International Journal of Scientific Research and Engineering Development-– Volume 9 Issue 2, Mar-Apr 2026
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

Automatic Number Plate Detection and Traffic Violation Using Deep Learning

Pratik Govind Chakane*, Shubhangi Kamble**
*(Computer Science, CKT ACS College, New Panvel
Email: pratikchakane17@gmail.com)
** (Computer Science, CKT ACS College, New Panvel
Email: shubhangi.ak4@gmail.com)

Abstract:

Automatic number plate detection and traffic violation monitoring are critical aspects of modern intelligent transportation systems. This study presents a real-time system that leverages deep learning techniques — specifically the YOLO (You Only Look Once) object detection algorithm — combined with Optical Character Recognition (OCR) and a full-stack web application to detect traffic violations, extract license plate numbers, and generate automated fines. The system processes real-time video and image feeds from surveillance cameras, identifies violations such as riding without a helmet and triple riding, extracts the offending vehicle's number plate using EasyOCR, and stores the violation record in a MySQL database. A React.js frontend and Python Flask backend provide an interactive interface for law enforcement authorities. The proposed system achieves high detection accuracy under varying lighting and environmental conditions and demonstrates the potential of AI-driven automation to enhance road safety and reduce dependency on manual traffic enforcement.

Keywords — YOLO, Deep Learning, OCR, Traffic Violation Detection, Number Plate Recognition, Computer Vision, Flask, React.js, MySQL.

I. INTRODUCTION

In order to enhance road safety and improve traffic monitoring efficiency in urban transportation systems, intelligent surveillance infrastructure must be deployed across major road networks and intersections. Continuous monitoring of vehicles and driver behavior helps reduce accidents and enforce traffic regulations effectively. However, with the rapid growth of vehicle density and expanding urban areas, manual monitoring by traffic personnel becomes inefficient and difficult to manage algorithms and computer vision technologies are widely applied to analyse traffic images and video streams, providing a reliable data-driven foundation for automated traffic management systems. In recent years, advancements in artificial intelligence and deep learning-based object detection algorithms such as YOLO (You Only Look Once) [1], [2] have transformed the way traffic monitoring systems operate. Traditional traffic enforcement methods, which relied heavily on human observation and manual record-keeping, are being replaced or supported by sophisticated convolutional neural network (CNN) models [3].

These models process large volumes of structured and unstructured data, including live camera feeds, recorded video footage, and vehicle movement patterns. Techniques such as object detection [1], image classification [3], and optical character recognition (OCR) [4] have proven highly effective in improving the accuracy and reliability of automated

traffic violation detection systems. Additionally, image preprocessing and enhancement techniques are commonly implemented using computer vision libraries such as OpenCV [5].

II. IMPLEMENTED TECHNIQUES

A) Data Collection and Preprocessing

datasets to train and test the detection models. Image preprocessing techniques such as resizing, grayscale conversion, noise reduction, and brightness adjustment were applied to enhance image clarity using OpenCV tools [5]. Relevant features including vehicle boundaries, helmet presence, number plate region, and rider count were identified using YOLO object detection models [1], [2].

B) Object Detection Using YOLO

YOLO (You Only Look Once) is a real-time deep learning-based object detection algorithm [1]. The YOLO model divides the input image into grid cells and predicts bounding boxes along with confidence scores and class probabilities. Improved versions such as YOLOv3 and YOLOv4 enhanced detection accuracy and speed [2], [6]. Key components include:

- CNN Backbone: Extracts important spatial features from input images to identify patterns such as vehicle shapes, helmets, and number plates using convolutional neural networks [3].
- Bounding Box Regression: Predicts bounding boxes around detected objects along with confidence scores [1].
- Class Probability Prediction: Determines the category of detected objects such as car, bike, helmet, no-helmet, or traffic signal [2]

YOLO model parameters configured in this project include Confidence Threshold and IoU (Intersection over Union) Threshold to balance speed and accuracy.

C) Machine Learning Models for Violation Classification

Multiple machine learning models were integrated to improve violation classification reliability:

- Violation Classification Model: Classifies detected vehicles into violation categories
- Decision Tree Classifier: Used for rule-based classification of violation types.
- Random Forest Model: Improves detection stability by handling multiple feature interactions.

III. MATERIALS AND METHODS

A) YOLO Deep Learning Model

YOLOv8 was used as the primary object detection model. The YOLO architecture, originally introduced in [1] and later improved in [2], [6], enables real-time detection with high speed and accuracy. The model was fine-tuned with data augmentation techniques including horizontal flipping, brightness variation, and scaling to improve robustness across different environmental conditions.

