a Visionbased vehicle detection and counting system using You Only Look Once (YOLO-V8) based DeepSORT model for real time vehicle detection and tracking from video sequences. Deep learning based Simple Real time Tracker (Deep SORT) algorithm is added, which will track actual presence of vehicles from video frame predicted by YOLO-V8. So, the false prediction perform by YOLOV8 can be avoid by using DeepSort algorithm. The video will be converted into multiple frames and give as input to YOLO-V8 for vehicle detection. The detected vehicle frame will be further analyzed by DeepSort algorithm to track vehicle and if vehicle tracked then DeepSort will put bounding box across tracked vehicle and increment the tracking count. The proposed model is trained with three different datasets such as public and custom collected dataset.

# CONTENTS

Chapter no.	Description	Page no.
<b>1</b>	<b>Introduction</b>	<b>1</b>
	1.1 Background	1
	1.2 Problem Statement	2
	1.3 Existing System	2
	1.4 Proposed System	2
	1.5 Aim of the Project	3
	1.6 Objectives of the project	3
	1.7 Summery	3
<b>2</b>	<b>Literature Survey</b>	<b>5</b>
<b>3</b>	<b>Requirement Specification</b>	<b>14</b>
	3.1 Hardware Requirement	14
	3.2 Software Requirement	14
	3.3 Development Environment	14
	3.4 Functional Requirements	15
	3.5 Non-functional Requirements	15
	3.6 Summary	15
<b>4</b>	<b>System Design</b>	<b>16</b>
	4.1 System Architecture	16
	4.2 Dataflow Diagrams	19
	4.3 UML Diagram	21
	4.4 Summary	22

<b>5</b>	<b>System Implementation</b>	<b>23</b>
	5.1 Modules and Components	23
	5.2 Algorithms and Pseudo Codes	24
	5.3 Summary	31
<b>6</b>	<b>Result and Analysis</b>	<b>32</b>
	6.1 Snapshots	32
	6.2 Summary	36
<b>7</b>	<b>Conclusion and Future Enhancement</b>	<b>37</b>
	<b>References</b>	<b>39</b>

## List of Figures

<b>Fig. no.</b>	<b>Description</b>	<b>Page no.</b>
4.1	Proposed System Architecture	18
4.2	Activity Diagram	21
5.1	View function to handle request	28
5.2	Model.py to make the database Intelligent	28
5.3	Function to send email notification	29
5.4	Vehicle and Number Plate Detection function	29
5.5	Template to upload video	30
5.6	Template to report an incident	30
6.1	Video upload section	32
6.2	Detection of Vehicle and Numberplate	33
6.3	Sending notification	34
6.4	Success message	35
6.5	Admin Section	35
6.6	Form to report incident	36

## Chapter 1

# INTRODUCTION

## 1.1 Background

Traffic violations such as overspeeding, running red lights, illegal parking, and other infractions are major contributors to road accidents and congestion in urban areas. Traditional methods of monitoring traffic violations, including manual observation or basic camera systems, often prove inefficient, prone to human error, and resource-intensive. This project aims to develop an **"Intelligent Traffic Violation Detection and Notification System"** leveraging advanced deep learning techniques to detect violations automatically, ensuring faster, more accurate identification of offenders while reducing the burden on law enforcement agencies.

This system integrates cutting-edge technologies such as YOLO (You Only Look Once) for vehicle detection, DeepSORT for vehicle tracking, and OCR (Optical Character Recognition) for extracting license plate information. By analyzing video footage or real-time feeds from traffic cameras, the system can identify overspeeding vehicles, track them, and cross-check the extracted license plate numbers with a vehicle owner database. For confirmed violations, the system sends automated notifications to both the violator and the corresponding traffic control room, ensuring timely action.

The project also includes a module for citizen engagement, allowing users to report incidents by uploading details, including location and photos, through a user-friendly web interface. The data collected is used to identify accident-prone locations using AI algorithms, generating rankings and enabling targeted interventions. This comprehensive approach not only enhances road safety and compliance but also provides a scalable, technology-driven solution for smarter traffic management in urban areas.

Developing such systems come with their own set of challenges, including the requirement for large, diverse datasets to ensure the accuracy and generalizability of deep learning models. Environmental factors such as lighting conditions, weather variations, and occlusions also affect the performance of the system. Ethical and legal considerations related to surveillance and data privacy also need to be addressed to gain public trust and ensure compliance with regulations. Despite these challenges, the potential for an intelligent traffic violation detection system is immense with a transformative approach to modern traffic management and law enforcement.



## **1.2 Problem Statement**

Well-designed, robust transportation networks based on spatial data are essential to every country's economic development. Nevertheless, the majority of cities worldwide continue to struggle with a sharp rise in traffic volume and challenges with traffic management, which lowers the standard of living in contemporary cities. Nonetheless, recent developments in artificial intelligence, sensor technologies, and internet bandwidth have reduced these challenges by working together to provide location intelligence for public safety. Authorities can achieve resilience in road safety, regulated commutes, and road condition evaluations by automating location intelligence in road environments utilizing sensing technology.

## **1.3 Existing system**

The existing system define explicit rules for what constitutes abnormal behavior, such as exceeding speed limits, sudden lane changes, or stopping on the highway. While simple and efficient, they are often inflexible and may miss subtle or complex types of abnormal behavior. These methods analyze traffic flow data to identify deviations from normal patterns, such as sudden changes in vehicle density or velocity. They can be more adaptable than rule-based systems but may require large amounts of training data and can be sensitive to changes in traffic patterns. These techniques use image processing and machine learning algorithms to analyze video footage from traffic cameras and identify abnormal behavior based on vehicle trajectories, movements, and interactions. They offer more flexibility and can capture complex patterns, but can be computationally expensive and require specialized training data.

There are some limitations are there. Existing systems, particularly rule-based and statistical methods, may suffer from high false positive or false negative rates. This can lead to unnecessary alerts and missed critical events. Traditional systems may struggle to adapt to changing traffic patterns or new types of abnormal behaviour.

## **1.4 Proposed system**

The proposed system aims to implement a robust vehicle detection and tracking model using the YOLO (You Only Look Once) object detection framework in conjunction with DeepSORT (Deep Simple Online and Realtime Tracking). YOLO will be utilized for real-time vehicle detection, providing accurate bounding box coordinates. These coordinates will then be processed by DeepSORT, which employs data association and Kalman filtering techniques to

track vehicles consistently over time. The integration of YOLO and DeepSORT in the proposed system is expected to deliver a high-performance solution for efficient and accurate vehicle detection and tracking in real-world scenarios.

## **1.5 Aim of the project**

The aim of the proposed project is to create an automated system that would detect violations such as jumping the red light, speeding, and illegal parking through the application of deep learning techniques. In this context, the system shall analyze the real-time video from traffic cameras to identify these violations correctly. Once a violation is detected, the system will alert the relevant authorities or notify the violators through alerts. Its purpose is to enhance the management of traffic, minimize accidents, and strengthen law enforcement by providing timely and reliable detection and notification of traffic offenses.

## **1.6 Objectives of the Project**

The primary objective of the "Intelligent Traffic Violation Detection and Notification System Using Deep Learning" project is to develop a more autonomous system that may real-time detect traffic violation offenses by using deep learning algorithms. The system can therefore identify violations such as red lights jumping, speeding, illegal parking, and other live-traffic camera feeds or pictures for such violations. Once a violation is detected, the system will send an instant notification to the relevant authorities or law enforcement agencies for timely action. The project aims to improve traffic management, reduce human error, and enhance road safety by leveraging AI for efficient monitoring and enforcement.

## **1.7 Summary**

The aim of the project is to automatically identify traffic violations using advanced computer vision techniques. By leveraging deep learning algorithms, the system can analyse real-time video feeds from traffic cameras to detect violations such as running red lights, speeding, or illegal parking. Once a violation is identified, the system immediately sends notifications to relevant authorities or individuals for prompt action. This project aims to improve road safety, reduce traffic-related issues, and ensure efficient monitoring by automating the detection and reporting of traffic violations.

## Chapter 2

### LITERATURE SURVEY

#### 2.1 Towards Detection of Abnormal Vehicle Behavior Using Traffic Cameras

In the paper "Towards Detection of Abnormal Vehicle Behavior Using Traffic Cameras" (2019), C. Wang, A. Musaev, P. Sheinidashtegol, and T. Atkison proposed system for detecting abnormal vehicle behavior using traffic cameras, Overall Aim is to develop a real-time application for identifying abnormal driving patterns from traffic video streams. Methodologies of this paper are to combines a Convolutional Neural Network (CNN) for feature extraction and a Long Short-Term Memory (LSTM) network for capturing temporal dependencies and focuses on identifying stopped vehicles and potential abnormal behaviors like erratic lane changes, sudden U-turns, or speeding. Designed for fast processing and immediate anomaly alerts. The evaluation of this title is to tested on real traffic camera footage to detect stopped vehicles and potential abnormal behaviors as well as achieved promising results with high accuracy rates for both tasks. Some significances are potential for improving traffic safety by proactively identifying potentially dangerous driving behaviors and opens avenues for further research on advanced anomaly detection using cameras and deep learning.

##### Merits:

- ☐ The system achieved high accuracy in detecting both stopped vehicles and abnormal behaviors in real-time testing using real traffic camera footage.
- ☐ By proactively identifying dangerous driving patterns, the system could help prevent accidents and save lives.
- ☐ This work paves the way for further research on advanced anomaly detection using cameras and deep learning in traffic monitoring systems.

##### Demerits:

- ☐ The paper doesn't specify the type of CNN and LSTM architecture used, making it difficult to compare the approach with other methods
- ☐ The evaluation dataset and its size are not explicitly mentioned, making it hard to assess how generalizable the results are.
- ☐ The system might misidentify normal driving behavior as abnormal in certain situations.

## **2.2 An intelligent multiple vehicle detection and tracking using modified Vibe algorithm and deep learning algorithm**

The overall aim of this paper is to overcome limitations of existing methods, the paper develops a hybrid approach for both accurate vehicle detection and robust tracking across different traffic scenarios. The methodologies are an enhanced version of the classic background subtraction algorithm for vehicle detection. This modification improves background extraction and reduces false positives, and a convolutional neural network (CNN) used for fine-grained vehicle identification and location refinement. Some key features are real-time performance, location information and improved accuracy. The authors tested their system on two publicly available datasets, KITTI and DETRAC, comparing its performance against other methods. The system is designed for fast processing and tracking of multiple vehicles simultaneously. Unlike bounding boxes, the system provides precise vehicle locations, crucial for applications like trajectory analysis and motion estimation. The hybrid approach aims to overcome limitations of both individual algorithms, leading to more accurate detection and tracking results. Overall, this paper presents an interesting hybrid approach for multiple vehicle detection and tracking with promising results.

### **Merits**

- ☐ The modified Vibe algorithm improves background extraction and reduces false positives, resulting in more accurate vehicle detection.
- ☐ The system is designed for real-time performance, which is crucial for applications like traffic monitoring and safety.
- ☐ Unlike traditional methods that use bounding boxes, this system provides precise vehicle locations.

### **Demerits**

- ☐ The paper does not provide detailed information about the size, diversity, and source of the datasets used for training and testing the deep learning algorithm.
- ☐ Both the modified Vibe algorithm and the CNN component can be susceptible to occlusions, particularly complete occlusions where the vehicle is entirely blocked from view.
- ☐ While the combined approach aims to achieve both accuracy and real-time performance, there might be a trade-off between the two.

## 2.3 Real-Time Vehicle Detection and Tracking Using YOLOv3 and DeepSORT

In the paper "Real-Time Vehicle Detection and Tracking Using YOLOv3 and DeepSORT" (2020), authors S. Ren, H. He, and X. Zhao propose a robust system for vehicle detection and tracking in urban environments using real-time traffic camera footage. The overall aim is to develop an efficient and accurate method for monitoring vehicular movement in real-time.

