International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 14, Issue 1, 2026

Helmet And Seatbelt Detection System

Mohd Basit Mohiuddin[1],K Likhitha[2],G Sneha[3],B Srikavya[4]

[1]Assistant Professor; Department Of Computer Science And Engineering(AI & Ml), Bhoj Reddy Engineering College For Women, Hyderabad, India.

[2,3,4]B.Tech Students; Department Of Computer Science And Engineering(AI & Ml), Bhoj Reddy Engineering College For Women, Hyderabad, India.

Mail Id; nehareddy2807@gmail.com[3]

Abstract

Road traffic accidents remain a significant global concern, frequently caused by violations such as riding without helmets, neglecting seatbelts, carrying more than two passengers on two-wheelers, and using mobile phones while driving. Conventional monitoring methods depend heavily on manual inspection of surveillance footage, which is labor-intensive, time-consuming, and susceptible to human error, particularly in high-density traffic environments.This paper presents an AI-based system for automated detection of multiple traffic safety violations, including helmet absence, seatbelt non-compliance, triple riding, and mobile phone usage. The proposed approach utilizes computer vision and deep learning techniques to analyze real-time or recorded video streams from traffic surveillance cameras. A YOLO-based object detection framework is employed to identify vehicles, riders, drivers, protective gear, and handheld devices. Additionally, the system counts the number of occupants on two-wheelers to determine triple riding violations.Each video frame is processed using rule-based decision logic to identify unsafe behaviors such as missing helmets, lack of seatbelt usage, excessive riders, and mobile phone interaction by drivers. Detected violations are marked with bounding boxes and descriptive labels, while supporting evidence is stored along with timestamps for documentation and enforcement purposes.The proposed solution minimizes manual supervision, enhances detection accuracy, and enables simultaneous identification of multiple violations in real time. By assisting traffic authorities in automated rule enforcement, the system promotes responsible driving practices and contributes to improved road safety and accident reduction.

Keywords—*Traffic Safety, Road Traffic Violations, Computer Vision, Deep Learning, YOLO, Object Detection, Helmet Detection, Seatbelt Detection, Triple Riding Detection, Mobile Phone Usage Detection, Intelligent Transportation System (ITS), Video Surveillance, Real-Time Monitoring, Automated Traffic Enforcement.*

Introduction

Rapid urbanization and the continuous growth in vehicle population have intensified road safety challenges across modern cities. A considerable proportion of traffic accidents occur due to non-compliance with fundamental safety measures, including riding without helmets, failure to use seatbelts, triple riding on two-wheelers, and the use of mobile phones while driving. Such unsafe behaviors significantly increase the risk of severe injuries and fatalities.

Conventional traffic monitoring approaches primarily rely on manual observation of surveillance footage by traffic personnel. This method is labor-intensive, time-consuming, and highly dependent on human attention. As traffic density increases, maintaining continuous monitoring becomes difficult, leading to missed violations and delayed enforcement. The absence of intelligent automation further limits the effectiveness of these systems in ensuring road discipline.To overcome these limitations, this work proposes an AI-based detection system capable of identifying helmet violations, seatbelt non-compliance, triple riding, and mobile phone usage. The system employs computer vision and deep learning techniques to analyze live or recorded video streams obtained from traffic surveillance cameras. A YOLO-based object detection model is utilized to detect vehicles, riders, drivers, protective gear, and handheld devices. By automating the detection process, the proposed solution enhances accuracy, reduces human intervention, and supports real-time monitoring. Consequently, the system assists traffic authorities in enforcing regulations more efficiently and contributes to improved road safety.

Existing System

Current traffic enforcement mechanisms mainly depend on manual supervision of CCTV camera feeds. Traffic personnel either monitor live traffic conditions or review recorded videos to identify rule violations. Most surveillance infrastructures function solely as recording systems without built-in intelligence for automatic violation detection.Although some advanced implementations focus on specific tasks such as helmet detection or speed monitoring, they are typically limited to a single type of violation. These systems lack the capability to simultaneously detect multiple unsafe behaviors. Continuous human

observation is difficult to sustain over extended periods, particularly in high-traffic environments, increasing the likelihood of overlooked violations. Furthermore, delays in identifying violations reduce the effectiveness of enforcement actions Another limitation is the lack of centralized data management. Many existing setups do not provide automated storage, tracking, or reporting of violations, making analysis and decision-making inefficient. Overall, traditional monitoring systems are resource-intensive, less accurate, and difficult to scale for large urban traffic networks.