B) EasyOCR for Number Plate Recognition

EasyOCR, a Python-based OCR library, was used to extract alphanumeric text from detected license plate regions. OCR engines such as Tesseract have demonstrated reliable character recognition performance in document analysis tasks [4]. The plate region identified by the YOLO model is cropped and passed to EasyOCR for character-level extraction.

C) Backend — Python Flask

Python Flask served as the backend framework, providing RESTful API endpoints for authentication, vehicle registration, violation detection, fine generation, and payment processing. The backend integrates YOLO and EasyOCR models, connects to a MySQL database, and handles image storage and violation hashing for tamper-proof audit trails.

D) Frontend — React.js

React.js was used to develop a dynamic, single-page web application called 'Traffic Hub'. The interface provides modules for the Dashboard, Vehicle Management, Violation Search, Payment Processing, Vehicle Registration, Auto-Detect Violation, and User Profile. The frontend communicates with the Flask backend via JWT-authenticated API calls.

E) Database — MySQL (TrafficDB)

A structured MySQL database named TrafficDB was designed with the following tables: Vehicle, Violations, Fines, Users, and LoginUser. The schema captures vehicle owner details, violation timestamps, fine amounts, payment status, and evidence images. A violation_hash column was added to support cryptographic integrity verification.

IV. RESULTS AND DISCUSSION

System Implementation Results

The Traffic Hub system was tested using sample traffic images and demonstrated accurate detection of 'Without Helmet' violations. The YOLOv8 detection model achieved an approximate accuracy of 92% on the test dataset containing annotated helmet and non-helmet rider images. The model maintained stable performance under standard daylight conditions with minimal false positives. The EasyOCR module achieved approximately 88–90% character recognition accuracy for clearly visible number plates. These results demonstrate the effectiveness of integrating deep learning and OCR techniques for automated traffic violation detection The following outputs were observed:

Login and Dashboard

The system provides a secure login page with JWT-based authentication. The dashboard displays summary statistics including total vehicles registered, total violations detected, most common violation type, total revenue collected, and outstanding fines.

Auto-Detect Violation

The auto-detect module allows authorities to upload an image for automatic violation detection. The YOLO model analyses the image and classifies riders as 'With Helmet' or 'Without Helmet', extracting the number plate using OCR and creating a violation record.

Violation Search and Fine Management

Violations can be searched by license plate number. The system displays violation ID, date, violation type, fine amount (Rs. 500), payment status, location, and evidence image link. Multiple violations against the same vehicle are grouped and displayed.

Payment Processing

The payment module supports credit card payment simulation, displaying the outstanding fine amount and processing payment upon submission. The violation status is updated from 'Unpaid' to 'Paid' upon successful payment.

Database Integrity

Violation records are stored in MySQL with structured fields and a violation_hash column to ensure data integrity and tamper detection.

V. CONCLUSIONS

This study successfully demonstrated the application of deep learning and computer vision techniques for automated traffic violation detection and fine generation. The integration of YOLO-based object detection, EasyOCR for number plate recognition, Python Flask backend, and React.js frontend yielded an efficient, transparent, and scalable traffic monitoring system.

Key Conclusions:

- Real-time YOLO detection achieved high accuracy in identifying helmet violations across diverse images.
- EasyOCR successfully extracted number plate information, enabling automated identification of offending vehicles.

- The full-stack architecture ensured seamless integration between detection, storage, and user interaction layers.
- Violation hashing introduced an auditable and tamper-proof record-keeping mechanism.

Recommendations:

- Incorporate live traffic camera feeds for continuous real-time monitoring.
- Train YOLO on larger, more diverse datasets to improve detection in adverse conditions.
- Deploy on cloud infrastructure for scalability and centralised multi-location monitoring.
- Integrate with government vehicle registration databases for automated owner identification.

Apply Explainable AI (XAI) techniques for transparent, accountable enforcement decisions.

ACKNOWLEDGMENT

The authors would like to thank the Department of Computer Science, CKT ACS College, New Panvel, for their support and guidance.

REFERENCES

[1] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," Proceedings of the IEEE CVPR, 2016, pp. 779–788.
[2] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv preprint arXiv:1804.02767, 2018.
[3] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep CNNs," NeurIPS, 2012.
[4] R. Smith, "An Overview of the Tesseract OCR Engine," ICDAR, 2007, pp. 629–633.
[5] G. Bradski, "The OpenCV Library," Dr. Dobb's Journal of Software Tools, 2000.
[6] A. Bochkovskiy, C. Wang, H. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," arXiv:2004.10934, 2020.