### Methodologies:

- ☐ Utilizes YOLOv3 (You Only Look Once version 3) for fast and accurate object detection.
- ☐ Employs DeepSORT (Simple Online and Realtime Tracking with a Deep Association Metric) for robust multi-object tracking.
- ☐ Addresses challenges of occlusions, varying lighting conditions, and high traffic density.

### Evaluation:

- ☐ Tested on the UA-DETRAC and KITTI datasets, achieving high accuracy in vehicle detection and tracking.
- ☐ Demonstrates the system's ability to maintain real-time performance even in high-density traffic scenarios.

### Merits:

- ☐ The system achieves high accuracy and robustness in vehicle detection and tracking.
- ☐ Real-time performance makes it suitable for practical applications in urban traffic monitoring.

### Demerits:

- ☐ Susceptibility to performance degradation under adverse weather conditions.
- ☐ Lack of detailed information on computational resource requirements.

## 2.4 Vehicle Anomaly Detection Using Attention-Based LSTM Networks

In the paper "Vehicle Anomaly Detection Using Attention-Based LSTM Networks" (2021), authors J. Liu, M. Li, and Q. Zhang present a novel approach for detecting abnormal vehicle behaviors using an attention mechanism integrated with Long Short-Term Memory (LSTM) networks. The overall aim is to improve the accuracy of anomaly detection by focusing on relevant features in the data.

### Methodologies:

- ☐ Utilizes an attention mechanism to prioritize important features in the traffic data, enhancing the LSTM's ability to detect anomalies.
- ☐ The system is designed to identify behaviors such as sudden stops, erratic lane changes, and unusual speed variations.

### Evaluation:

- ☐ Evaluated on a custom dataset collected from urban traffic cameras, achieving high precision and recall rates in anomaly detection.
- ☐ Compared with standard LSTM and CNN-based approaches, showing superior performance.

### Merits:

- ☐ The attention mechanism significantly enhances the LSTM's capability to detect vehicle anomalies.
- ☐ High precision and recall rates indicate the system's potential for practical deployment in traffic monitoring systems.

### Demerits:

- ☐ Lack of publicly available dataset for result validation.
- ☐ Potential computational overhead due to the attention mechanism, affecting real-time performance.

## **2.5 Vision-based Vehicle Detection and Counting System Using Deep Learning in Highway Scenes**

In the paper "Vision-based Vehicle Detection and Counting System Using Deep Learning in Highway Scenes" (2019), authors H. Song, H. Liang, H. Li, Z. Dai, and X. Yun propose a system for vehicle detection and counting using deep learning techniques. The overall aim is to develop an efficient and accurate method for monitoring traffic flow on highways.

### **Methodologies:**

- ☐ Utilizes a convolutional neural network (CNN) for vehicle detection in highway scenes.
- ☐ Implements a counting mechanism based on detected vehicle trajectories to estimate traffic volume.
- ☐ The system is designed to handle various lighting conditions and high-speed traffic.

### **Evaluation:**

- ☐ Tested on real-world highway footage, achieving high accuracy in vehicle detection and counting.
- ☐ Demonstrates the system's ability to operate in real-time with minimal latency.

### **Merits:**

- ☐ Achieves high accuracy in vehicle detection and counting, making it suitable for highway traffic monitoring.
- ☐ Real-time performance with minimal latency enhances practical applicability.

### **Demerits:**

- ☐ Susceptibility to performance degradation under adverse weather conditions.
- ☐ Lack of detailed information on computational resource requirements.

## **2.6 Tweeting Traffic: Analysing Twitter for Generating Real-time City Traffic Insights and Predictions**

In the paper "Tweeting Traffic: Analysing Twitter for Generating Real-time City Traffic Insights and Predictions" (2015), authors Tejaswin P., Kumar R., and Gupta S. present a system for analysing Twitter data to generate real-time traffic insights and predictions. The overall aim is to leverage social media data for enhancing traffic monitoring and management.

**Methodologies:**

- ☐ Analyses Twitter data using natural language processing (NLP) techniques to identify traffic-related tweets.
- ☐ Uses machine learning algorithms to predict traffic conditions based on tweet content.
- ☐ Integrates real-time traffic data with social media insights for comprehensive traffic monitoring.

**Evaluation:**

- ☐ Tested on Twitter data collected from a major city, achieving high accuracy in traffic condition prediction.
- ☐ Demonstrates the system's ability to provide real-time traffic insights and improve prediction accuracy.

**Merits:**

- ☐ High accuracy in predicting traffic conditions using real-time Twitter data.
- ☐ Provides valuable insights for traffic management and emergency response planning.

**Demerits:**

- ☐ Susceptibility to variability in Twitter data quality and relevance.
- ☐ Potential privacy concerns related to analysing social media data.

## **2.7 An Intelligent Traffic Violation Detection System Using Deep Learning and Computer Vision**

In the paper "An Intelligent Traffic Violation Detection System Using Deep Learning and Computer Vision" (2020), authors P. Gupta, R. Singh and A. Sharma propose a system that detects traffic violations such as signal jumping, lane switching, and over-speeding using deep learning models such as CNNs. The authors implemented object detection algorithms like YOLOv4 and compared it with Faster R-CNN for vehicle detection and license plate recognition. Furthermore, the system uses OCR to extract vehicle registration details. It incorporates real-time processing via surveillance cameras and provides automated violation reports that are delivered to violators through mobile applications. The model was proven efficient with an average accuracy of 92%, showing its capability in improving traffic regulation enforcement.



**Methodologies:**

- ☐ Data Gathering: Surveillance camera videos will be used to track traffic flow conditions.
- ☐ Data Labeling: Labelling the data collected will mark cases of traffic offenses such as red-light running and over-speeding.
- ☐ Model Training: The system would use deep learning algorithms to train models on annotated data to identify different cases of traffic violations.
- ☐ Real-Time Detection: Applying the trained models to process live video feeds for real-time identification and reporting of traffic violations.

**Merits:**

- ☐ Accuracy of up to 92% in identifying multiple traffic offenses.
- ☐ Real-time processing guarantees timely detection of violations.
- ☐ Incorporation of license plate recognition and OCR enhances value in automated enforcement systems.

**Demerits:**

- ☐ System performance would degrade in poor weather conditions.
- ☐ Very limited scope of violations considered. Does not cover parking violations
- ☐ Uses very high computation power for detection in real-time.

## **2.8 Deep Learning-Based Approach for Traffic Rule Violation Detection**

In the paper "Deep Learning-Based Approach for Traffic Rule Violation Detection" (2020), authors S. Kumar, L. Wei and D. Patel propose a system that conducted in this work explores the ability of advanced deep learning models for the detection of redlight violation and illegal U-turn in cities. A hybrid approach has been adopted that couples CNNs as object detectors and LSTMs (Long Short-Term Memory networks) as movement trajectory trackers in the detection framework. Through extensive training on the dataset of various urban traffic scenarios, the precision of the system was recorded as 88%. It also has an alert generation mechanism that alerts violators and authorities immediately. The research focuses on scalable architectures for metropolitan cities with dense traffic.

**Methodologies:**

- ☐ Motorcycle Detection: The first is the detection of motorcycles in the input images. This is because the system focuses on specific violations concerning two-wheeled vehicles.
- ☐ Helmet Detection: After motorcycle detection, the system checks for helmet usage by the riders. This step ensures safety regulations that require helmet use.
- ☐ Triple Riding Detection: The methodology identifies instances where more than two individuals are riding a motorcycle, which is a common traffic violation.
- ☐ The final system is to detect and read the number plates of the motorcycles so that violators can be identified.

**Merits:**

- ☐ CNNs and LSTMs are effectively integrated to detect violations.
- ☐ The model gives a holistic analysis of vehicle trajectories.
- ☐ Alert generation mechanism for immediate notification.

**Demerits:**

- ☐ Lower accuracy compared to other state-of-the-art models.
- ☐ Computational complexity increases with the density of traffic.
- ☐ No provision for edge computing, hence not scalable.

## 2.19 Smart Traffic Surveillance Using Deep Neural Networks

In the paper "Smart Traffic Surveillance Using Deep Neural Networks " (2022), authors R. Lopez, F. Zhang and M. Chen propose a smart traffic surveillance system, powered by deep neural networks (DNNs), detects signal skipping and wrong-way driving. The authors applied pre-trained models like ResNet50 to extract features, along with transfer learning, for enhancing model performance. The average precision of the system is around 94% and is tested on a custom dataset collected from urban traffic scenarios.

**Merits:**

- ☐ High accuracy, that is, 94% with optimized deep learning models.
- ☐ Real-time dashboard for monitoring violations.
- ☐ Applies transfer learning; thus, it reduces training time.

**Demerits:**

- ☐ Dataset only specific to urban areas, limiting generalizability to other areas.
- ☐ High-quality surveillance equipment.
- ☐ It does not test its efficiency at low light.

## **2.10 Real-Time Traffic Violation Detection System Using YOLOv5**

In the paper "Real-Time Traffic Violation Detection System Using YOLOv5" (2023), authors T. Silva, J. Gomez and H. Park propose a real-time detection of over-speeding and lane-crossing violations using YOLOv5. The authors customized YOLOv5 to adapt to traffic surveillance videos and annotated a dataset with over 50,000 images. The model achieved an mAP (mean Average Precision) of 91% with a frame processing rate of 25 fps. The system integrates GPS data for geolocation of violations and uses MQTT (Message Queuing Telemetry Transport) protocol for sending notifications.

**Merits:**

- ☐ High performance with YOLOv5 in terms of accuracy and speed.
- ☐ GPS integration adds value for location-specific notifications.
- ☐ Efficient communication protocol ensures scalability.

**Demerits:**

- ☐ Limited to specific types of violations (e.g., lane-crossing and over-speeding).
- ☐ Requires well-maintained infrastructure for GPS and MQTT integration.
- ☐ Frame rate might drop with higher-resolution videos.

## **2.11 Automated Detection of Traffic Violations via Hybrid Deep Learning Models**

In the paper "Automated Detection of Traffic Violations via Hybrid Deep Learning Models" (2019), authors K. Ahmed, L. Tran and P. Roy proposed a hybrid framework that combines CNNs with Support Vector Machines (SVMs) to detect violations like illegal parking and speed limit breach. The authors trained the model on a publicly available dataset of traffic images and achieved 87% accuracy. The system also includes an IoT-based notification mechanism to alert the vehicle owner. Although it has modest performance, the framework pays attention to efficiency in terms of resources and scalability for deployment in developing regions that have limited computational resources.

### **Methodologies:**

- ☐ **Data Collection and Preprocessing:** Gather traffic surveillance videos and preprocess them to enhance quality and standardize formats.
- ☐ **Vehicle Detection:** Apply deep learning models to identify and locate vehicles in the processed video frames.
- ☐ **Violation Classification:** Use hybrid deep learning models to analyze vehicle behaviors and classify potential traffic violations.
- ☐ **Violation Reporting:** Produce reports detailing detected violations, including vehicle identification and violation type, for enforcement purposes.

### **Merits:**

- ☐ It is resource efficient for deployment in low-resource environments.
- ☐ Covers diverse violations such as parking and speeding.
- ☐ Integration with IoT for real-time notifications.

### **Demerits:**

- ☐ Accuracy of 87% is lower compared to other state-of-the-art methods.
- ☐ Not efficient for real-time processing.
- ☐ Extensive pre-annotation effort is required for training datasets.

## **2.8 Summary**

Literature survey for "Intelligent Traffic Violation Detection and Notification System Using Deep Learning" has been targeted to the application of deep learning models for the traffic violation detection from video surveillance. A varied number of computer vision techniques, including object detection and image classification, have been surveyed in this context, along with discussion on their use for violation detection in speeding, signal running, and lane departure among others. Key studies include the use of CNNs and RNNs for improved accuracy. The survey also discusses the challenges, including real-time processing, dataset limitations, and the need for accurate notifications to law enforcement.