Problems in Existing System

Existing traffic monitoring systems rely heavily on manual inspection of surveillance footage, which is both time-consuming and labor-intensive. Traffic authorities are required to continuously observe video feeds to identify violations such as riding without helmets, not wearing seatbelts, triple riding, and mobile phone usage while driving. This approach significantly increases the workload on personnel and reduces overall operational efficiency. Moreover, prolonged monitoring leads to human fatigue, which directly affects concentration levels and results in missed violations and inconsistent enforcement. Another major limitation is scalability; as the number of vehicles and surveillance locations increases, manual supervision becomes impractical and inefficient for large-scale traffic management. These challenges highlight the need for an automated and intelligent system to ensure effective monitoring and enforcement.

Proposed System

The proposed system introduces an AI-driven traffic monitoring framework designed to automatically detect and analyze multiple traffic violations using video surveillance. The system processes live or recorded video streams through advanced computer vision techniques to identify vehicles, drivers, riders, helmets, seatbelts, and mobile phone usage. It also determines the number of occupants on two-wheelers to detect triple riding violations. A YOLO-based deep learning model is utilized for accurate object detection and classification in real time. The system categorizes traffic conditions into compliant and violation scenarios, including helmet absence, seatbelt non-compliance, triple riding, and mobile phone interaction while driving. Whenever a violation is detected, bounding boxes and descriptive labels are generated on the video frames, and supporting evidence is stored along with timestamps for documentation purposes. Additionally, the system can generate real-time alerts and can be integrated with traffic management platforms to facilitate faster enforcement actions. By automating the detection process, the proposed solution significantly reduces manual workload, improves detection accuracy, and promotes safer driving practices.

Requirements Analysis

The requirements analysis of the proposed AI-based traffic violation detection system is divided into functional and non-functional requirements to ensure efficient and reliable system performance.

The functional requirements define the key operations of the system, which is organized into three primary modules: User Module, Admin Module, and System Module. The User Module is designed for traffic authorities and includes functionalities such as user registration, secure login, viewing real-time violation alerts, and monitoring live CCTV video streams. The Admin Module provides administrative control, including administrator authentication, monitoring violations across multiple locations, and managing user accounts. The System Module is responsible for automated processing tasks, including capturing video input from surveillance cameras, detecting violations such as helmet absence, seatbelt non-compliance, triple riding, and mobile phone usage, and generating alerts when violations are identified.

The non-functional requirements focus on performance and quality attributes essential for system efficiency. The system must process video frames in real time with minimal delay to ensure timely detection of violations. High accuracy must be maintained under varying traffic densities and lighting conditions to minimize false detections. The user interface should be intuitive, displaying clear bounding boxes and labels for easy interpretation. Reliability is crucial, as the system must operate continuously for extended periods without failure. Security measures must be implemented to protect violation data and images through authentication and controlled access. Additionally, the system should be maintainable, allowing easy updates to detection models, datasets, and operational rules without requiring major architectural changes.

Computational Resource Requirements

The system requires adequate hardware and software resources to ensure smooth operation. The hardware requirements include a processor such as Intel Core i5 or higher, a minimum of 8 GB RAM, and at least 500 GB of storage capacity. A camera or video input source is also necessary for capturing surveillance footage. On the software side, the system is developed using Windows 10 or later as the operating system and Python 3.12 as the primary programming language. Frontend technologies such as HTML, CSS, and JavaScript are used to design the user interface, while a Python-based backend framework manages system operations. Additionally, annotation tools like LabelImg are used for dataset labeling and training the deep learning models.

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026

Life Cycle Model

The development of the proposed system follows the Waterfall model, a sequential software development approach where each phase is completed before moving to the next. This structured methodology ensures systematic planning, development, and evaluation of the system. The process begins with requirement analysis, where system needs such as traffic violation detection, real-time monitoring, and automated reporting are identified. This is followed by the system design phase, where the architecture, data flow, and module interactions are defined, and appropriate technologies such as Python, deep learning frameworks, and YOLO models are selected. During the implementation phase, various system modules are developed, including video processing, object detection, user interface, and database integration. The system is then tested using different traffic video samples to evaluate detection accuracy, performance, and reliability, and any identified issues are resolved. Finally, the system is deployed for operational use, and future enhancements such as model improvements and additional features can be incorporated during the maintenance phase.