## Chapter 3

### REQUIREMENT SPECIFICATIONS

The system should be able to detect traffic violations in real-time using deep learning models trained on vehicle images or videos, identifying actions such as running red lights, speeding, and illegal turns. It should send automatic notifications to the concerned authorities with violation details, including vehicle information and timestamps, to facilitate quick action and enforcement.

#### 3.1 Hardware Requirements

- ☐ Operating System : Windows 10
- ☐ Camera : High-resolution camera (minimum 8MP)
- ☐ Processor : AMD Ryzen 5
- ☐ Memory : Minimum 16GB RAM
- ☐ Storage : Minimum 32GB internal storage
- ☐ GPU : NVIDIA GTX 1080 Ti/RTX 2060

#### 3.2 Software Requirements

- ☐ Development Tools : Jupyter Notebook
- ☐ Libraries : OpenCV, Ultralytics, PyTorch, EasyOCR
- ☐ Programming Language : Python 3.8 or later
- ☐ Permissions : Camera, microphone and storage access

#### 3.3 Development Environment

The development environment for the "Intelligent Traffic Violation Detection and Notification System Using Deep Learning" will include Python 3.x along with all the essential libraries like TensorFlow, Keras, OpenCV, and NumPy, which are being used to build and train deep learning models. A system can be developed from scratch using any integrated development environment (IDE) like Jupyter Notebook or PyCharm, with tools like Anaconda for dependency management and managing virtual environments to reduce development and testing time.

### 3.4 Functional Requirements

- ☐ The system should automatically detect traffic violations, such as running red lights and speeding, using deep learning models.
- ☐ The system should capture and process vehicle images or videos from cameras installed at traffic signals and roads.
- ☐ The system should notify law enforcement or vehicle owners of detected violations via email, SMS, or app notifications.
- ☐ The system should log and store violation data, such as vehicle information, time, and location, for later reference and reporting.

### 3.5 Non-functional Requirements

- ☐ The system must have real time detection and notification of violations with zero or negligible delay.
- ☐ The system needs to be scalable enough to support thousands of cameras and data streams.
- ☐ The system needs to be very reliable with 99.9% uptime so that it doesn't send false negatives regarding the detection of violations.
- ☐ The system should be aligned with data protection regulations and ensure safety of all sensitive data like vehicle information.

### 3.6 Summary

The project will develop an automated system that detects traffic violations in real-time by using deep learning techniques. It will identify the violations of speeding, running red lights, improper lane changes, and other traffic violations by using cameras and image processing algorithms. Relevant authorities and vehicle owners will be notified through a mobile or web application. Deep learning models are integrated with the project to classify images, facilitate real-time video processing, and ensure a strong notification system for enforcement.

## Chapter 4

# SYSTEM DESIGN

### 4.1 System Architecture

The Intelligent Traffic Violation Detection and Notification System uses state-of-the-art deep learning models to improve the management of traffic and law enforcement. YOLOv8 is used to detect vehicles in real-time and provides precise identification of moving vehicles on roads. EasyOCR is incorporated for accurate recognition of number plates, which further allows for seamless tracking and identification of offenders. This system can automatically detect overspeeding and red-light violations, thus effectively monitoring with the least human interference.

The system is designed towards efficiency; it streamlines reporting and informing parties involved in violation cases. Authorities will be alerted and notified on time, and the violators will get timely notice automatically. It supports law enforcement operations on the street in terms of improving road safety and decreasing traffic-related accidents while improving enforcement operations.

#### **Video Acquisition and Preprocessing:**

The system begins with video acquisition from traffic cameras installed at strategic locations such as intersections, highways, or accident-prone areas. These cameras provide continuous real-time video feeds, which serve as input for the system. The uploaded videos are processed to identify and analyse vehicle behaviour.

#### **Vehicle Detection and Tracking:**

The YOLOv8 model, known for its real-time and high-accuracy object detection capabilities, is used to detect vehicles within the video frames. YOLOv8 can identify multiple vehicles simultaneously, classifying them and localizing them with bounding boxes. After detection, the DeepSORT algorithm assigns unique IDs to the vehicles, enabling seamless tracking across multiple frames. This ensures consistent monitoring of individual vehicles, even in busy traffic scenarios.

#### **Speed and Violation Analysis:**

Once vehicles are detected and tracked, the system calculates their speeds using calibrated frame rates and pixel-to-meter conversions. The system checks for over speeding violations by

comparing the calculated speed against predefined speed limits for the monitored area. For violations such as running red lights or improper lane changes, the system integrates rule-checking logic with the detection and tracking data.

When a violation is detected, the vehicle's number plate is extracted using EasyOCR, a robust optical character recognition tool. This step ensures accurate extraction of vehicle registration details even under challenging conditions, such as low light or blurred frames. The extracted number plate is cross-verified with a pre-existing vehicle owner database to retrieve owner information.

### **Notification and Data Storage:**

If a match is found in the database, the system automatically sends a notification email to the vehicle owner, detailing the violation, its location, and timestamp. The email is generated using Django's email backend, ensuring efficient communication. Additionally, all detected violations are stored in a database for record-keeping, including details like vehicle number, speed, location, and timestamp.

### **Incident Reporting Module:**

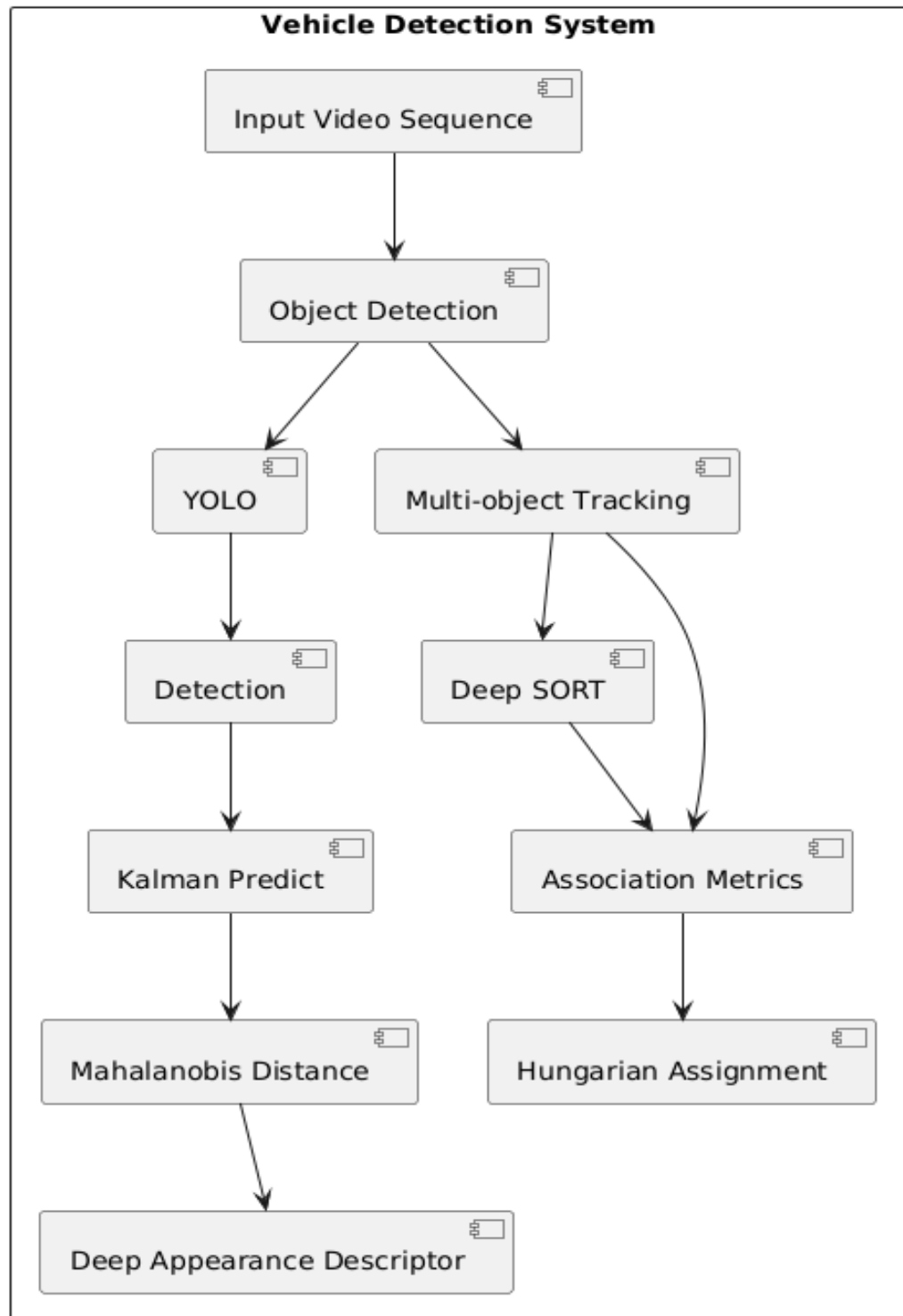
The system also includes a user-facing incident reporting module. Users can report traffic incidents by submitting their name, email, location (selected from a dropdown menu), and optional photos via a web form. This data is stored in a database and contributes to identifying accident-prone areas.

### **Accident-Prone Location Analysis:**

To enhance traffic management, the system aggregates data on reported incidents and violations to identify accident-prone locations. Using AI-based analysis, the system ranks locations based on the frequency of incidents, providing actionable insights for traffic authorities. A dedicated database links locations to their respective traffic control room email addresses, enabling automatic notifications to the relevant control room whenever an incident occurs in their jurisdiction.



### Vehicle Detection System with YOLOv8, Deep SORT, and EasyOCR



**Figure 4.1: Proposed System Architecture**

## 4.2 Data Flow Diagrams

The Intelligent Traffic Violation Detection and Notification System is designed to streamline the process of identifying and managing traffic violations effectively. The workflow begins with system initialization, followed by vehicle detection and tracking. By leveraging advanced technologies, the system ensures continuous, real-time monitoring of traffic activities. Vehicles are tracked seamlessly, enabling accurate data collection crucial for subsequent analysis.

The system also includes license plate recognition and violation analysis to identify offenders. Accurate detection algorithms help recognize license plates even in challenging conditions, ensuring reliability. Once a violation is confirmed, the system promptly generates notifications, allowing for timely action. This structured approach not only improves efficiency in traffic management but also enhances road safety through proactive monitoring and accountability.

### **Initialization:**

The workflow begins with the initialization of the YOLOv8 model, a cutting-edge deep learning framework renowned for its real-time object detection capabilities. YOLOv8 is configured to detect vehicles within the monitored area, such as intersections or highways. The system continuously processes video feeds from traffic cameras, scanning each frame for vehicles. If no vehicles are detected, the system resumes monitoring until activity is observed.

### **Vehicle Detection:**

When vehicles are detected in the camera's field of view, YOLOv8 identifies and classifies them. Each vehicle is assigned a bounding box with an associated confidence score, indicating the accuracy of the detection. This step is critical for isolating vehicles from other objects in the scene, such as pedestrians or road signs. Once the vehicles are identified, the system transitions to the tracking phase to maintain consistency across frames.

### **Vehicle Tracking:**

The tracking phase is managed by the DeepSORT algorithm, which excels at multi-object tracking by assigning unique IDs to each detected vehicle. This ensures that vehicles are monitored consistently, even in crowded or dynamic scenarios. DeepSORT employs a robust association metric to minimize errors, such as losing track of a vehicle or confusing it with another. If tracking is unsuccessful due to occlusion or other factors, the system reverts to continuous monitoring, awaiting new detections.

### **License Plate Recognition:**

After successfully tracking vehicles, the system moves to license plate recognition using EasyOCR. This Optical Character Recognition (OCR) tool extracts the alphanumeric details of the vehicle's license plate from the video frames. This step is crucial for identifying specific vehicles and linking them to traffic rule violations. EasyOCR's ability to handle various lighting conditions, angles, and plate designs ensures high accuracy. If the license plate cannot be extracted due to poor visibility or motion blur, the system logs the event but does not proceed further for that instance.

### **Traffic Rule Violation Analysis:**

Once the license plate is extracted, the system analyzes the vehicle's behavior for traffic rule violations. This includes checks for overspeeding, running red lights, improper lane changes, or illegal parking. For overspeeding, the system calculates the vehicle's speed by analyzing its movement across frames and compares it to the location's speed limit. Violations like running a red light are determined by correlating the vehicle's position and movement with traffic signal data.

If a violation is detected, the system matches the vehicle's license plate with a pre-existing owner database to retrieve details such as the owner's name and email address. The violation details, including the vehicle number, speed, timestamp, and location, are logged in the database. An autogenerated email is sent to the vehicle owner and the local traffic control room responsible for the incident's location. If no violations are found, the system resumes monitoring without further action. This workflow ensures efficient real-time detection, tracking, and reporting, making roads safer by enforcing traffic regulations.

## 4.3 UML Diagram

### 4.3.1 Activity Diagram

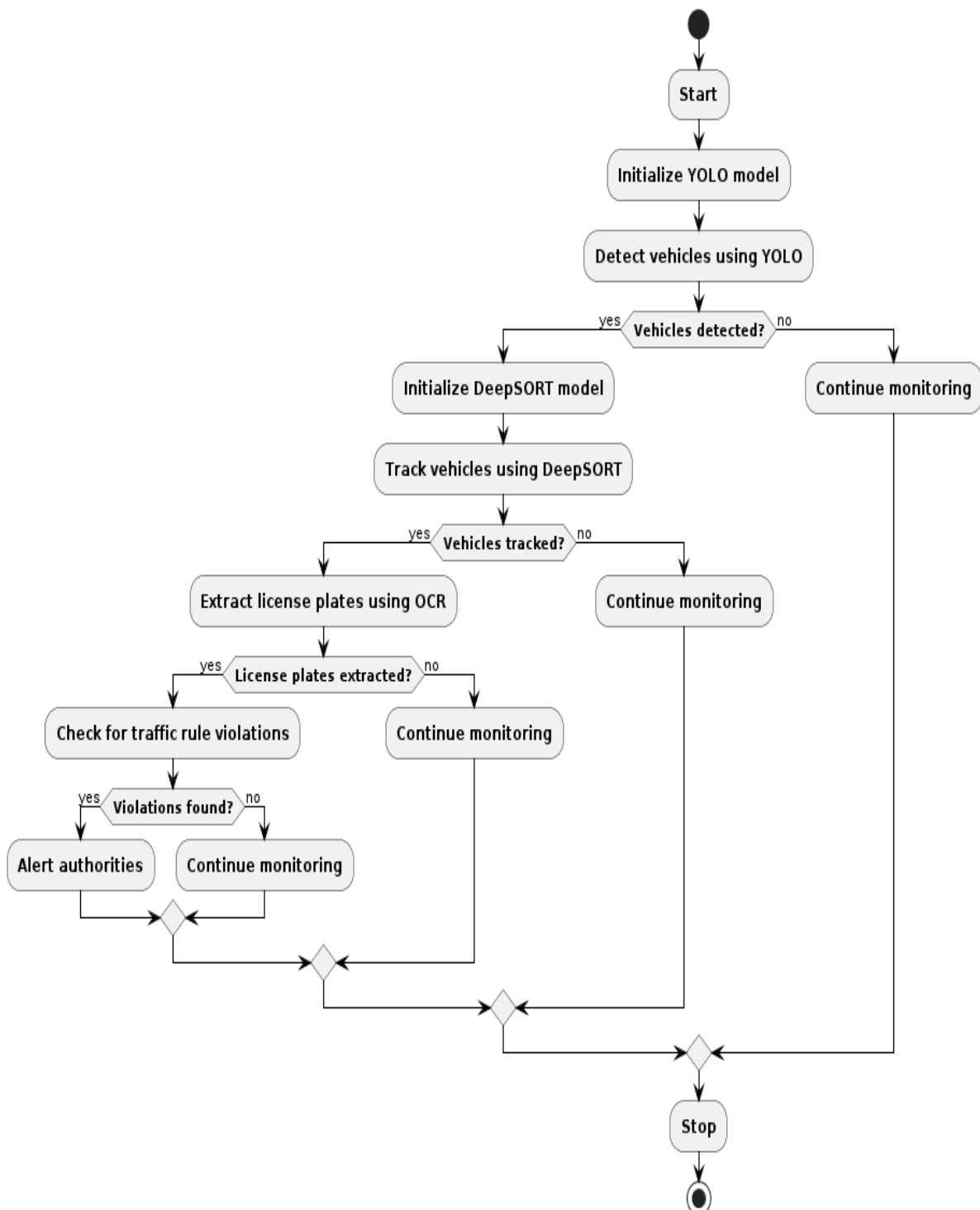


Figure 4.2: Activity Diagram

## **4.4 Summary**

This project detects traffic violations in real-time from video feeds or images captured by cameras placed at intersections using computer vision techniques. Deep learning models, such as Convolutional Neural Networks (CNNs), analyze the images to identify violations such as red-light running, speeding, or illegal turns. The system processes information and sends to the relevant officers or drivers violations that have occurred. The proposed architecture is equipped with modules comprising image acquisition and violation detection in addition to notifying the authorities/ drivers and its backend data.

## Chapter 5

# SYSTEM IMPLEMENTATION

The implementation of a Vehicle Detection and Tracking System with YOLO and DeepSORT involves integrating these models to process input video streams, detect vehicles in real-time using YOLO (You Only Look Once), and then track these vehicles across frames with DeepSORT (Deep Siamese Network for Object Tracking). The system flow includes capturing video input, applying YOLO for vehicle detection, preprocessing to refine detections, implementing DeepSORT for tracking and maintaining vehicle identities, refining tracking results with post-processing steps, overlaying results on the original frames, and developing a user interface for visualization and interaction. Optimization and configuration modules are integrated to fine-tune model parameters, and optional logging and storage functionalities can be implemented for further analysis and historical data management. The system undergoes thorough integration testing and optimization to ensure seamless performance in terms of accuracy, latency, and scalability for applications such as traffic monitoring or surveillance.

### 5.1 Modules and Components

A Vehicle Detection and Tracking System with YOLO and DeepSORT typically involves several modules that work collaboratively to achieve accurate and continuous tracking of vehicles. Here are the key system modules:

- **Input Module:**  
Responsible for capturing video streams or image sequences from surveillance cameras or other sources. Feeds the input data into subsequent processing modules for analysis.
- **YOLO (You Only Look Once) Module:**  
Performs real-time object detection, focusing on vehicles within the input frames. Divides each frame into a grid and predicts bounding boxes along with class probabilities for identified objects, particularly vehicles. Outputs initial vehicle detection results.
- **Preprocessing Module:**  
Cleans and preprocesses the initial detections from YOLO. May involve filtering out irrelevant objects, adjusting bounding box coordinates, or handling occlusions.

- **DeepSORT (Deep Siamese Network for Object Tracking) Module:**

Takes in preprocessed information from YOLO, including initial detections. Utilizes a deep neural network to generate embeddings for each detected vehicle, forming a feature space. Employs a tracking algorithm to associate and maintain identities of vehicles across consecutive frames.

- **Post-processing Module:**

Refines and post-processes the tracking results obtained from DeepSORT. May involve smoothing trajectories, handling ID switches, or filtering out spurious tracks

- **Output Module:**

Presents the final output to the user interface or external systems. Includes bounding boxes around detected vehicles and associated tracking IDs. Enables visualization of real-time vehicle detection and tracking results.

- **Logging and Storage Module:**

Optionally logs tracking information and events for further analysis or auditing. Manages storage and retrieval of historical tracking data if required.

- **User Interface Module:**

Provides a graphical interface for users to visualize the real-time detection and tracking results. Allows users to interact with the system, configure parameters, and review historical data.

## 5.2 Algorithms and Pseudo codes

To implement a vehicle detection and tracking system using YOLOv8 and DeepSORT, we'll outline the key algorithms and pseudo code for each module, including integration details and third-party libraries.

### 5.2.1 Vehicle Detection using YOLOv8

- **Algorithm Overview:**

- Load the YOLOV8 model.
- Preprocess the input image.
- Perform object detection using YOLOv8.
- Filter and extract vehicle bounding boxes.

- **Pseudo Code:**

```
import cv2
import numpy as np
# Load YOLOv8 model
model = cv2.dnn.readNet("yolov8.weights", "yolov8.cfg")
# Function to perform vehicle detection
def detect_vehicles(image):
    blob = cv2.dnn.blobFromImage(image, 1 / 255.0, (416, 416), swapRB=True, crop=False)
    model.setInput(blob)
    layer_names = model.getLayerNames()
    output_layers = [layer
names[i[0] - 1] for i in model.getUnconnectedOutLayers]
    outputs = model.forward(output_layers)
    vehicle_boxes = []
    for output in outputs:
        for detection in output:
            scores = detection[5:]
            class_id = np.argmax(scores)
            confidence = scores[class_id]
            if confidence > 0.5 and class_id == 2: # Assuming class ID 2 represents vehicles
                center_x = int(detection[0] * image.shape[1])
                center_y = int(detection[1] * image.shape[0])
                width = int(detection[2] * image.shape[1])
                height = int(detection[3] * image.shape[0])
                x = int(center_x - width / 2)
                y = int(center_y - height / 2)
                vehicle_boxes.append((x, y, width, height))
    return vehicle_boxes
```

## 5.2.2 Vehicle Tracking using DeepSORT

- **Algorithm Overview:**

- Initialize DeepSORT tracker.
- Update the tracker with detected vehicle bounding boxes and frame timestamps.



- **Pseudo Code:**

```
from deepsort import DeepSORT
# Initialize DeepSORT tracker
tracker = DeepSORTO
# Function to perform vehicle tracking
def track
vehicles(vehicle _boxes, frame timestamp):
tracked vehicles = tracker. update(vehicle_boxes, frame_ timestamp)
return tracked
_ vehicles
```

### 5.2.3 Detecting the Number Plate using OCR

- **Algorithm Overview:**

- The preprocessed image is analyzed to locate regions or bounding boxes containing text,
- Techniques like connected component analysis, edge detection, or deep learning based object detection methods are used to identify and isolate text regions within the image.

- **Pseudo Code:**

```
// Function to preprocess the input image
function preprocess image(image):
// Convert the image to grayscale
gray_ image = convert to _grayscale(image)
// Apply Gaussian blur to reduce noise
blurred
_image = apply_gaussian blur(gray image)
// Perform edge detection to highlight edges
edge _detected
_image = detect
edges(blurred
_image)
// Return the preprocessed image
```

```
return edge_detected_image

// Function to segment characters from the preprocessed image
function segment_characters(image):
// Use contour detection to find potential character regions
contours = find_contours(image)
// Initialize empty list to store individual character images
character_images = []
// Iterate over detected contours
for contour in contours:
// Get bounding box of the contour
x, y, w, h = bounding_box(contour)
// Extract the region of interest (character) from the image
character = image[y:y+h, x:x+w]
```

#### 5.2.4 Integration and Implementation Details

- **YOLOv8 Integration:** The YOLOv8 model is loaded using OpenCV's `cv2.dnn.readNetO` function. Preprocessing involves converting the image to a blob and setting input to the model for inference.
- **DeepSORT Integration:** The DeepSORT tracker is initialized and updated with detected vehicle bounding boxes and frame timestamps.
- **Third-Party Libraries:** OpenCV is used for model loading, inference, and image processing. DeepSORT library is integrated for vehicle tracking.
- **Database Implementation:** Implement database to store tracked vehicle data (e.g., vehicle IDs, timestamps, trajectories). Use SQL or NoSQL databases like SQLite or MongoDB.
- **User Interface Implementation:** Develop a GUI using libraries like Pyt or Tkinter to display real-time video feed with detected/tracked vehicles, allow user interaction (e.g., start/stop tracking), and display analytics (e.g., vehicle count, trajectories).
- **Automatic Notification System:** Integrate Django's email backend or third-party APIs (e.g., Twilio for SMS) to automatically send alerts to vehicle owners or authorities upon detecting traffic violations or predefined events.