System Design

Architecture

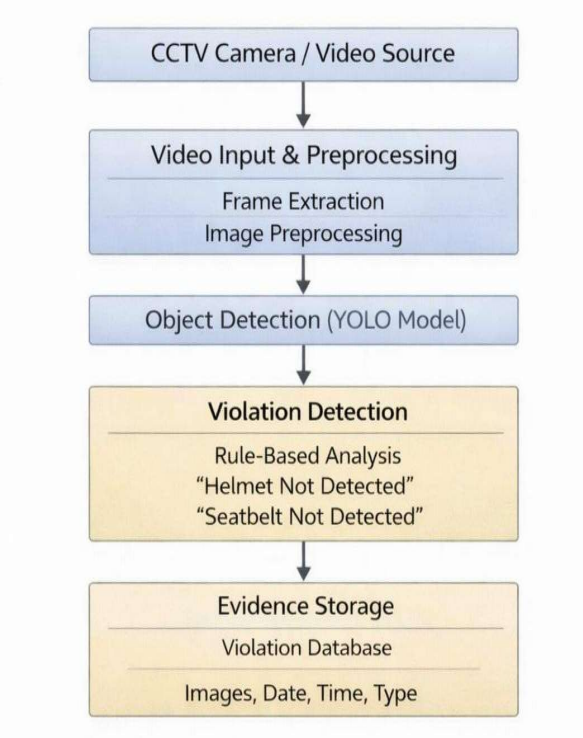

Fig 1 Software Architecture

The software architecture of the proposed system defines the interaction between different modules involved in traffic violation detection. The architecture consists of video input, processing module, detection model, violation analysis, database storage, and user interface.Initially, live or recorded video streams are obtained from surveillance cameras. These video frames are processed using image preprocessing techniques such as resizing and normalization. The processed frames are then passed to the YOLO-based object detection model, which identifies vehicles, riders, helmets, seatbelts, and mobile phones.

After detection, rule-based logic is applied to determine violations such as helmet absence, seatbelt non-compliance, triple riding, and mobile phone usage. Detected violations are highlighted with bounding boxes and labels. The results are stored in the database along with timestamps and location details. Finally, the processed output is displayed to users through a monitoring interface.

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026

Technical Architecture

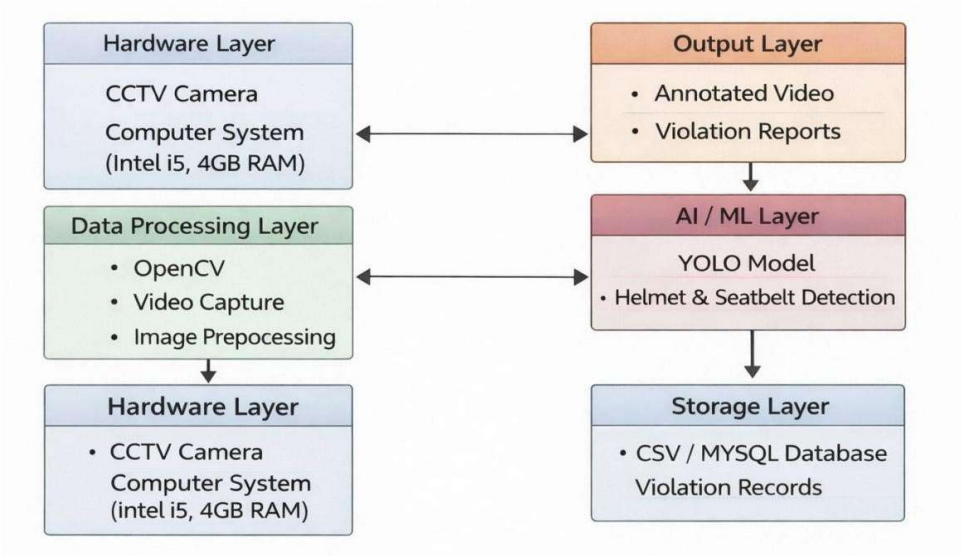

Fig 2 Technical Architecture

The technical architecture describes the hardware and software components required to implement the system. Surveillance cameras act as the input source for capturing traffic video. The captured video is transmitted to the processing unit where the detection model is deployed.The processing unit executes computer vision algorithms using Python and deep learning libraries. The YOLO model performs object detection, and the violation detection logic analyzes the results. Identified violations are stored in a structured database.The system also includes a web-based interface developed using HTML, CSS, and JavaScript, which allows users and administrators to monitor violations and access reports. Communication between modules is handled through backend services implemented in Python.

UML Diagrams

Unified Modeling Language (UML) is a standardized modeling language used to represent the structure and behavior of software systems. UML diagrams provide a visual representation of system components, interactions, and workflows. These diagrams help in understanding system functionality, improving design clarity, and supporting software development.