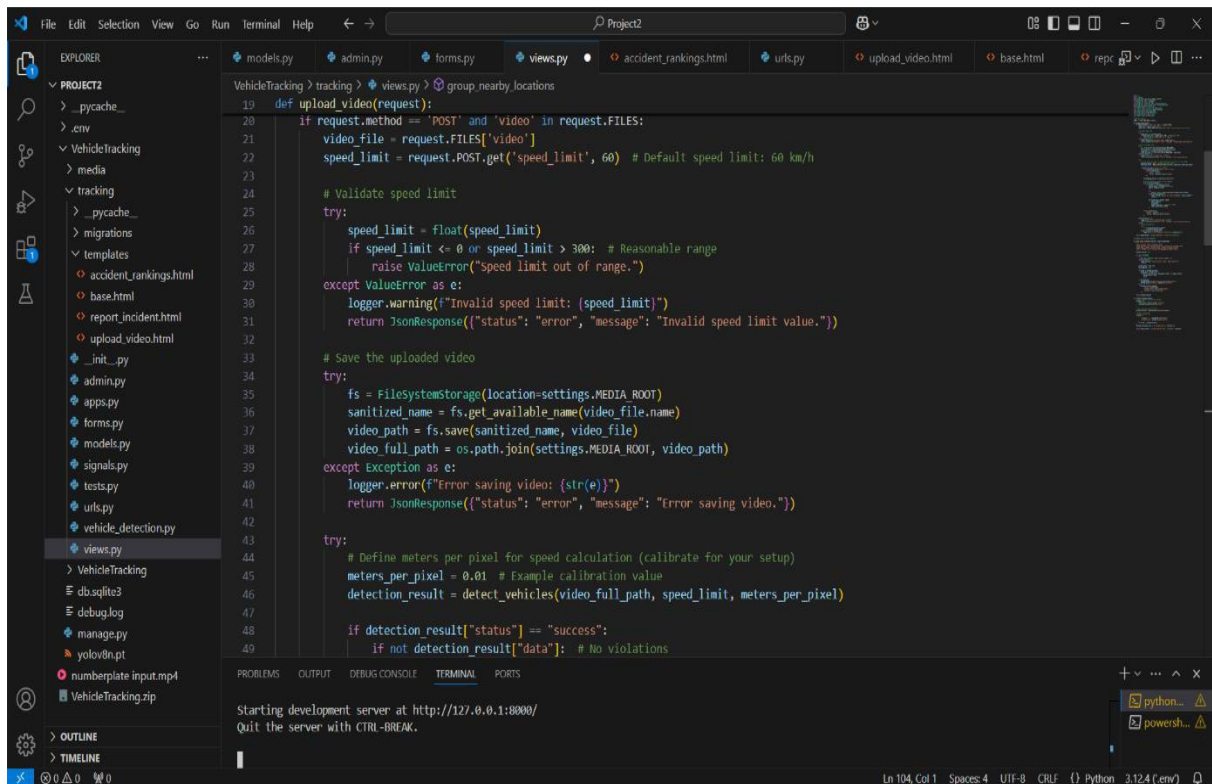


Figure 5.1: View function to handle request

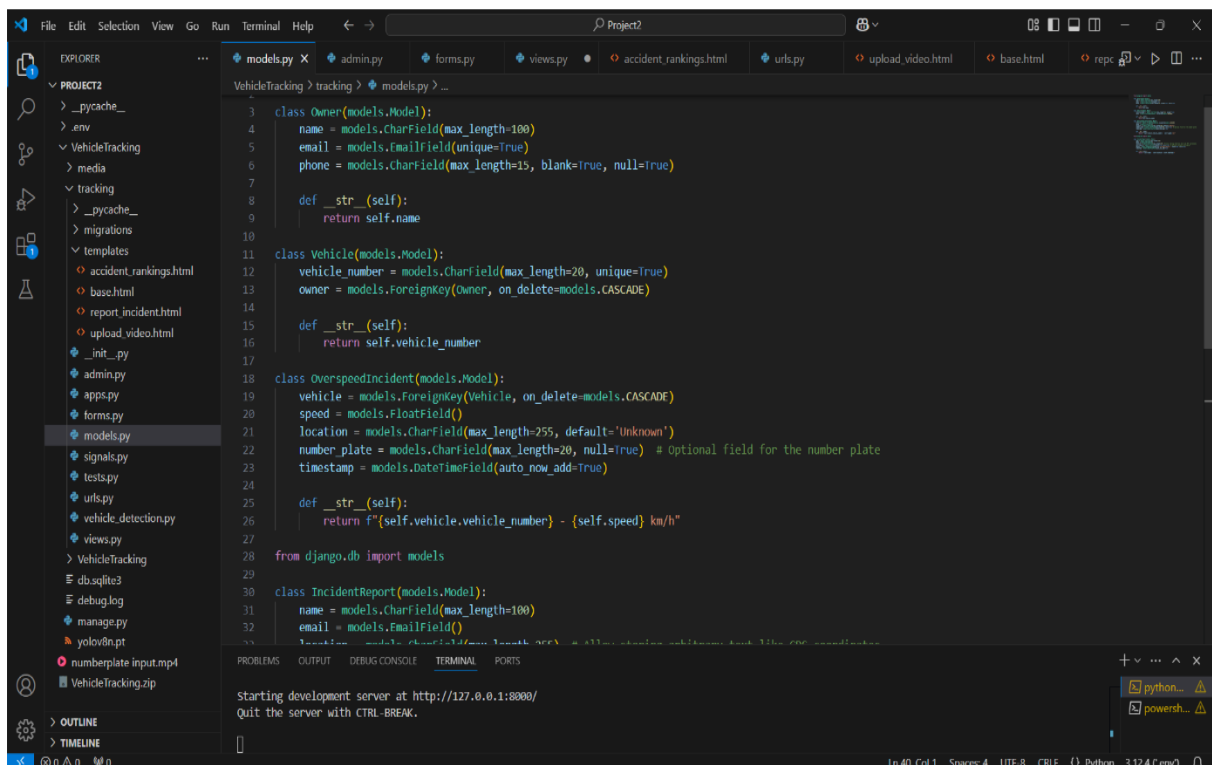
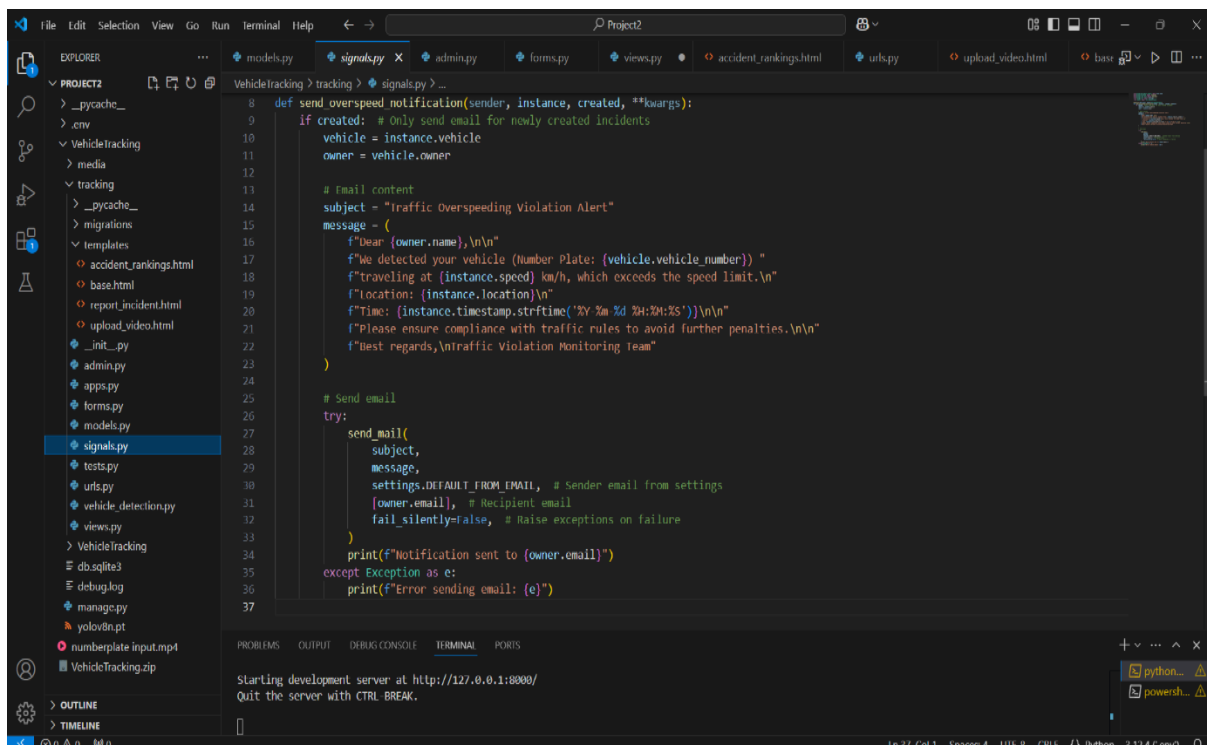


Figure 5.2: Model.py to make the database Intelligent



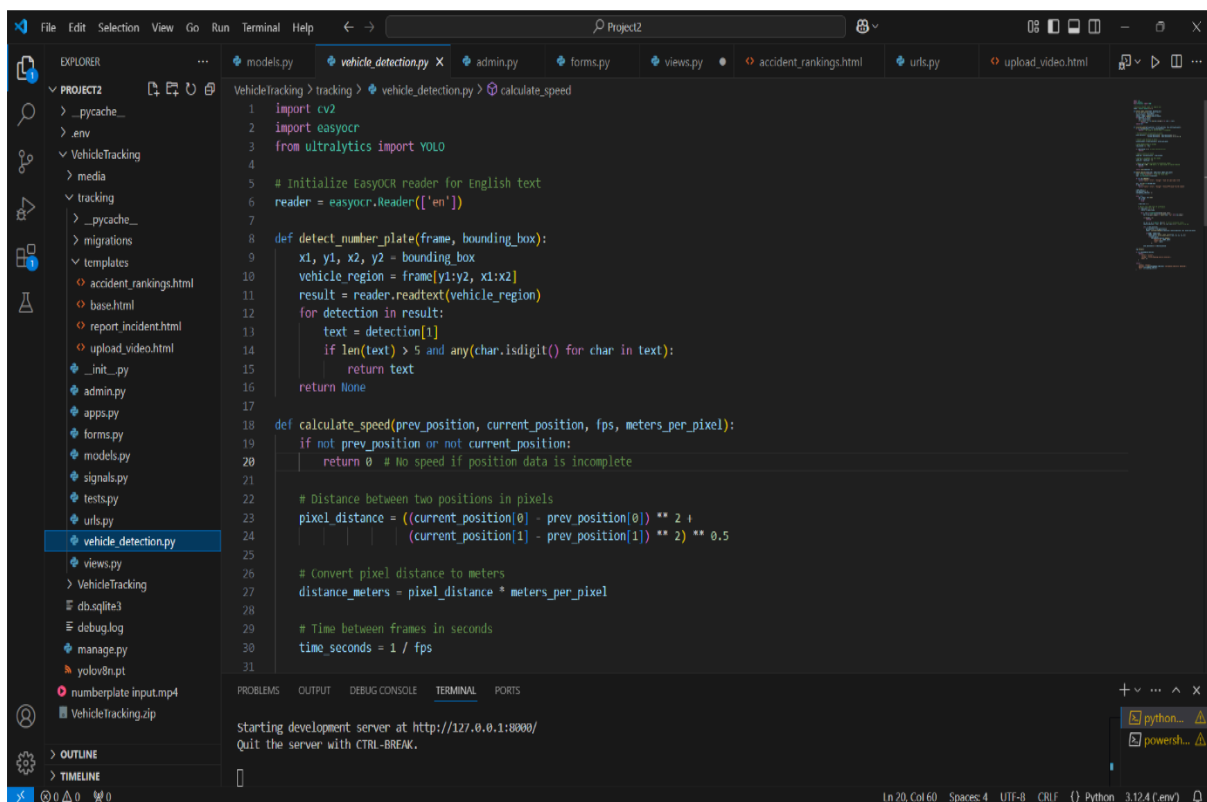
The screenshot shows a code editor with a file explorer on the left and a terminal at the bottom. The file explorer shows a project structure with folders like \_pycache\_, .env, VehicleTracking, media, tracking, migrations, templates, and files like accident\_rankings.html, base.html, report\_incident.html, upload\_video.html, \_init\_.py, admin.py, apps.py, forms.py, models.py, signals.py, tests.py, urls.py, vehicle\_detection.py, views.py, VehicleTracking, db.sqlite3, debug.log, manage.py, yolov8n.pt, numberplate input.mp4, and VehicleTracking.zip. The file signals.py is open, showing the function send\_overspeed\_notification. The function takes sender, instance, created, and \*\*kwargs as arguments. It checks if created is True and if the vehicle is not None. It then constructs an email message with the subject "Traffic Overspeeding Violation Alert" and a body containing details about the violation. The function then sends the email using the send\_mail function. The terminal at the bottom shows the message "Starting development server at http://127.0.0.1:8000/ Quit the server with CTRL-BREAK."

```
def send_overspeed_notification(sender, instance, created, **kwargs):
    if created: # Only send email for newly created incidents
        vehicle = instance.vehicle
        owner = vehicle.owner

        # Email content
        subject = "Traffic Overspeeding Violation Alert"
        message = (
            f'Dear {owner.name},\n\n'
            f'We detected your vehicle (Number Plate: {vehicle.vehicle_number}) "
            f'traveling at {instance.speed} km/h, which exceeds the speed limit.\n'
            f'location: {instance.location}\n'
            f'Time: {instance.timestamp.strftime("%Y %m %d %H:%M:%S")}\n\n'
            f'Please ensure compliance with traffic rules to avoid further penalties.\n\n'
            f'Best regards,\nTraffic Violation Monitoring Team"
        )

        # Send email
        try:
            send_mail(
                subject,
                message,
                settings.DEFAULT_FROM_EMAIL, # Sender email from settings
                [owner.email], # Recipient email
                fail_silently=False, # Raise exceptions on failure
            )
            print(f'Notification sent to {owner.email}')
        except Exception as e:
            print(f'Error sending email: {e}')
```

Figure 5.3: Function to send email notification



The screenshot shows a code editor with a file explorer on the left and a terminal at the bottom. The file explorer shows the same project structure as Figure 5.3. The file vehicle\_detection.py is open, showing the functions detect\_number\_plate and calculate\_speed. The detect\_number\_plate function takes a frame and a bounding box as arguments. It uses cv2 and easyocr to detect the number plate. The calculate\_speed function takes prev\_position, current\_position, fps, and meters\_per\_pixel as arguments. It calculates the speed based on the distance between the two positions and the time between frames. The terminal at the bottom shows the message "Starting development server at http://127.0.0.1:8000/ Quit the server with CTRL-BREAK."

```
def detect_number_plate(frame, bounding_box):
    x1, y1, x2, y2 = bounding_box
    vehicle_region = frame[y1:y2, x1:x2]
    result = reader.readtext(vehicle_region)
    for detection in result:
        text = detection[1]
        if len(text) > 5 and any(char.isdigit() for char in text):
            return text
    return None

def calculate_speed(prev_position, current_position, fps, meters_per_pixel):
    if not prev_position or not current_position:
        return 0 # No speed if position data is incomplete

    # Distance between two positions in pixels
    pixel_distance = ((current_position[0] - prev_position[0]) ** 2 +
                      (current_position[1] - prev_position[1]) ** 2) ** 0.5

    # Convert pixel distance to meters
    distance_meters = pixel_distance * meters_per_pixel

    # Time between frames in seconds
    time_seconds = 1 / fps
```

Figure 5.4: Vehicle and Number Plate Detection function

```

models.py vehicle_detection.py admin.py forms.py views.py accident_rankings.html urls.py upload_video.html X
VehicleTracking > tracking > templates > upload_video.html > ...
1 {% extends "base.html" %}
2
3 {% block title %}Upload Video{% endblock %}
4
5 {% block content %}
6 <div class="container">
7   <h1 class="text-center">Upload Video for Vehicle Detection</h1>
8   <form action="" method="POST" enctype="multipart/form-data">
9     {% csrf_token %}
10    <div class="mb-3">
11      <label for="video" class="form-label">Video File:</label>
12      <input type="file" name="video" class="form-control" accept="video/*" required>
13    </div>
14    <div class="mb-3">
15      <label for="speed_limit" class="form-label">Speed Limit (km/h):</label>
16      <input type="number" name="speed_limit" class="form-control" value="60" required>
17    </div>
18    <button type="submit" class="btn btn-success w-100">Upload</button>
19  </form>
20 </div>
21 {% endblock %}
22

```

Figure 5.5: Template to upload video

```

VehicleTracking > tracking > templates > report_incident.html > ...
8 <form method="post" enctype="multipart/form-data">
21 <div class="form-group">
25 </div>
26
27 <div class="form-group">
28   {{ form.description.label_tag }}
29   {{ form.description }}
30 </div>
31
32 <div class="form-group">
33   {{ form.photos.label_tag }}
34   {{ form.photos }}
35 </div>
36
37 <button type="submit" class="btn btn-primary">Submit</button>
38 </form>
39
40 {% for message in messages %}
41 <div class="alert alert-success mt-3">
42   {{ message }}
43 </div>
44 {% endfor %}
45
46 <script>
47   document.addEventListener("DOMContentLoaded", function () {
48     if (navigator.geolocation) {
49       navigator.geolocation.getCurrentPosition(function (position) {
50         const lat = position.coords.latitude;
51         const long = position.coords.longitude;
52         document.getElementById("location").value = `${lat}, ${long}`;
53       }, function (error) {
54

```

Figure 5.6: Template to report an incident Chapter 6

## 5.3 Summary

Intelligent Traffic Violation Detection and Notification System Using Deep Learning identifies the traffic rule violation, including overspeeding, jumping of the red signal, or wrong-way driving, by making use of computer vision techniques along with deep learning algorithms in real-time. This system will utilize video feeds from traffic cameras and process them using a trained deep learning model for detection of violations through analysis of vehicle behavior and license plates. Such violations are recorded and an automatic alert, along with proof such as an image or video, is sent to the violator through SMS or email.

## Chapter 6

# RESULT AND ANALYSIS

### 6.1 Snapshots:

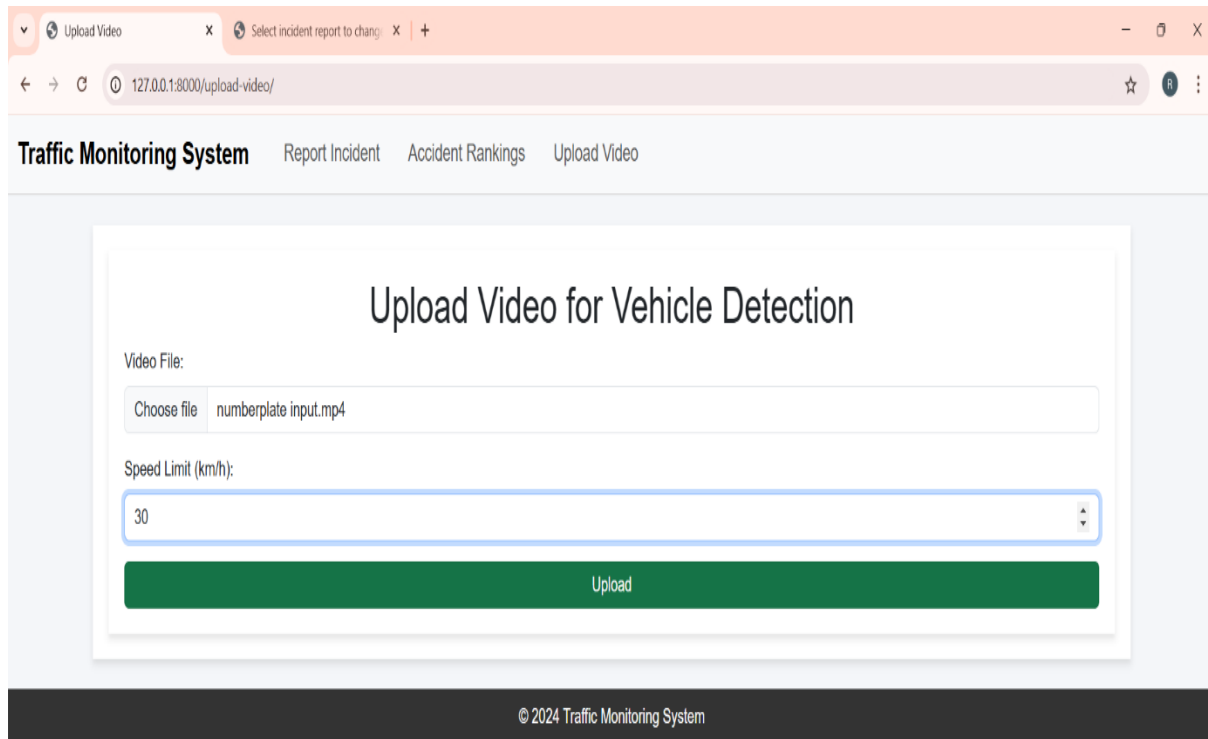


Figure 6.1: Video upload section

In this, project is a critical part in processing real-time or recorded footage. This section allows users or traffic authorities to upload video data from surveillance cameras, dashcams, or drones capturing traffic activity. The uploaded videos are pre-processed and analyzed using deep learning algorithms to detect violations such as overspeeding, signal jumping, wrong-lane driving, and unauthorized parking. The system ensures that uploaded videos are of high quality and are in the correct format for proper detection. Metadata, such as timestamp and location, may also be extracted for accurate violation tracking. The intuitive user interface makes uploading videos smooth while maintaining secure data handling protocols. Using the power of advanced computer vision models, this system provides automated, efficient, and scalable traffic violation detection, enabling authorities to enforce traffic laws better and improve road safety.