Use Case Diagram

The use case diagram illustrates the interaction between users and the system. It identifies the primary actors such as traffic authorities and administrators, along with the functionalities provided by the system.

Key use cases include login, monitoring live video, viewing detected violations, accessing reports, managing users, and storing violation data. The diagram also represents how different actors interact with system modules.

Class Diagram

The class diagram represents the static structure of the system by showing classes, attributes, methods, and relationships. It provides an overview of the software components and their interactions.

The main classes include User, Admin, VideoProcessor, DetectionModel, ViolationDetector, and Database. Each class contains relevant attributes and methods required for system functionality. Relationships such as association and dependency between classes are also illustrated.

Sequence Diagram

The sequence diagram describes the interaction between system components in chronological order. It illustrates how requests flow from the user to the system and how responses are generated.

The sequence begins with video input capture, followed by frame processing, object detection, violation analysis, and data storage. The final step involves displaying the annotated results to the user. This diagram helps in understanding the execution flow of the system.

Testing

Software testing is conducted to evaluate whether the developed application performs according to specified requirements and operates reliably under different conditions. It helps identify errors, improve system performance, and ensure stability before deployment. In the proposed AI-based traffic violation detection system, testing is performed to verify the accuracy of violation detection and proper functioning of modules such as user login, video processing, violation identification, and database

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026

storage.Testing plays a crucial role because the system operates on real-time video input and automated decision-making. Proper validation ensures that violations such as helmet absence, seatbelt non-compliance, triple riding, and mobile phone usage are detected accurately. Effective testing also improves reliability and ensures uninterrupted monitoring.

Stage of Testing

Unit Testing

Unit testing focuses on verifying individual components of the system independently. Modules such as user login, video capture, object detection, violation detection, and database storage are tested separately. This approach helps identify defects early and ensures each component functions correctly.

Integration Testing

Integration testing is performed after combining different modules. The interaction between video processing, YOLO-based detection, violation logic, and database storage is verified. This stage ensures that data flows correctly between components and that combined functionality operates without errors.

System Testing

System testing evaluates the complete application as a whole. The system is tested using various traffic video inputs to validate detection of helmet violations, seatbelt non-compliance, triple riding, and mobile phone usage. User interface functionality, alert generation, and data storage are also verified during this phase.

Acceptance Testing

Acceptance testing is conducted to confirm that the developed system meets user requirements. Traffic authorities or evaluators verify whether the system detects violations accurately, generates alerts in real time, and stores relevant information. After successful validation, the system is considered ready for deployment.

Types of Testing

Black Box Testing

Black box testing evaluates system functionality without considering internal implementation. Various video inputs are provided, and the outputs are checked to ensure that violations are detected correctly. This testing verifies system behavior from the user perspective.

White Box Testing

White box testing involves examining the internal logic and code structure. It is applied to detection algorithms, preprocessing steps, and violation identification logic. Techniques such as statement coverage and branch coverage are used to ensure proper implementation and improved reliability.

Test Cases

st Case ID	Scenario	Input	Expected Output	Actual Output	Status
TC01	Triple riding detection	Video with 3 people	Detect "Triple Riding"	Detected correctly	Pass
TC02	Normal riding (2 people)	Video with 2 riders	No violation	No violation detected	Pass
TC03	No helmet detection	Rider without helmet	Detect "No Helmet"	Detected correctly	Pass
TC04	Helmet worn	Rider with helmet	Rider with helmet	Extracted / UNKNOWN	Pass
TC05	Walking person	Pedestrian with helmet	Car driver without belt	Detected correctly	Pass
TC06	Mobile plate OCR	Clear number	Car drivler in frame	Detected correctly	Pass
TC07	Number plate OCR	Clear number plate image	Sour dile wet driver image	Detected nultiple	Pass
TC08	Database storage	Any violation	Run Flask app | Live	Whare Flask app	Pass
TC09	Webcam input	Live camera feed	Show violations table	Show violiations in folder	Pass
TC10	Webcam input	C4Γviolation	Process video IR	Working	Pass
TC11	Video input	Litv cam feed	Live camera feed	Process.wotencd	Partial
TC12	OpenCV window error	Run Flask app	Run Flask app	Process video open	Pass
TC13	Webcam input	Mitxcaamera feed	Mpkcam foee	Working	Acceptable
TC14	Blarsy.eboura.ton	Live camera feed	Live camera feed	Processrod in frame	Pass
TC15	Image lewdens fille	Multiple bikes in frame	Empty road video	Detect in real-time	Pass
TC16	Wecbcm input	Live camera feed	Process video	Working	Pass
TC17	Video input	Video input	Video input	Working	Pass