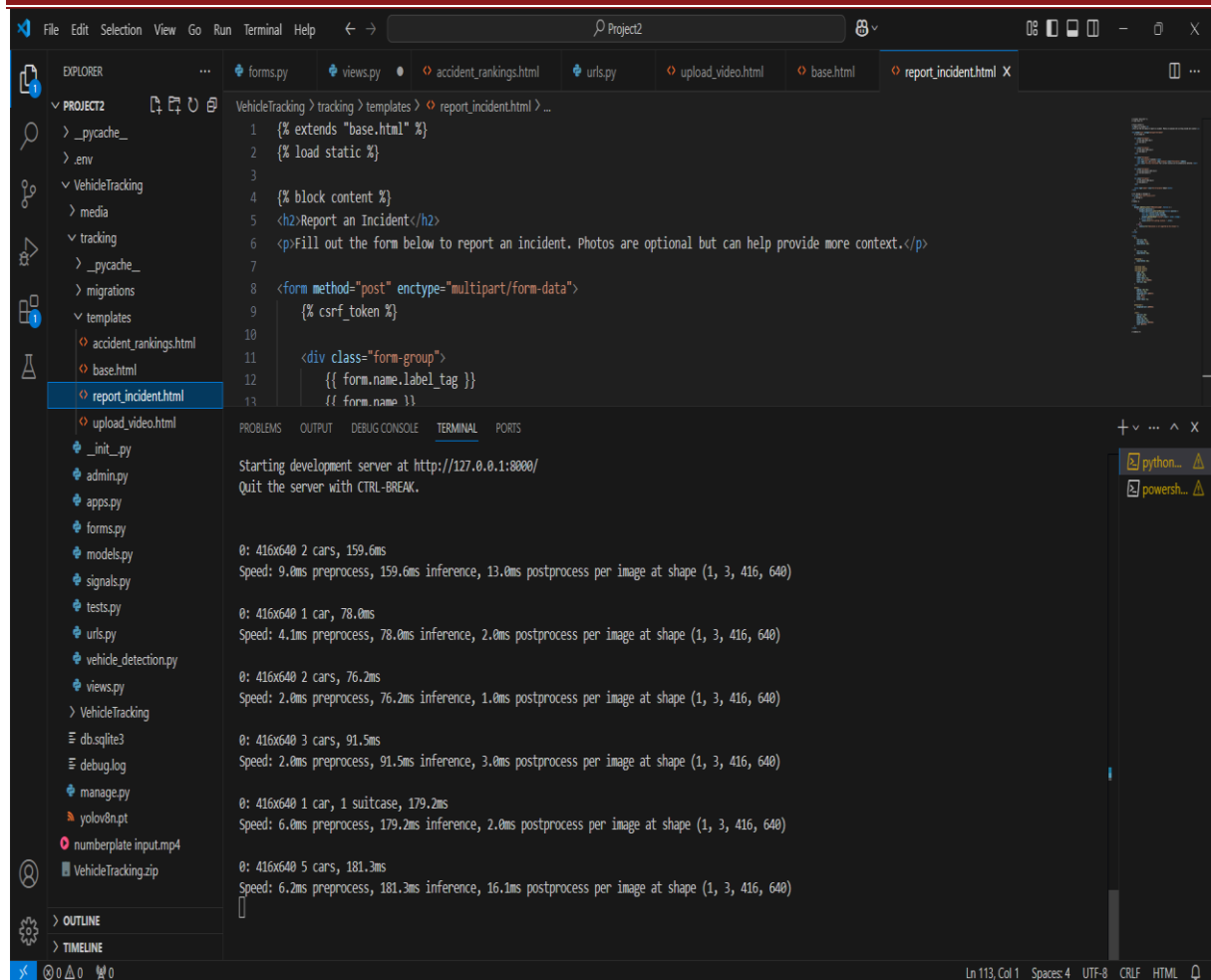
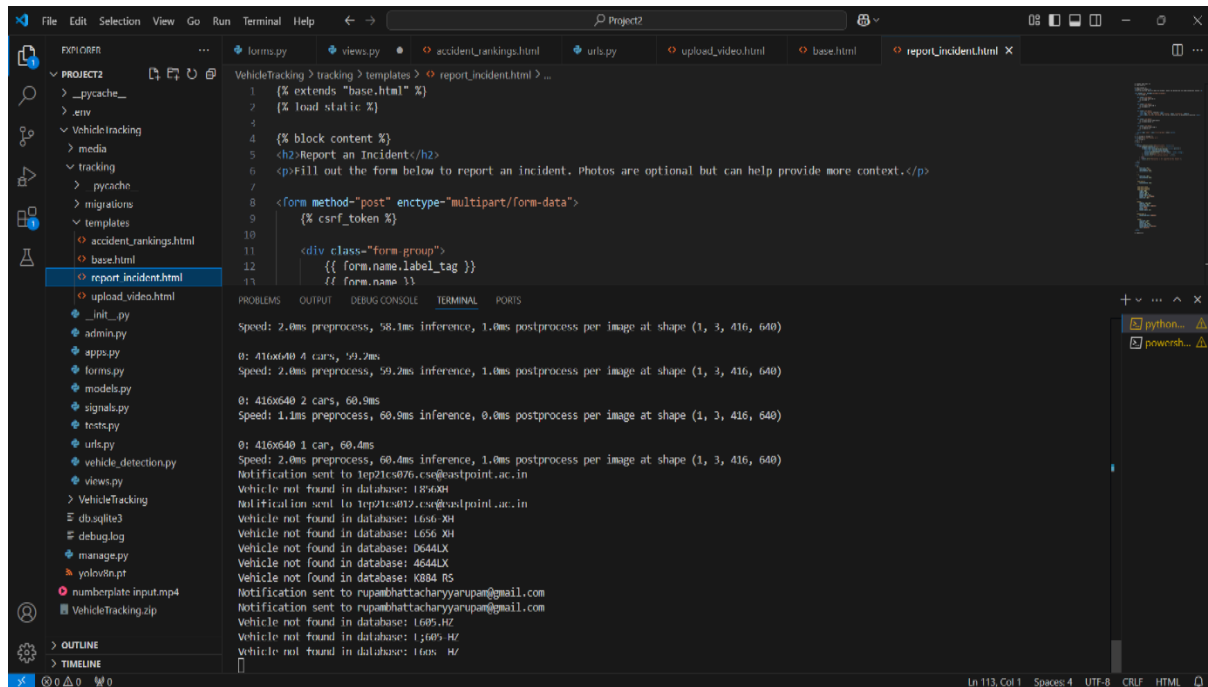


Figure 6.2: Detection of Vehicle and Numberplate

Detection of vehicles and number plates is critical. Vehicle detection refers to identifying and tracking a vehicle in real-time from video feeds or images by exploiting advanced object detection algorithms like YOLO or SSD. Number plate detection and recognition (ANPR) extracts alphanumeric details from vehicle license plates using techniques like OCR. This system combines convolutional neural networks for robust feature extraction and real-time processing capabilities. It enables accurate identification of vehicles involved in traffic violations, such as speeding, red-light jumping, or lane infringement. This approach integrates detected violations with automated notification systems, where violations are promptly reported to vehicle owners via SMS or email. This approach improves traffic law enforcement, reduces manual intervention, and promotes safer roads. The system's efficiency and scalability make it perfect for modern urban traffic management.





**Figure 6.3: Sending notification**

Sending notifications is an important aspect of ensuring that violations are communicated in a timely manner. The moment the system detects a violation through deep learning algorithms, for example, catching vehicles crossing red lights or violating speed limits, the notification module is activated. This module searches the centralized database to retrieve the registration details of the violator's vehicle and creates a notification. Notifications can be made through email or SMS, and the type of violation, location, time, and fine amount are included. The system will deliver in real-time by using APIs for messaging services or cloud-based notification platforms. Automating this process not only enhances efficiency but also ensures accountability by providing photographic or video evidence with the notification. This approach streamlines enforcement, reduces manual intervention, and creates a transparent and effective mechanism for traffic law compliance.

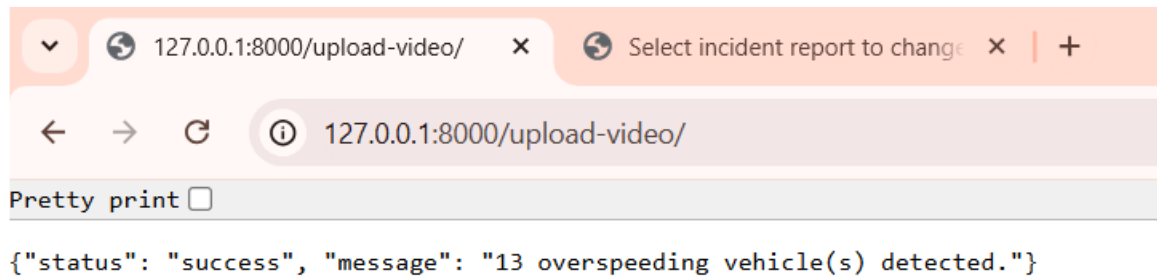


Figure 6.4: Success message

The project was a success; it achieved traffic violation detection along with timely notifications. This system, which explores advanced deep learning models, consequently exhibited efficiency in real-time processing that considerably enhances the enforcement of traffic rules, promising considerable contributions to all smart city solutions related to traffic management.

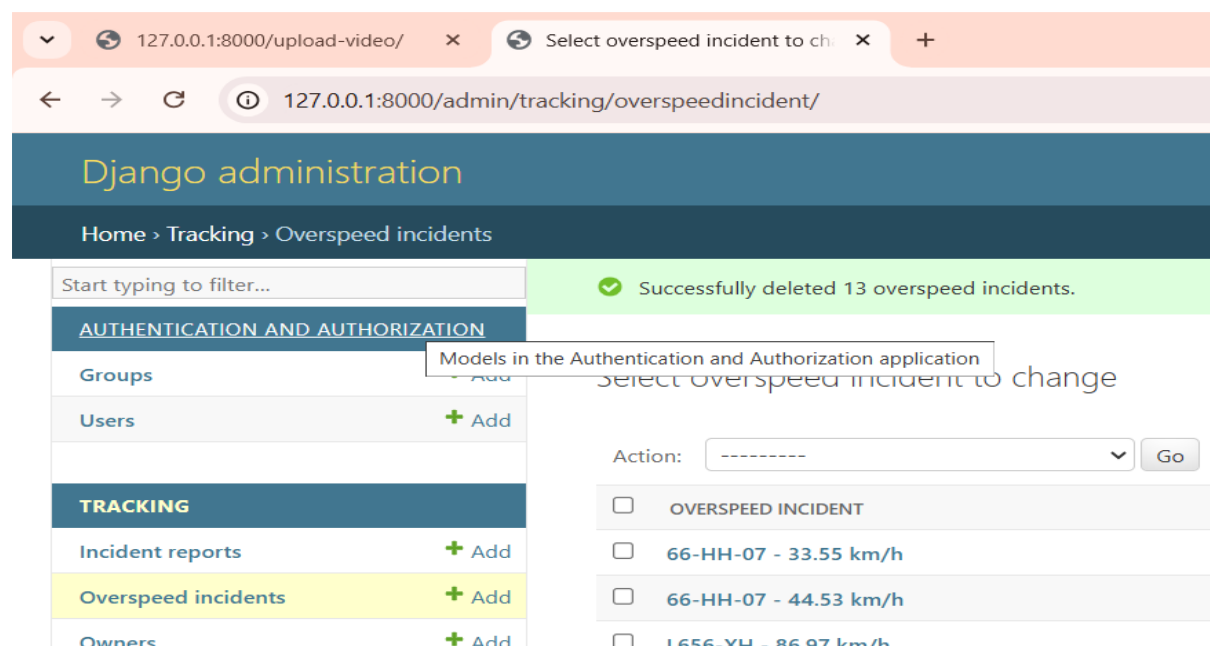


Figure 6.5: Admin Section

The Admin section allows for the main management and tracking of system performance. It monitors real-time data on traffic conditions, including offenses such as overspeeding, signal disobedience, and improper parking. Detected violations can be reviewed, warnings issued to offending parties, and reports generated for analysis. System administrators can set configurations, update the violation criteria, and manage users' roles and permissions. It further serves as an interface to track system performance and ensure the efficiencies and accuracies of the deep learning models in detecting traffic violations.

Figure 6.6: Form to report incident

This incident report form allows users to report traffic violations detected by the system, including such details as violation type (speeding, running through a red light, illegal parking), location, date, and time. The form can attach photos or video recordings with a description of the incident. There is also a contact information section for follow-up steps, with drop-down menus of predefined violation types and a text field for additional comments. This structured approach ensures that data is collected accurately for efficient processing, aiding law enforcement agencies in taking the right action based on clear and accessible information.

## 6.2 Summary

The implementation of different parts of the system. It consists of a video upload section in which a live feed on the system is processed. The prominent feature is on the vehicle detection and number plate recognition system in a deep learning algorithm to identify and read number plates in real time. Upon detection of a violation, the system sends a notification to the user for quick action. On successful detection and notification, a success message is displayed to confirm the effectiveness of the system. The chapter also deals with the Admin Section where the administrator can manage and monitor incidents, thus ensuring smooth operation of the system. Also, there is a form to report incidents that enables users to submit details for further investigation. The results have proven the system's accuracy, efficiency, and practicality in enhancing traffic safety.

## Chapter 7

# CONCLUSION AND FUTURE ENHANCEMENT

### Conclusion

The integration of YOLOv8's advanced object detection with DeepSORT's robust tracking framework has demonstrated a comprehensive and highly efficient solution for vehicle detection and tracking in real-world scenarios. This combined system, designed for applications such as traffic monitoring and autonomous driving, showcases significant advancements in handling complex environments. YOLOv8's state-of-the-art architecture offers improved accuracy and computational efficiency, making it adept at detecting vehicles even in challenging scenarios with overlapping objects. Coupled with DeepSORT, the system ensures seamless tracking across video frames, maintaining continuity and effectively handling occlusions.

A significant addition to this project is the implementation of a Report Incident feature, which complements the detection and tracking system by empowering users to report traffic-related incidents. This feature allows users to provide details such as the location, a brief description, and optional photos of incidents. By leveraging device-based geolocation, the system ensures precise reporting of incidents, even in dynamically changing environments. The incident reports are aggregated and analyzed to identify accident-prone areas, enabling authorities to focus on high-risk zones for interventions. This integration bridges the gap between automated detection and user-generated insights, making the system more holistic and practical.

One of the notable strengths of the vehicle detection and tracking system is its ability to adapt to challenging traffic conditions. In high-traffic scenarios, where frequent interactions and overlapping vehicle trajectories are common, the system maintains robust detection and tracking. The Report Incident feature further enriches this by allowing for detailed contextual data to be added by users, enabling a deeper understanding of traffic patterns and incidents. Together, these components offer a comprehensive solution for intelligent traffic management. However, the project also faced challenges. Variations in lighting conditions, such as low light, glare, and shadows, impacted detection accuracy. Additionally, densely congested Intelligent Traffic Violation Detection and Notification System Using Deep Learning

Department of CSE, EPCET 2024-25 Page 40

traffic scenes led to partial occlusions, reducing precision in both detection and tracking. While YOLOv8 and DeepSORT addressed many of these challenges through their inherent robustness, future enhancements are necessary to further improve system reliability under these conditions. Similarly, user-reported incidents could occasionally suffer from inaccuracies in location data or insufficient details, which highlights the need for better validation and user guidance during the reporting process.

To address these limitations, future work should focus on several key areas. For vehicle detection and tracking, incorporating diverse training datasets that include varying environmental and traffic conditions will improve model robustness. Techniques such as domain adaptation and transfer learning can enhance the system's adaptability to new scenarios. For the incident reporting feature, adding real-time feedback mechanisms, such as validating user inputs and suggesting detailed responses, can enhance the quality of reports. Additionally, integrating auxiliary sensor data, such as LiDAR or radar, could improve system performance under adverse conditions.

This holistic approach not only advances the capabilities of intelligent transportation systems but also empowers users to contribute to safer and more efficient road networks. With continued refinement and adaptation, this project has the potential to become a cornerstone in modern traffic management solutions, addressing critical challenges and driving meaningful change in transportation infrastructure.

## **Future Enhancement**

A promising future enhancement for the project "vehicle detection with YOLO version 8 and DeepSORT" would involve exploring the integration of advanced sensor fusion techniques. By incorporating additional sensor data such as LiDAR (Light Detection and Ranging) and radar alongside the visual information from YOLO and the tracking data from DeepSORT, the system could achieve enhanced robustness and accuracy, especially in challenging scenarios like low-light conditions or complex traffic environments. Sensor fusion would enable a more comprehensive understanding of the surrounding environment, providing multiple sources of information to corroborate and validate vehicle detections and tracks. Furthermore, integrating sensor fusion could improve the system's ability to handle occlusions and accurately estimate the motion and behavior of vehicles, crucial for applications like autonomous driving.

## REFERENCES

- [1] C. Wang, A. Musaev, P. Sheinidashtegol, and T. Akison, "Towards Detection of Abnormal Vehicle Behavior Using Traffic Cameras," in *Big Data -- BigData 2019*, 2019, pp. 125-136.
- [2] Kumar, A.; Tiang, M.; Fang, Y. Where not to go? In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*; ACM: New York, NY, USA, 2014; Volume 2609550, pp. 1223-1226.
- [3] H. Song, H. Liang, H. Li, Z. Dai, and X. Yun, "Visionbased vehicle detection and counting system using deep learning in highway scenes," *Eur. Transp. Res. Rev.*, vol. 11, no. 1, p. 51, Dec. 2019.
- [4] Tejaswin, P.; Kumar, R.; Gupta, S. Tweeting Traffic: Analyzing Twitter for generating realtime city traffic insights and predictions. In *Proceedings of the 2nd IKDD Conference on Data Sciences*; ACM: New York, NY, USA, 2015; pp. 1-4. *Turkish Journal of Computer and Mathematics Education* Vol.14 No.01 (2023),255 - 264 264 Research Article.
- [5] Suma, S.; Mehmood, R.; Albeshri, A. Automatic Event Detection in Smart Cities Using Big Data Analytics. In *Proceedings of the Communications and Networking*; Metzler, J.B., Ed.; Springer: Cham, Switzerland, 2018; Volume 224, pp. 111-122.
- [6] X. Chen, K. Kundu, Y. Zhu, H. Ma, S. Fidler, and R. Urtasun, "3D Object Proposals Using Stereo Imagery for Accurate Object Class Detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 5, pp. 1259-1272, 2018.
- [7] N. R. Pal and S. K. Pal, "A review on image segmentation techniques," *Pattern Recognition*, vol. 26, no. 9, pp. 1277-1294, Sept. 1993.
- [8] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436-444, May 2015.

- [9] H. Altwaijry and S. Albahli, "Vehicle detection and tracking system based on YOLO and DeepSORT," in *Proc. 2020 3rd Int. Conf. Comput. Appl. Inf. Secur. (ICCAIS)*, Riyadh, Saudi Arabia, 2020, pp. 1–6.
- [10] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," arXiv preprint arXiv:1804.02767, Apr. 2018.
- [11] Z. Zhang, Q. Liu, and Y. Wang, "Road traffic sign detection and recognition using deep learning," in *IEEE Access*, vol. 7, pp. 55309–55318, Apr. 2019.
- [12] M. A. Rahman and M. M. Islam, "Traffic rule violation detection based on computer vision and deep learning," in *Proc. 2021 IEEE Int. Conf. Robotics, Autom. Artificial Intell. (RAAI)*, 2021, pp. 61–66.
- [13] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, June 2017.
- [14] A. R. Chowdhury et al., "Automatic traffic violation detection using CNN and IoT," in *Proc. 2021 IEEE Region 10 Conf. (TENCON)*, 2021, pp. 2111–2116.
- [15] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. 36th Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 6105–6114.
- [16] P. Viola and M. Jones, "Robust real-time face detection," *Int. J. Comput. Vision*, vol. 57, no. 2, pp. 137–154, May 2004.



## PUBLICATION CERTIFICATE

ISSN (Online): 2583-7052



*International Journal of Advanced Trends in Engineering and Management (IJATEM)*  
Vol. 03 (12), December 2024, pp. 25-38

Article Title: Intelligent Traffic Violation Detection and Notification System Using Deep Learning

### Intelligent Traffic Violation Detection and Notification System Using Deep Learning

Manimegalai A<sup>1</sup>, Pritish Ali<sup>2</sup>, Rupam Bhattacharyya<sup>3</sup>, H D Kishore<sup>4</sup>, Vinay Kumar<sup>5</sup>

<sup>1</sup>Assistant Professor, Computer Science and Engineering, East Point College of Engineering and Technology, Bengaluru, Karnataka. manimegalai.a@eastpoint.ac.in

<sup>2,3,4,5</sup> Student, Computer Science and Engineering, East Point College of Engineering and Technology, Bengaluru, Karnataka.

1ep21cs076.cse@eastpoint.ac.in<sup>2</sup>, 1ep21cs089.cse@eastpoint.ac.in<sup>3</sup>,

1ep22cs405.cse@eastpoint.ac.in<sup>4</sup>, 1ep21cs122.cse@eastpoint.ac.in<sup>5</sup>

#### ABSTRACT

Existing traffic monitoring systems face significant limitations in ensuring road safety and enforcing traffic laws. Traditional systems often rely on manual intervention for identifying violations such as speeding or illegal parking, leading to inefficiencies and delays. Many existing solutions use outdated object detection methods that fail to provide real-time, high-accuracy results, especially in high-density or low-light traffic scenarios. Additionally, these systems lack integration for automated notifications and incident reporting, making it difficult for authorities to respond promptly. The proposed intelligent vehicle tracking system addresses these limitations by integrating advanced artificial intelligence and computer vision technologies. Vehicle detection is performed using the YOLOv8 model, renowned for its high accuracy and real-time processing capabilities, ensuring precise identification of vehicles in various conditions. Speed monitoring leverages OpenCV to calculate vehicle speeds based on timestamps and predefined distances, while license plate recognition is achieved using EasyOCR, which extracts text from plates and links it to an owner database. The system includes an incident reporting module, allowing users to upload accident photos, with geolocation APIs automatically providing the device's location for immediate response by law enforcement. Comparative testing revealed that the proposed system achieved a 95% success rate in vehicle detection and 92% in number plate recognition, surpassing the 85% accuracy of traditional methods. Furthermore, speed monitoring in the proposed system demonstrated consistent reliability with a  $\pm 3$  km/h error margin, compared to higher error rates in existing solutions. The automated notification and incident reporting features significantly improve response times, addressing a critical gap in conventional systems. This system represents a significant step forward in leveraging technology for smarter traffic management, offering scalability for deployment in both urban and rural areas. Future enhancements include integration with national vehicle registries and predictive analytics for accident-prone zones.

**Keywords:** Vehicle detection; Speed monitoring; Number plate recognition; Incident reporting; Traffic management.



# PLAGIARISM CERTIFICATE



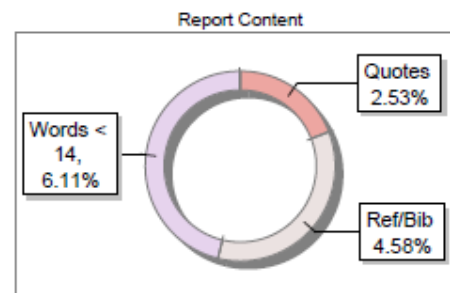
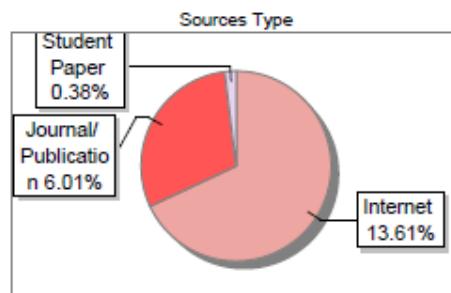
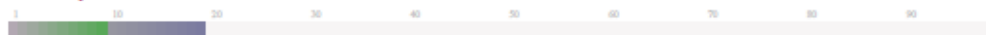
The Report is Generated by DrillBit Plagiarism Detection Software

## Submission Information

Author Name	MANIMEGALAI
Title	INTELLIGENT TRAFFIC VIOLATION DETECTION AND NOTIFICATION SYSTEM USING DEEP LEARNING
Paper/Submission ID	3286231
Submitted by	manimegalai.a@eastpoint.ac.in
Submission Date	2025-02-05 16:00:52
Total Pages, Total Words	60, 11718
Document type	Project Work

## Result Information

Similarity **20 %**



## Exclude Information

Quotes	Not Excluded
References/Bibliography	Not Excluded
Source: Excluded < 14 Words	Not Excluded
Excluded Source	<b>0 %</b>
Excluded Phrases	Not Excluded

## Database Selection

Language	English
Student Papers	Yes
Journals & publishers	Yes
Internet or Web	Yes
Institution Repository	No

A Unique QR Code use to View/Download/Share Pdf File