Test Cases

Test cases are designed to validate system behavior under different scenarios such as:

- Rider wearing helmet (no violation expected)
- Rider without helmet (helmet violation expected)
- Car driver without seatbelt (seatbelt violation expected)
- Two-wheeler with three riders (triple riding violation expected)

Conclusion

This work presented an AI-based real-time traffic violation detection system designed to identify helmet absence, seatbelt non-compliance, triple riding, and mobile phone usage using computer vision and deep learning techniques. The system processes surveillance video streams and automatically detects violations with minimal human intervention.By utilizing a YOLO-based object detection model, the proposed approach enables accurate identification of vehicles, riders, and safety equipment. Automated violation detection reduces dependency on manual monitoring, improves consistency, and supports real-time enforcement. The system also stores violation records with timestamps, which helps

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026

traffic authorities in documentation and analysis.The implementation demonstrates that intelligent video analytics can significantly improve traffic monitoring efficiency. The proposed solution contributes to safer driving practices by encouraging adherence to traffic regulations. Overall, the system provides a scalable and effective framework for enhancing road safety and reducing accident risks.

Screen Shots

Fig. 1 Detection of Triple Riding

Fig 2 Detection of Helmet

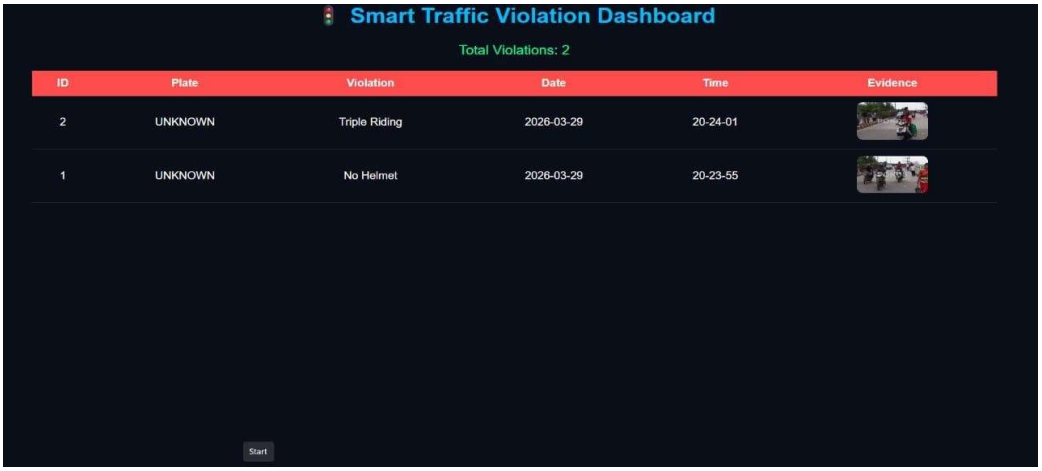

Fig. 3 Dashboard

Future Scope

The proposed system can be further enhanced by incorporating additional features and expanding deployment capabilities. Possible future improvements include:

- Development of a mobile application to provide real-time monitoring and instant notification alerts
- Integration of GPS-based location tracking for precise identification of violation locations
- Implementation of Automatic Number Plate Recognition (ANPR) for automated vehicle identification
- Support for multiple languages to improve accessibility for diverse users
- Enhanced authentication and role-based access control for improved security
- Integration with traffic police or government enforcement systems
- Use of cloud storage for centralized data management and scalability
- Advanced data analytics for traffic pattern analysis and decision-making
- Real-time dashboard for monitoring multiple camera feeds simultaneously

These enhancements can further improve automation, scalability, and operational efficiency of the proposed traffic monitoring system.

References

[1] R. Sharma and A. Gupta, "Real-Time Helmet Detection Using Deep Learning for Traffic Safety," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 5, 2024.

[2] P. Kumar and S. Verma, "Automatic Helmet Detection for Motorcyclists Using YOLO-Based Object Detection," *IEEE Access*, 2025.

[3] M. Reddy and K. Rao, "AI-Based Traffic Monitoring System for Helmet and Seatbelt Violation Detection," *IEEE International Conference on Artificial Intelligence and Smart Systems*, 2025.

[4] S. Patel and N. Singh, "Computer Vision-Based Seatbelt Detection for Driver Safety Using Deep Learning," *IEEE Transactions on Vehicular Technology*, vol. 74, 2026.

[5] A. Das and R. Mehta, "Intelligent Traffic Surveillance System Using YOLO for Road Safety Enforcement," *IEEE International Conference on Smart Transportation Systems*, 2026.